The European Conference
on Machine Learning and Principles
and Practice of Knowledge Discovery in Databases

ECML/PKDD 2014
PhD Session Proceedings

September 15-19, 2014
Nancy, France

Edited by

Radim Belohlavek
Bruno Crémilleux

Program Committee

Program Committee Co-Chairs

Radim Belohlavek	(Palacky University, CZ)
Bruno Crémilleux	(University of Caen, FR)

Program Committee Members

Stephane Canu	(INSA Rouen, FR)
Peggy Cellier	(INSA Rennes, FR)
Sanjay Chawla	(University of Sydney, AU)
Carlotta Domeniconi	(George Mason University, VA, US)
Antoine Doucet	(University of Caen, FR)
Richard Emilion	(University of Orleans, FR)
Georgiana Ifrim	(University College Dublin, IE)
Dino Ienco	(IRSTEA, FR)
François Jacquenet	(University Jean Monnet, Saint-Etienne, FR)
Jiri Klema	(Czech Technical University, Prague, CZ)
Rudolf Kruse	(Otto von Guericke University Magdeburg, DE)
Marzena Kryszkiewicz	(Warsaw University of Technology, PL)
Sergei Kuznetsov	(National Research University HSE, Moscow, RU)
Donato Malerba	(University of Bari, IT)
Pauli Miettinen	(Max-Planck-Institut fr Informatik, DE)
Amedeo Napoli	(CNRS & INRIA Nancy Grand Est)
Sergei Obiedkov	(National Research University HSE, Moscow, RU)
Panagiotis Papapetrou	(Stockholm University, SE)
Chedy Rassi	(INRIA Nancy Grand Est)
Sebastian Rudolph	(University of Karlsruhe, DE)
Dan Simovici	(University of Massachusetts, Boston, MA, US)
Arnaud Soulet	(University Francois Rabelais of Tours, FR)
Gerd Stumme	(University of Kassel, DE)

Table of Contents

Papers with Oral Presentations

Papers with Poster Presentations

Preface

The present volume contains papers accepted for oral and poster presentation in the PhD Session of the 2014 EMCL/PKDD conference held September 15-19, 2014 in Nancy, France. The objective of the session was to provide an environment for students to exchange their ideas and experiences with peers and to get constructive feedback from senior researchers in machine learning, data mining and related areas. The topics for discussion were mainly ideas of students and their ongoing work in preparation of their PhD dissertations in machine learning and data mining.

The PhD session received 23 initial submissions from which 8 were finally accepted for oral presentation and 11for poster presentation. All submissions were reviewed typically by three reviewers on the basis of their originality, quality, significance, and presentation. The program of the PhD session consisted of oral and poster presentations of accepted papers and an invited talk by Bart Goethals (University of Antwerp, Belgium).

We would like to express our thanks to the authors who submitted their papers, to the invited speaker, and to the members of the Program Committee, who all helped make the PhD Session a successful event. We also thank Martin Trnecka (Palacky University, Olomouc) for his help in preparing this proceedings. Finally, we wish to thank the local organization team of ECML/PKDD 2014, and in particular Amedeo Napoli and Chedy Raïssi.

September 2014

Radim Belohlavek and Bruno Crémilleux
ECML/PKDD 2014 PhD Session Co-Chairs

Search for User-related Features in Matrix Factorization-based Recommender Systems

Marharyta Aleksandrova[1,2], Armelle Brun[1], Anne Boyer[1], and Oleg Chertov[2]

[1]Université de Lorraine - LORIA,
Campus Scientifique, 54506 Vandoeuvre les Nancy, France
[2]National Technical University of Ukraine "Kyiv Polytechnic Institute",
37, Prospect Peremohy, 03056, Kyiv, Ukraine
{marharyta.aleksandrova,armelle.brun,anne.boyer}@loria.fr,
chertov@i.ua

Abstract. Matrix factorization (MF) is one of the most powerful approaches used in the frame of recommender systems. It aims to model the preferences of users about items through a reduced set of latent features. One main drawback of MF is the difficulty to interpret the automatically formed features. Following the intuition that the relation between users and items can be expressed through a reduced set of users, referred to as representative users, we propose a simple modification of a traditional MF algorithm, that forms a set of features corresponding to these representative users. On one state of the art dataset, we show that proposed representative users-based non-negative matrix factorization (RU-NMF) discovers interpretable features and does not significantly decrease the accuracy on test with 10 and 15 features.

Keywords: Recommender systems, matrix factorization, features interpretation.

1 Introduction

Recommender systems, that belong to machine learning area, aim to estimate preferences (ratings) of target users on previously non-seen items, in order to recommend them those items, which would probably satisfy these target users [1]. Recommendation algorithms are used in a wide area of real services starting from electronic commerce to considering search engines as a special type of recommender systems.

In order to estimate unknown preferences, recommender systems can use information about the content of the items (content-based methods), preferences of other users (collaborative filtering) or the both sources (hybrid approaches) [1]. In the frame of collaborative filtering we can outline such approaches as neighborhood-based [1] and matrix factorization-based techniques [2]. The first approach searches for neighbor users, who have similar preferences as the target user and recommends items that were highly rated by his neighbors. Thus,

if needed, recommendation can be easily explained: a certain item was recommended because it was highly rated by the users having similar tastes to the active one.

Matrix factorization relies on the idea that there is a small number of latent factors (features) that underly the preferences (interactions) between users and items. As these features are defined so as to fit at best the data, no obvious interpretation can be made of them and as a result, unlike neighborhood-based approaches, recommendations have no obvious explanation. However, as it was shown in [3], providing explainable recommendation remains important for the users fidelity.

Based on the assumption that the preferences between users are correlated, we assume that within the entire set of users, there is a small set of users that have a specific role or have specific preferences. These users can be considered as representative of the entire population and we intend to discover features that are associated with these representative users. We think that if the discovered features would represent elements from the real world, they could not only be interpretable, but the recommendations could also be easily explained, similarly to the explanation provided by neighbor-based approaches, where neighbors are replaced by representative users. In order to identify these representative users we propose a representative users-based non-negative matrix factorization approach (RU-NMF), which is a slight modification of traditional non-negative matrix factorization technique based on multiplicative update rules.

This paper is a part of the work in progress about the discovery of features related to reality in matrix factorization-based approaches. In the current research we rise two main questions: (a) can MF algorithms result in features that can be associated with real users? (b) will the quality of recommendations be reduced if such associations are made explicit? We also point out some potential benefits of the resulting model. To the best of our knowledge, this work is the first one that is interested in not only interpreting features in MF-based recommendation approaches, but also in constraining these features so that they correspond to real elements of the system.

2 Related works

Let M be the number of users and N the number of items. The interaction between these entities is usually represented under the form of a matrix R ($\dim(R) = M \times N$) with elements r_{mn} corresponding to the rating assigned by the user m to the item n. Thus the recommendation problem is reduced to the task of estimating the missing values in R.

MF techniques decompose the original rating matrix R into two low-rank matrices U ($\dim(U) = K \times M$) and V ($\dim(V) = K \times N$) in such a way that the product of these matrices approximates the original rating matrix $R \approx R^* = U^T V$ with respect of the condition of minimal loss. This task is usually solved by optimization methods, such as gradient descent or alternating least squares [4]. The set of factors K can be seen as a joint latent space on which a mapping

of both users and items spaces is performed [4]. Thus matrices U and V can be considered as transfer matrices to the new feature space from the spaces of users and items respectively. MF techniques have recently attracted more attention than traditional neighborhood-based approaches [1], as they are adequate for large-scale and sparse datasets [5] and have proven to result in models of both low-complexity and good accuracy (see Netflix Prize competition [4, 6]).

Features resulting from factorization usually do not have any physical sense, what makes resulting recommendations less explainable. Some authors made attempts to interpret them by using non-negative matrix factorization based on multiplicative update rules (further referred to as NMF). NMF imposes the condition of non-negativity on the values of matrices U and V, to ensure that each user profile can be represented as an additive linear combination of coordinates [7]. [8, 7] assumed that the features formed can be related to behavioral patterns, or to groups of users. However, the interpretation of each feature is not so easy to perform as it has to be discovered manually, by analyzing the content of the matrices. Authors of [9] focused on the explanation of the recommendations with MF techniques. With this aim, MF and neighborhood-based approaches are combined through weighting schemes. Nevertheless, such a method allows only partial explanation of the recommendations.

So we can conclude that relatively few works concern feature interpretation in MF-based recommendation techniques and the proposed approaches either require human post-processing or provide only partial interpretation. Still such an interpretation could be useful not only for understanding and characterizing the relation between users and items but also to make recommendations explainable.

3 The proposed approach: RU-NMF

3.1 Preliminaries

Let us consider 2 linear spaces L_1 and L_2 of dimensionality respectively 6 and 3, with basic vectors in canonical form $\{\boldsymbol{u}_m\}$, $m \in \overline{1,6}$ and $\{\boldsymbol{f}_k\}$, $k \in \overline{1,3}$. Let the transfer matrix from L_1 to L_2 be specified by matrix (1).

$$P = \begin{pmatrix} 0 & 0 & p_{13} & p_{14} & 1 & p_{16} \\ 1 & 0 & p_{23} & p_{24} & 0 & p_{26} \\ 0 & 1 & p_{33} & p_{34} & 0 & p_{36} \end{pmatrix} \tag{1}$$

$\boldsymbol{u}_5$, $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are direct preimages of $\boldsymbol{f}_1$, $\boldsymbol{f}_2$ and $\boldsymbol{f}_3$ respectively. Indeed, $P\boldsymbol{u}_5 = P\begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}^T = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^T = \boldsymbol{f}_1$. By analogy, $P\boldsymbol{u}_1 = \boldsymbol{f}_2$, $P\boldsymbol{u}_2 = \boldsymbol{f}_3$. At the same time vectors $\boldsymbol{u}_3$, $\boldsymbol{u}_4$ and $\boldsymbol{u}_6$ will be mapped into linear combinations of basic vectors $\boldsymbol{f}_1$, $\boldsymbol{f}_2$ and $\boldsymbol{f}_3$. For example, $P\boldsymbol{u}_3 = p_{13}\boldsymbol{f}_1 + p_{23}\boldsymbol{f}_2 + p_{33}\boldsymbol{f}_3$ presents the linear combination for $\boldsymbol{u}_3$.

3.2 RU-NMF

As mentioned previously, matrix U can be considered as a transfer matrix from the space of users to the space of features. Analyzing the example considered

above, we can say that if matrix U has a form similar to (1), *i.e.* U has exactly K unitary columns with one non-zero and equal to 1 element on different positions, then the users corresponding to these columns are direct preimages of the K features. We say they represent the canonical coding of the features, following [10]. The features can thus be directly interpreted as users. These users will be referred to as representative users.

Obviously, in the general case, one cannot guarantee that the matrix U will be in a form similar to matrix (1). Worse, none of the column-vectors of matrix U may directly represent the canonical form of a feature. However we could design a matrix factorization approach that imposes appropriate constraints. In our case, the constraints would be the following: K columns in U have to represent the canonical coding of K different features. In order to solve this problem we propose the RU-NMF approach, that forms both matrices U and V, with features corresponding to representative users. The whole process consists of 6 steps, further detailed below.

Step 1. A traditional matrix factorization is performed, resulting in both matrices U and V with K features. As in [8, 7], that were also focusing on the interpretation of features, we used non-negative matrix factorization based on multiplicative update rules.

Step 2. A normalization of each of the M column vectors of the matrix U is performed so as to result in unitary columns. The resulting normalized matrix is denoted by U_{norm} and the set of normalization coefficients by C.

Step 3. This step is dedicated to the identification of the representative users in the U_{norm} matrix. We will consider user u_m as the best preimage candidate for the feature f_k if the vector in matrix U_{norm} corresponding to the user u_m will be the closest to the corresponding canonical vector (a vector with the only one non-zero and equal to 1 value on position k). The notion of closeness between vectors is expressed in Euclidean distance. That is the task of finding representative user u_m is reduced to solving the optimization problem (2).

$$dist(f_k, u_m^{norm}) \rightarrow \min \tag{2}$$

where u_m^{norm} is the vector from matrix U_{norm} corresponding to the user u_m. Let us consider the following example. Assume that we have vector α of the form $\begin{pmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_K \end{pmatrix}^T$ with unique norm ($\sqrt{\alpha_1^2 + \alpha_2^2 + \ldots + \alpha_K^2} = 1$). Then the distance between α and the first canonic vector $f_1 = \begin{pmatrix} 1 & 0 & \dots & 0 \end{pmatrix}^T$ is expressed by $dist^2(f_1, \alpha) = (1 - \alpha_1)^2 + \alpha_2^2 + \ldots + \alpha_K^2$. Performing simple mathematical transformations we can obtain equation (3).

$$dist^2(f_1, \alpha) = 2(1 - \alpha_1) \tag{3}$$

This means that the minimum of the distance is obtained under the condition $\alpha_1 \rightarrow \max$. Taking into account this reasoning, we consider a user u_m as a preimage candidate for the feature f_k if the maximum value of appropriate vector u_m^{norm} is situated on the position k; and the best preimage candidate is the one among all candidates with the highest maximum.

Let us analyze what can be the highest value of distance (2) between a canonic vector f_k and a preimage candidate vector u_m^{norm}. We have already noted that maximum value of the candidate vector u_m^{norm} is situated on the position k, otherwise u_m will be considered as a preimage candidate for another feature. Considering the formula (3) we can say that the maximum of distance is reached when the maximum value of the vector u_m^{norm} is as small, as possible. Obviously this condition holds only for the vector u_m^{norm} with all equal values $\left(\frac{1}{\sqrt{K}}\ \frac{1}{\sqrt{K}}\ \cdots\ \frac{1}{\sqrt{K}}\right)^T$, where K is the dimensionality of u_m^{norm}. In this case distance will be equal to $dist^{max} = \sqrt{2(1-\frac{1}{\sqrt{K}})}$.

As we can see maximal value of distance depends on the dimensionality of the feature space. So, in order to unify characteristics of representative users considering different number of features and for simplicity of analysis, we propose to use the quality score (4) for the identification of representative users and their characterization.

$$q(u_m) = 1 - \frac{dist(f_k, u_m)}{dist^{max}(K)} \tag{4}$$

The highest quality score, namely 1, is assigned to a candidate with a vector equal to the canonical coding as the appropriate distance is equal to 0. The lowest quality score, namely 0, is assigned to candidates with no influencing coordinates (all values are equal).

Thus on the third step all users are divided into subgroups of preimage candidates for each feature (according to the position of maximal value in u_m^{norm}). After this, the user with the highest quality score among all candidates is considered as the representative one for the current feature.

Once all preimages are identified, the matrix U_{norm} is modified so as to obtain a matrix in a form of (1): in each column that corresponds to a representative user, a 1 is assigned to the coordinate at the position of the maximum value and a 0 is assigned to all others. The resulting modified matrix is the matrix U_{norm}^{mod}.

In some cases, a feature, say feature f_k, may have no candidate preimage. In this case we can either decrease the number of features considered for factorization or search for a vector with the second maximum situated on that specific position.

Step 4. Each column of the matrix U_{norm}^{mod} is multiplied to the appropriate normalization factor from the set C resulting in matrix U^{mod}. After this, representative users will remain preimages of the features but with scaling coefficients.

Step 5. In order to obtain the best model we also have to modify the matrix V under the condition of minimal loss. The modification of V can be performed using optimization methods with the starting value obtained during the first step. As the objective of this paper is to determine the relevance of finding preimages of the features and to quantify the decrease of the quality of the recommendations, we did not consider this step.

Step 6. The resulting recommendation model is made up of matrices U^{mod} and V according to the formula $R^* = \left(U^{mod}\right)^T V$.

4 Experimental results

4.1 Datasets and Evaluation

In order to evaluate RU-NMF, we perform experiments on the 100k MovieLens dataset[1], which contains 100k ratings, ranging from 1 to 5, assigned by 943 users to 1682 movies (items). In all experiments, 80% of the ratings are used for learning the model and 20% for testing it. We prepare 30 different pairs of learning and test sets with the first one randomly chosen from the original 100k MovieLens dataset and the second one made up of the remaining part. The accuracy of the models are evaluated with two classical measures: mean absolute error (MAE) and root mean square error (RMSE) [11].

4.2 Quality of the Representative Users

In this section we answer the question (a), namely if MF algorithms can result in features that can be associated with real users. For each of the 30 datasets, we perform NMF with 10, 15 and 20 features. After that, for each run we order the representative users by their quality score and compute the mean quality at each rank (considering the number of features used for factorization). The corresponding values are presented in figure 1. When the number of features is equal to 10, the quality score of the representative users is particularly high: 90% of them have a quality score higher than 0.8 and 60% have a score higher than 0.9. With 15 and 20 features the quality of representative users decreases. For example, when the number of features is equal to 15, only half of the vectors corresponding to representative users have a quality score above 0.8 and only 20% above 0.9. With 20 features only 30% of the representative users have a quality above 0.8 and 10% above 0.9. As a result we can say that when the number of features is equal to 10 NMF naturally results in features, which are very close to the searched canonic form, that means in features that can be interpreted as real users.

4.3 Traditional NMF versus RU-NMF

In this subsection we seek an answer to the question (b) will RU-NMF have a considerable impact on the accuracy of the recommendations. The left part of the table 1 presents the resulting mean and standard deviation values of both errors (MAE and RMSE) on the 30 datasets for NMF with 10, 15 and 20 features. The mean error value, for both MAE and RMSE, goes down on the learning set when the number of features grows up. In contrast, on the test set the errors increase with the number of features. This fact confirms the overfitting problem mentioned in many works [12] and partially supports the conclusion of [9] that the more adequate number of features on the MovieLens dataset is close to 10. We can mention that on the test set, standard deviation seems to decrease as the number of features increases, confirming the consistent increase in the error.

[1] http://grouplens.org/datasets/movielens/

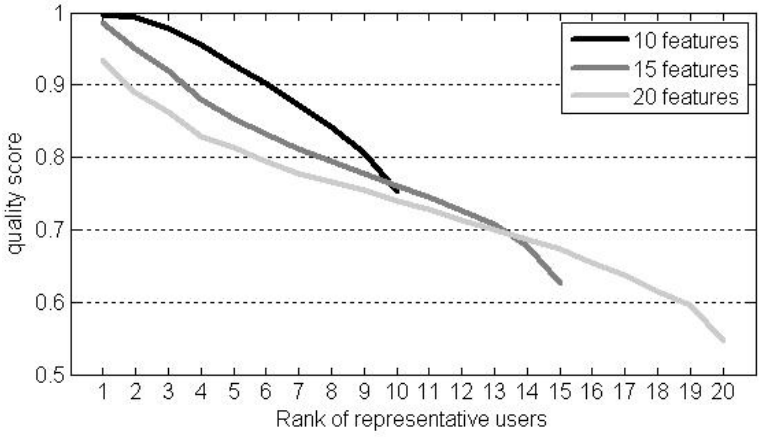


Fig. 1. Quality score for 10, 15 and 20 features.

Table 1. NMF vs RU-NMF: mean and standard deviation values of errors with 10, 15 and 20 features on learning and test sets.

	NMF					RU-NMF			
	Learning set		Test set			Learning set		Test set	
	MAE	RMSE	MAE	RMSE		MAE	RMSE	MAE	RMSE
	10 features					10 features			
mean	0.5482	0.7154	0.8014	1.0507	mean	0.5491	0.7168	0.8018	1.0512
std	0.0019	0.0019	0.0067	0.0096	std	0.0021	0.0022	0.0067	0.0096
	15 features					15 features			
mean	0.4933	0.6529	0.8393	1.1035	mean	0.4982	0.6613	0.8417	1.1071
std	0.0017	0.0020	0.0046	0.0068	std	0.0025	0.0039	0.0047	0.0071
	20 features					20 features			
mean	0.4461	0.5988	0.8689	1.1412	mean	0.4585	0.6232	0.8749	1.1505
std	0.0011	0.0012	0.0057	0.0063	std	0.0029	0.0062	0.0060	0.0069

The right part of the table 1 presents the mean and standard deviation values of the two error measures, computed on the same datasets and the same number of features for RU-NMF. We can see similar dependences as for the traditional NMF: both errors decrease on the learning set while the number of features grows, and both errors increase on the test set. As on NMF, standard deviations seem to decrease on the test set. We can conclude that RU-NMF preserves almost the same characteristics as traditional NMF.

Next we compare the accuracies of RU-NMF and NMF. The accuracy loss ρ, defined by formula (5), computes the relative difference between the error obtained with RU-NMF ($err\,(RU\text{-}NMF)$) and the error obtained with traditional NMF ($err\,(NMF)$). A positive loss value means that NMF performs better than RU-NMF. Table 2 reports the accuracy loss ρ, computed on the same 30 datasets.

$$\rho = \frac{err\,(RU\text{-}NMF) - err\,(NMF)}{err\,(NMF)} 100\% \tag{5}$$

Table 2. Accuracy loss ρ between RU-NMF and the traditional NMF, for 10, 15 and 20 features, %.

	10 features				15 features				20 features			
	mean	std	min	max	mean	std	min	max	mean	std	min	max
Learning set												
MAE	0.17	0.09	0.03	0.38	0.98	0.34	0.49	1.71	2.78	0.67	1.38	4.27
RMSE	0.19	0.10	0.03	0.46	1.29	0.49	0.61	2.38	4.08	1.08	1.94	6.64
Test set												
MAE	0.05	0.06	**-0.06**	0.18	0.29	0.19	**-0.06**	0.77	0.70	0.27	0.13	1.43
RMSE	0.05	0.07	**-0.07**	0.20	0.33	0.20	**-0.04**	0.79	0.82	0.31	0.12	1.53

The first conclusion that we can make when analyzing table 2 is that the accuracy loss increases with the number of features, on both learning and test sets, and for both error measures. In the worst case, the accuracy loss equals to 6.64%, for RMSE with 20 features, which is quite small. The lowest accuracy loss (0.05%) is obtained with 10 features for both errors. Standard deviation holds the same dependence: on test and learning sets, the accuracy loss is the least dispersive with 10 features. When comparing the accuracy loss between test and learning sets, we can note that the average loss is 3 times lower on test than on learning, for both errors and for all the number of features: thus we can say that RU-NMF has a lower relative loss between learn and test compared to NMF. A thorough analysis of the losses obtained on the 30 sets has shown that the accuracy loss on the test set is lower than the one on the learning set, in all cases, whatever is the error and the number of features. In some runs, RU-NMF has even a higher accuracy than NMF (see values in bold in table 2). This holds for 23% and 3% of the runs with 10 and 15 features respectively.

In order to estimate if the loss in accuracy between RU-NMF and NMF is statistically significant, we perform a statistical test. The null hypothesis H_0 denotes "The loss in error between NMF and RU-NMF is null". Student's tests with 99% confidence ($\alpha = 0.01$) have been performed and the results are presented in Table 3. In this table "H_0" represents the acceptance of the hypothesis and "-" its rejection. Considering 10 features on both learning and test sets and 15 features on the test set, both MAE and RMSE are not increased by RU-NMF (for example, the p-values with 10 features on MAE is equal to 0.8072). The null hypothesis is thus accepted for these numbers of features. In other cases, the errors on RU-NMF and traditional NMF models can not be considered as equal.

Considering this, we can conclude that a number of features equal to 10 provides not only the smallest values of errors on the test set, but also results in the representative users of the highest quality. Thus the quality of representative users can be considered as one of the potential indicators of the optimal number of features (the number of features, that results in the smallest error on the test set and that, thus, must be used in the factorization process). Also it may mean that an inverse logical conclusion takes place, notably representative users will be of a high quality only if the number of features is close to the optimal one.

Table 3. Results of Student's test with hypothesis H_0: "The loss in error between NMF and RU-NMF is null" for $\alpha = 0.01$.

	Learning set		Test set	
	MAE	RMSE	MAE	RMSE
10 features	H_0	H_0	H_0	H_0
15 features	-	-	H_0	H_0
20 features	-	-	-	-

5 Discussions and future work

This paper proposes a simple modification of the traditional matrix factorization approach (RU-NMF), that aims at forming not only interpretable features, but also features that represent elements from the reality (users). This work is a preliminary one and its main goal is to show that such features can be formed.

We have shown that the features resulting from a traditional approach (NMF) when the number of features is close to the optimal are naturally close to the canonic form. Thus the model can be slightly modified so as to correspond to real users, resulting in a small loss in the accuracy. When the number of features is equal to 10 and 15, this loss is even not statistically significant on the test set. Also it was shown that RU-NMF mainly preserves the same characteristics as traditional NMF. Thus the both questions raised in this paper were answered. The analysis of the accuracy loss has shown that the features formed by RU-NMF consistently disturb the accuracy on the test set less than on the learning set. This can be considered as a potential ability of factorization techniques with features related to reality to limit overfitting problem faced by many others.

From our point of view, the proposed interpretation has several important positive impacts on the way the model can be exploited. First, if such an interpretation can be made, the recommendations can be easily explained. Indeed, if each feature corresponds to a representative user, then the matrix V expresses preferences of representative users on items. Meanwhile, each line in the matrix U reveals interactions between the user, corresponding to this line, and a set of representative users. Thus each user of the population is linearly mapped on the basis related to representative users and the preferences of the latter ones are used to estimate the ratings of the whole population. That makes the recommendation process ideologically close to the neighborhood-based approaches with representative users used instead of neighbors. Second, as the estimated ratings of all users of the population are computed through the representative users, the latter can be viewed as mentor users in the population: the users who represent the preferences of entire population. They can also be viewed as the users to choose in poll studies. They can thus also be considered as those to be tracked, so as to follow the evolution of the preferences of the population. Finally, the approach we propose for this interpretation is automatic, it does not require any human expertise, unlike of other works focused on features interpretation.

In a future work, we would like to focus first of all on the verification of the hypothesis that users associated with the features can be really considered as representative ones. We consider that it can be done while solving the new item cold-start problem. Indeed, knowing preferences of identified users on a new item and their relations with all other users of the population (what is represented by matrix U) we can try to predict ratings of other users on this item. Accuracy of the resulting predictions will indicate if feature-related users can actually represent the whole population. At the opposite of many state of the art approaches that aim at tackling the cold-start problem, this one also requires no information about the content of the items. Second revealed properties of RU-NMF should be verified on other datasets. We would also like to investigate if other factorization techniques (such as those, based on gradient descent and alternating least squires) will result in features, that can be interpreted as real users.

References

1. G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art," *IEEE tran. on knowledge and data engineering*, vol. 17, no. 6, pp. 734–749, 2005.
2. M. Wu, "Collaborative filtering via ensembles of matrix factorizations," in *Proceedings of KDD Cup and Workshop*, vol. 2007, 2007.
3. R. Sinha and K. Swearingen, "The role of transparency in recommender systems," in *CHI'02 extended abstracts on Human factors in computing systems*. ACM, 2002, pp. 830–831.
4. Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.
5. G. Takacs, I. Pilaszy, B. Nemeth, and D. Tikk, "Scalable collaborative filtering approaches for large recommender systems," *Journal of Machine Learning Research*, vol. 10, pp. 623–656, 2009.
6. ——, "On the gravity recommendation system," in *Proc. of KDD cup and workshop*, 2007.
7. S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization." in *Proc. of the 6th SIAM Conf. on Data Mining*, 2006.
8. J.-F. Pessiot, V. Truong, N. Usunier, M. Amini, and P. Gallinari, "Factorisation en matrices non-négatives pour le filtrage collaboratif," in *3rd Conf. en Recherche d'Information et Applications (CORIA 2006)*, 2006, pp. 315–326.
9. A. Moin and C. Ignat, "Hybrid weighting schemes for collaborative filtering," INRIA, Tech. Rep., 2014.
10. Y. Guermeur, A. Lifchitz, and R. Vert, *Kernel Methods in Computational Biology*. MIT Press, 2004, no. 9, ch. A kernel for protein secondary structure prediction, pp. 193–206.
11. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 5–53, 2004.
12. M. Graus, "Understanding the latent features of matrix factorization algorithms in movie recommender systems," Master's thesis, University of Eindhoven, Master of Science in Human-Technology Interaction, 2011.

Optimistic Active Learning for Classification

Timothé Collet[1] and Olivier Pietquin[2]

[1] Supelec, MaLIS Research group, GeorgiaTech-CNRS UMI 2958, France
timothe.collet@supelec.fr
[2] University Lille 1, LIFL (UMR 8022 CNRS / Lille 1), SequeL Team, France
olivier.pietquin@univ-lille1.fr

Abstract. In this paper, we propose to reformulate the active learning problem occurring in classification as a sequential decision making problem. We particularly focus on the problem of dynamically allocating a fixed budget of samples. This raises the problem of the trade off between exploration and exploitation which is traditionally addressed in the framework of the multi-armed bandits theory. Based on previous work on bandit theory applied to active learning for regression, we introduce two novel algorithms for solving the online allocation of the budget in a classification problem. Experiments on a generic classification problem demonstrate that these new algorithms compare positively to state-of-the-art methods.

1 Introduction

We place ourselves in the supervised learning framework, especially in a noisy 2-class classification problem. Our work focuses on active learning, which is the process consisting in driving the choice of the examples that need to be labelled in order to minimize the number of queries to the oracle. To do so in an online context the algorithm successively chose the best example to present to the oracle taking into account the information provided from all the previous samples, thus, it can be seen as a sequential decision process. The noisy aspect of this problem —which may come from an intrinsic noise on the data or from an inability of the classifier to distinguish examples— and the fact that the noise is not the same for all examples, implies that some examples can be more or less difficult to classify. Indeed, it is relatively intuitive that little effort must be put into the least noisy examples as they are easy to classify. While it is less intuitive that little effort must also be put into very noisy examples too. Thus, this is an online allocation problem with respect to the noise value, and can be represented by a multi-armed bandit setting, introduced in [14] and surveyed in [3]. A major issue of this problem is that we do not know in advance the noise of an example, but it can be learnt while we present examples to the oracle. We therefore have to make a trade-off between learning the noise of examples and presenting examples according to the noise. Leading to the approach of *Optimism in the face of uncertainty* and algorithms based on *Upper Confidence Bounds* introduced in [1], with the advantage of working under a finite budget.

In the past few years, the field of active learning of noisy (or not) binary classification algorithms has been largely studied, and is surveyed in [12]. In [8], the authors introduced the Uncertainty Sampling Algorithm. It uses probabilistic classifiers, which output a probability of the example having a particular label. The idea is to sample examples for which the classifier is least certain of class membership, *i.e.* the probability is the closest to 0.5. In [7], the authors add to this the quality of estimation of this output. The more the output is close to 0.5, and the less the quality of the estimation is, the highest the probability of giving the wrong label, and so, the more the algorithm tends to sample the example. Other authors make use of a set of classifiers. In [6], [13] and [10], the authors use a version space being the set of classifiers consistent with all labels revealed so far, and a region of uncertainty being the region where there exist a pair of hypotheses that disagrees, at each time step, a sample is taken in the region of uncertainty and all the classifiers inconsistent with the sample are eliminated from the version space. Some work adapt this to noisy classification. In [2], a confidence interval on the performance of the classifiers is established and the classifiers eliminated are those for which it is not possible that they perform best. In [11] and [9], a probability of being the best is affected to each classifier, a sample is taken where the classifiers disagree most, and a Bayesian update is applied to this probability with each sample.

In [5] and [4], the authors study the problem of estimating uniformly well the mean values of several distributions which is equivalent to the problem of regression by a piece-wise constant function and thus, can be seen as a classification problem with an infinite number of classes. This is done under the constraint of using a finite number of samples, referring thereby to active learning. To this end, they model the problem under a multi-armed bandit setting, in which pulling an arm correspond to taking a sample in one of the distributions. The goal is to define an allocation strategy which aims to minimize a loss function. In [5] the loss is the maximum one-arm loss defined by the distance between the mean values and their estimate. Whereas in [4], the loss is the weighted sum of this one-arm loss. To minimize those losses samples need to be allocated in proportion to the variance or the standard deviation depending on the loss. The variance/standard deviation being unknown it has to be estimated at the same time as the allocation of the samples, resulting in a dilemma between using samples to learn the variance or to estimate the mean values. The authors use *Optimism in the face of uncertainty* which is a common approach to solve this dilemma by computing high probability bounds on the value to estimate and sampling the arm with the highest bound.

This paper shows how to use the multi-armed bandit setting for active learning in classification by adapting the algorithms designed in [5] and [4] to the specific case of binary classification. To do so, the two kinds of loss have been redefined using a new one-arm loss, which represents the expected regret of the true risk. Indeed, the optimal risk of a noisy distribution is non zero, thus efforts could be spent for nothing when trying to decrease a risk that cannot be decreased. Therefore, the loss function is the expected difference between the

risk and the optimal risk. Having redefined them, we had too deal with the fact that they aren't inversely proportional to the number of samples any more. An other advantage of classification is that the shape of the distribution of samples is known. Indeed, the samples belong only to $\{0, 1\}$, the distribution is thus Bernoulli. This allows us to derive extremely tight bounds. In the adaptive allocation setting, the parameters of the distributions cannot be accessed, we use the approach of *Optimism in the face of uncertainty*. Allocation strategies have to be defined in the full knowledge setting, in which the parameters of distributions are known in advance, and afterwards build algorithms that sample according to this strategy plus some uncertainty.

In Section II, we define several loss and pseudo-loss functions which have to be minimized. Then, we place ourselves in the full knowledge setting and find the optimal allocation strategies which minimizes those losses. In Section III, we place ourselves in the adaptive allocation setting, and define high probability bounds on the losses. We then present our algorithms which sample arms according to these bounds. In Section IV, we describe a toy problem and show that it is representative of more general problems. Then, we evaluate our algorithms on this toy problem and show that our algorithms perform better than algorithms initially designed for regression. In Section V, we show a conclusion.

2 Allocation strategy in full knowledge

After formalizing our problem under a K-armed bandit setting, we define several kinds of losses to be minimized. We then give the optimal allocation strategy in full knowledge as well as an online criteria to sample according to this strategy.

Let X be an instance space and $Y = \{0, 1\}$ be the set of possible labels. Let an oracle label $x \in X$ with $y \in Y$. Let $N = \{X_1, ..., X_K | \cup_{k=1}^{K} X_k = X, \cap_{k=1}^{K} X_k = \emptyset\}$ be a fixed clustering of X, and $\mathcal{H} = \{f : X \to Y, f(x) = \sum_{i=1}^{K} \mathbb{1}_{\{x \in X_k\}} y_k, y_k \in Y, X_k \in N\}$ be the hypotheses space defined by piecewise constant functions. The goal is to learn the hypothesis which predictions are as close as possible to the oracle's, with as few queries as possible to it.

This problem can be formalized under a K-armed bandit setting where each cluster is an arm k = 1,...,K characterized by a Bernoulli distribution ν_k with mean value μ_k. Indeed, samples taken in a given cluster can only have a value of 0 or 1. At each round, or time step, $t \geq 1$, an allocation strategy selects an arm k_t, which corresponds to picking an example randomly in a cluster and presenting it to the oracle, and receives a sample $y_{k,t} \sim \nu_k$, independently of the past samples. Let $\{w_k\}_{k=1,...,K}$ denote the relative importance of a cluster, summing to 1, coming from the knowledge of how much a cluster will be solicited while using the classifier (not learning it). Most of the time, we will take the distribution of unlabelled data among clusters for $\{w_k\}_{k=1,...,K}$. The goal is to define a strategy that finds the best labels to assign to clusters using a budget of n samples.

Let us write $T_{k,t} = \sum_{s=1}^{t} \mathbb{1}\{k_s = k\}$ the number of times arm k has been pulled up to time t, this way $(T_{k,t})_{k\in\{1,\dots,K\}}$ denotes the allocation strategy. Let $\hat{\mu}_{k,t} = \frac{1}{T_{k,t}} \sum_{s=1}^{T_{k,s}} y_{k,s}$ be the empirical estimate of the mean μ_k at time t.

We now introduce two kinds of losses, as in [5] and [4], each of them is based on a one-arm loss: the maximum one-arm loss among clusters and the sum of the one-arm losses. Let us now see how the one-arm loss is built.

Usually, in a classification setting, we judge on the performance of an algorithm by measuring the risk incurred. Here, the risk is based on the binary loss $L_{0/1}(y, f(x)) = 1$ if $f(x) \neq y$ and 0 otherwise. Thus, the one-arm true risk is $R_k(y) = 1 - \mu_k$ if $y = 1$ and μ_k if $y = 0$, and the one-arm empirical risk is $\hat{R}_{k,n}(y) = 1 - \hat{\mu}_{k,n}$ if $y = 1$ and $\hat{\mu}_{k,n}$ if $y = 0$. In order to minimize the empirical risk, which is the best an algorithm can do knowing only the samples received, the algorithm always assigns the label $[\hat{\mu}_{k,t}]$ to arm k at round t. The risk of the algorithm must always be compared to the best risk that can be reached, otherwise efforts could be spent for nothing when trying to decrease a risk that cannot be decreased. Here, the best risk of arm k can be reached with the label $[\mu_k]$. We therefore define the one arm loss for classification which is the expected regret of the one arm true risk,

$$L_{k,n} = \mathbb{E}[R_k([\hat{\mu}_{k,n}]) - R_k([\mu_k])] = 2|\mu_k - 0.5|\mathbb{P}([\hat{\mu}_{k,n}] \neq [\mu_k]), \tag{1}$$

where the expectation is taken over all the samples.

The two kinds of losses now become

$$L_n^s((T_{k,n})_{k\in\{1,\dots,K\}}) = \sum_{k=1}^{K} w_k L_{k,n} \tag{2}$$

$$\text{and } L_n^m((T_{k,n})_{k\in\{1,\dots,K\}}) = \max_k w_k L_{k,n}. \tag{3}$$

The objective would now be to build algorithms that minimize those losses. However, the method we use to find the best allocation strategy is based on some conditions on the shape of the loss. In order to get a function to minimize with a more convenient shape, we prefer to bound those losses by a pseudo-loss.

We therefore use the knowledge that the mean values of each cluster follow a binomial distribution, allowing us to give a tight bound to the probability $\mathbb{P}([\hat{\mu}_{k,t}] \neq [\mu_k])$ while keeping pseudo-losses for which the one-arm pseudo-loss is strictly decreasing with $T_{k,t}$.

Let $\mathcal{I}_{1-\mu_k}(T_{k,n} - \lfloor T_{k,n}\hat{\mu}_{k,n}\rfloor, \lfloor T_{k,n}\hat{\mu}_{k,n}\rfloor + 1)$ be the cumulative distribution function of $T_{k,n}\hat{\mu}_{k,n}$ following the binomial distribution with parameters $T_{k,n}, \mu_k$. Then,

$$\mathbb{P}([\hat{\mu}_{k,n}] \neq [\mu_k]) = \mathbb{1}_{[\mu_k]=0}\mathbb{P}(\hat{\mu}_{k,n} \geq 0.5) + \mathbb{1}_{[\mu_k]=1}\mathbb{P}(\hat{\mu}_{k,n} < 0.5) \tag{4}$$

$$= \mathbb{1}_{[\mu_k]=0}(1 - \mathcal{I}_{1-\mu_k}(T_{k,n} - \lfloor T_{k,n}/2\rfloor, \lfloor T_{k,n}/2\rfloor + 1)) \tag{5}$$

$$+ \mathbb{1}_{[\mu_k]=1}\mathcal{I}_{1-\mu_k}(T_{k,n} - \lfloor T_{k,n}/2\rfloor, \lfloor T_{k,n}/2\rfloor + 1). \tag{6}$$

Note that the probability given above is a step function of $T_{k,n}/2$ and so is not a strictly decreasing function of $T_{k,n}$. That is not convenient as we require

this condition in the later. That is why we bound this probability by bounding the truncated value $\lfloor T_{k,n}/2 \rfloor$. Then,

$$\mathbb{P}([\hat{\mu}_{k,n}] \neq [\mu_k]) \leq \mathbb{1}_{[\mu_k]=0}(1 - \mathcal{I}_{1-\mu_k}(T_{k,n}/2+1, T_{k,n}/2)) \tag{7}$$
$$+ \mathbb{1}_{[\mu_k]=1}\mathcal{I}_{1-\mu_k}(T_{k,n}/2, T_{k,n}/2+1). \tag{8}$$

We therefore define the two following pseudo-losses:

$$\tilde{L}_n^{s,bin}((T_{k,n})_{k\in\{1,\dots,K\}}) = \sum_{k=1}^{K} l_k(T_{k,n}, \mu_k) \tag{9}$$
$$\text{and } \tilde{L}_n^{m,bin}((T_{k,n})_{k\in\{1,\dots,K\}}) = \max_{k\in\{1,\dots,K\}} l_k(T_{k,n}, \mu_k), \tag{10}$$

$$\text{with } l_k(T_{k,n}, \mu_k) = 2w_k|\mu_k - 0.5|[\mathbb{1}_{[\mu_k]=0}(1 - \mathcal{I}_{1-\mu_k}(T_{k,n}/2+1, T_{k,n}/2)) \tag{11}$$
$$+\mathbb{1}_{[\mu_k]=1}\mathcal{I}_{1-\mu_k}(T_{k,n}/2, T_{k,n}/2+1)] \tag{12}$$

the one-arm pseudo-loss.

One quality of this pseudo-loss is that while it has a convenient shape, it remains very tight. This means that minimizing this pseudo-loss acts almost as good as minimizing the true losses from equations (2) and (3).

Note that l_k as well as $\partial l_k/\partial T_{k,n}$ are both strictly decreasing functions of $T_{k,n}$. Thus, they admit inverses l_k^{-1} and $l_k^{'-1}$ respectively.

Let us remind that the objective is to find allocation strategies that minimize the pseudo-losses. Let $T_{k,n}^{*s,bin}$ and $T_{k,n}^{*m,bin}$ be the optimal number of samples to take in each cluster in order to minimize $\tilde{L}_n^{s,bin}$ and $\tilde{L}_n^{m,bin}$ respectively under the constraint that $\sum_{k=1}^{K} T_{k,n}^{*s,bin} = n$ and $\sum_{k=1}^{K} T_{k,n}^{*m,bin} = n$. Then,

$$T_{k,n}^{*s,bin} = l_k^{'-1}(c^*, \mu_k) \text{ and } T_{k,n}^{*m,bin} = l_k^{-1}(R^*, \mu_k), \tag{13}$$

with c^* and R^* such that $\sum_{k=1}^{K} l_k^{'-1}(c^*, \mu_k) = n$ and $\sum_{k=1}^{K} l_k^{-1}(R^*, \mu_k) = n$.

In the online allocation setting, at each round t a full knowledge algorithm would the arm $k_t^{s,bin}$ or $k_t^{m,bin}$ depending on the pseudo-loss considered. Under the condition that l_k and $\frac{dl_k}{dT_{k,t}}$ are strictly decreasing function of $T_{k,t}$, we have

$$k_t^{s,bin} \in \arg\max_{1\leq k\leq K} \frac{T_{k,t}^{*s,bin}}{T_{k,t}} = \arg\max_k l_k(T_{k,t}, \mu_k) \tag{14}$$
$$\text{and } k_t^{m,bin} \in \arg\max_{1\leq k\leq K} \frac{T_{k,t}^{*m,bin}}{T_{k,t}} = \arg\max_k \frac{\partial l_k}{\partial T_{k,t}}(T_{k,t}, \mu_k). \tag{15}$$

However, the μ_k values are unknown, therefore we cannot build an algorithm that picks $T_{k,n}^*$ samples in each cluster and attain optimality.

We thus use an optimistic approach to estimate the μ_k and at the same time allocate samples as close as possible to the optimum.

3 Allocation strategy using estimated means

In this section, we introduce two algorithms derived from the full knowledge criterion of the previous chapter. The full knowledge setting is not a realistic approach as if the means of the distribution were known in advance, then the optimal labels could be easily deducted from them. Thus, we now refer to the setting of adaptive allocation, in which we have to deal with learning the parameters of the distributions and allocate the samples optimally with respect to these parameters. This problem is usually called the exploration/exploitation dilemma. In order to solve this problem, we use the *Optimism in the face of uncertainty* approach which computes a high probability bound on the value to maximize, and sample the arm with the highest bound. This way, if the value to estimate is close to the upper bound, then the sample is well chosen, otherwise, the bound tighten up, the upper bound decreases, and the arm will not be sampled next time, the uncertainty has reduced. We therefore use this approach in our problem to build adaptive algorithms that allocate samples closest to the optimal.

Input: δ
Initialize: Pull each arm twice
for $t = 2K + 1, \ldots, n$ **do**
  Compute $B_{k,t} = w_k e_k$ with e_k such as $\mathbb{P}(f(T_{k,t}, \mu_k) > e_k | \hat{\mu}_{k,T_{k,t}}) = \delta$
  for each arm $1 \leq k \leq K$ Pull an arm $k_t \in argmax_{1 \leq k \leq K} B_{k,t}$
end
Output: $[\hat{\mu}_{k,n}]$ for all arms $1 \leq k \leq K$

Algorithm 1: Core algorithm

Each algorithm follows the same core, which is described in Algorithm 1, where the difference lies in the criteria f which should be replaced by $l_k(T_{k,t}, \mu_k)$ or $\frac{\partial l_k}{\partial T_{k,t}}(T_{k,t}, \mu_k)$ depending on the pseudo-loss considered. They take one parameter as input: δ which defines the confidence level of the bound. The amount of exploration of the algorithms can be adapted by properly tuning δ.

Usually, the high probability bounds are derived from general concentration inequalities where the shape of the distribution is unknown. Here, we search a confidence interval on the criterion which are functions of the μ_k values. Moreover, we know that the estimated means are drawn from a Bernoulli distribution.

Let us state that Beta distributions provide a family of conjugate prior probability distributions for binomial distributions. The uniform distribution Beta(1,1) is taken as the prior probability distribution, because we have no information about the true distribution. Using the Bayesian inference :

$$\mathbb{P}(\mu_k = x | \hat{\mu}_{k,t}, T_{k,t}) = \frac{x^{T_{k,t}\hat{\mu}_{k,t}}(1-x)^{T_{k,t}(1-\hat{\mu}_{k,t})}}{Beta(T_{k,t}\hat{\mu}_{k,t}, T_{k,t}(1-\hat{\mu}_{k,t})+1)} \tag{16}$$

Let $I_k = \{\mu_k | f(T_{k,t}, \mu_k) > e_k\}$, then

$$\mathbb{P}(f(T_{k,t}, \mu_k) > e_k | \hat{\mu}_{k,t}, T_{k,t}) = \int_{x \in I_k} x^{T_{k,t}\hat{\mu}_{k,t}}(1-x)^{T_{k,t}(1-\hat{\mu}_{k,t})} dx. \tag{17}$$

4 Results

In this section, we evaluate empirically the algorithms introduced in the previous section on a built-in problem. We first demonstrate the performance of those algorithms in full knowledge, to establish the goal standard (the best we can expect from those algorithms). Then, we evaluate the algorithms for adaptive allocation and check if the exploration/exploitation trade-off is well achieved.

Any classification problem which involves a fixed clustering can be modelled by the following parameters: *(i)* the number of clusters K, *(ii)* the mean value of the labels drawn from each clusters $(\mu_k)_{k\in\{1,\dots,K\}}$, *(iii)* the relative importance of each cluster $(w_k)_{k\in\{1,\dots,K\}}$.

Indeed, the relative position of a cluster to an other has no influence on the problem, this is the reason why we could model it under a K-armed bandit problem. The fact that we only care about the label of samples belonging only to $\{0,1\}$ implies that the distribution is Bernoulli and so each cluster is only characterized by its mean value of labels.

To evaluate our algorithms we use the following parameters: *(i)* $K = 16$, being large enough to get some diversity on clusters but not too large as it would be useless, *(ii)* $\forall k, \mu_k = \frac{k}{16}$, this way we represent the largest variety of values possible, and thus our evaluation concerns all the values, *(iii)* the relative importance of each cluster $(w_k)_{k\in\{1,\dots,K\}} = \frac{1}{16}$ thus, our evaluation concerns equally all the values.

At each time step, each algorithm pick a sample in a cluster according to its allocation strategy and select the best hypothesis. Then its prediction is evaluated using the true risk, knowing the true mean values and the weights vector.

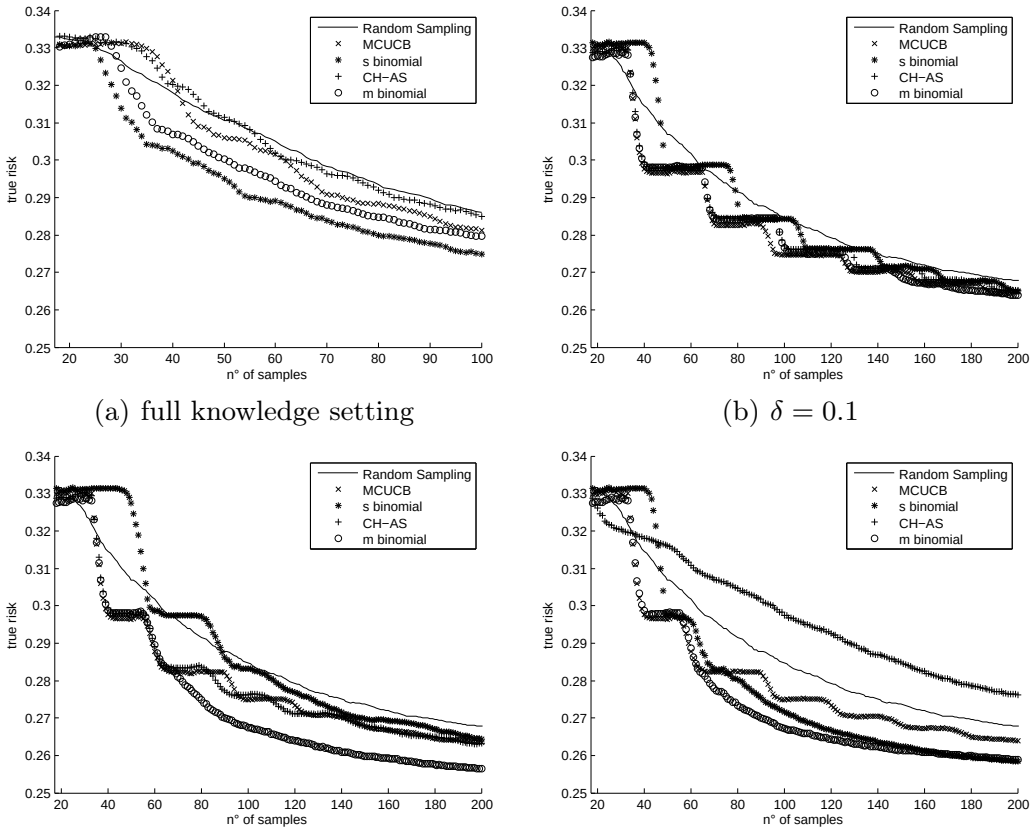
During our evaluations, the algorithms are called as follows:

- Random sampling is the algorithm for which samples are taken uniformly in each cluster, regardless of the mean value of distributions (baseline),
- CH-AS and MCUCB are algorithms introduced in [5] and [4] respectively.
- **m binomial** is the algorithm based on the **maximum one-arm loss** bounded using the knowledge of the **binomial distribution**,
- **s binomial** is the algorithm based on the **sum of the one-arm losses** bounded using the knowledge of the **binomial distribution**,

4.1 Evaluation of the algorithms in full knowledge

First, we evaluate the algorithms in full knowledge. At each time step, the algorithm picks a sample in the arm for which the online allocation criteria from equations (14) and (15) is maximum. The results of the evaluation are shown in Figure 1(a). We can see that the methods introduced by [5] and [4] —respectively MCUCB and CH-AS— do not perform better than random sampling. This is due to the fact that the algorithms are designed for regression and the evaluation is taken from a classification point of view. Those algorithms will allocate more samples to clusters with a mean value far to 0.5 than it has to. Indeed,

the goal of regression is to estimate precisely the mean value, whereas the goal of classification is to be able to predict a good label, so being able to know if a mean value is closer to 0.86 or 0.87 is of no interest for classification because in both cases the predicted label will be 1.



(a) full knowledge setting (b) $\delta = 0.1$

(c) $\delta = 0.9$ (d) $\delta = 0.99999$

Fig. 1. True risk of the algorithms in the full knowledge setting (1(a)) or in the adaptive allocation setting with different values of δ (1(b), 1(c) and 1(d))

4.2 Evaluation of the algorithms under adaptive allocation

Let us now evaluate the results of the algorithms for adaptive allocation. One notable feature of those algorithms is the ability to control the amount of exploration versus exploitation through value of the parameter δ. A low value of δ will result in more exploration and the algorithms will become close to random sampling. On the other side, a high value of δ will result in more exploitation and so, the algorithm will be much more confident about the first estimation of the mean values. The exploration is necessary because, in the pure exploitation setting, when having received 2 samples from each clusters, all the estimations are either 0, 1 or 0.5, sampling a 3rd point in a random cluster will change

the estimation of its mean value to a one with an increased corresponding loss, keeping it the next cluster to sample. This state will not change and this first random cluster would drain all the samples.

We first evaluated our algorithms with a low value for δ resulting in high probability bounds, which was intended. We ran our algorithms with $\delta = 0.1$. We display the results of this run on Figure 1(b).

A strange phenomenon appears in Figure 1(b), we see sort of steps. We remind here that we evaluate the algorithms via 1207 trials and display the average true risk. In fact, these steps are due to the lack of exploitation. Indeed, exploring implies to sample clusters which will not help to decrease the true risk. One could wonder why the arrangement in a way that it highlights two phases, one decreasing phase followed by a constant phase. First let us clear up the fact that this does not reveal an exploitation phase and an exploration phase, everything happens at the same time. But, the important exploration makes the clusters be sampled equally. One step (two phases) correspond to one number of samples in each cluster, i.e. we sample each cluster once then each cluster once again and so on. Now, inside a step the exploitation still plays a role, whereas the clusters are samples equally, the algorithm still starts by the most important clusters. Anyway, the steps are due to the fact that the exploration prevails on the exploitation. If we want to improve the algorithm we have to increase the value of δ.

We now evaluate our algorithms with $\delta = 0.9$. We display the results on Figure 1(c).

We can see that, apart from the first one, the steps have disappeared. The remaining step correspond to the sampling of the third sample in each cluster. This step remains because it is very hard to decide with very few information. Here, the algorithm has already taken 2 samples, so the possible values of the estimated means are 0, 1, and 0.5. Even the cluster requiring most samples will have such a value, thus, the algorithm cannot decide to leave aside one cluster and take a fourth sample while other clusters only have two. This is why this step in the risk is unavoidable.

An other thing we can see is that for this value of δ, the algorithm m binomial have an acceptable performance while s binomial perform poorly. To improve s binomial, we increase again the value of δ.

We now evaluate our algorithms with $\delta = 0.99999$. We display the results on Figure 1(d).

We can see that s binomial improved. One could ask if this value of δ is not exaggerated because its definition tells that it must have a small value, in order to derive high probability bounds. But, here, we see that the algorithms behave well with this value of δ. The only effect of a high value would be that the algorithm exploit to much, and the performance would be affected. Moreover, we can see that this value of δ is not well suited for CH-AS, which see its performance affected.

Finally, we can see that the algorithms built in this paper behave better than those designed for regression in [5] and [4].

5 Conclusion

The paper propose a method to use the *Optimism in the face of uncertainty* approach in an active learning problem for classification. It introduces two algorithms which perform comparatively well and open a new avenue of research. The framework established in this paper is related to the problems encountered in text classification and resembles the problem of parameters estimation in multinomial distributions. Working with a fixed clustering generates new questions that state-of-the art on active learning do not have. In *Uncertainty Sampling* the intent is to sample close to the boundary ($\hat{\mu}(x)$ close to 0.5) because this will redefine it, whereas in our work the clustering remains the same. This leads to finding new loss functions. Our future work concern will be about an adaptive clustering of the space as well as the combination of the information providing from several different clustering .

References

1. P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.
2. M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic Active Learning. In *Proceedings of the International Conference on Machine Learning*, pages 65–72, 2006.
3. S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Machine Learning*, 5(1):1–122, 2012.
4. A. Carpentier, A. Lazaric, M. Ghavamzadeh, R. Munos, and P. Auer. Upper-Confidence-Bound Algorithms for Active Learning in Multi-Armed Bandits. In *Algorithmic Learning Theory*, pages 189–203, 2011.
5. A. Carpentier and R. Munos. Finite Time Analysis of Stratified Sampling for Monte Carlo. In *Advances in Neural Information Processing Systems*, 2011.
6. D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine learning*, 15(2):201–221, 1994.
7. A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Active Learning with Gaussian Processes for Object Categorization. In *International Conference on Computer Vision*, pages 1–8, 2007.
8. D. Lewis and W. Gale. A Sequential Algorithm for Training Text Classifiers. In *SIGIR Conference*, pages 3–12, 1994.
9. M. Naghshvar, T. Javidi, and K. Chaudhuri. Noisy Bayesian Active Learning. In *Allerton Conference on Communication, Control, and Computing*, pages 1626–1633, 2012.
10. R. Nowak. Generalized Binary Search. In *Allerton Conference on Communication, Control, and Computing*, pages 568–574, 2008.
11. R. Nowak. Noisy Generalized Binary Search. In *Advances in Neural Information Processing Systems*, pages 1366–1374, 2009.
12. B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.
13. H.S. Seung, M. Opper, and H. Sompolinsky. Query by Committee. In *Proceedings of Computational Learning Theory*, pages 287–294, 1992.
14. W. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, pages 285–294, 1933.

Recommender-based Multiple Classifier System

Yury Kashnitsky

National Research University Higher School of Economics
Scientific-Educational Laboratory for Intelligent Systems and Structural Analysis
Moscow, Russia
ykashnitsky@hse.ru

Abstract. The paper briefly introduces multiple classifier systems and describes a new algorithm, which improves classification accuracy by means of recommendation of a proper algorithm to an object classification. This recommendation is done assuming that a classifier is likely to predict the label of the object correctly if it has correctly classified its neighbors. The process of assigning a classifier to each object is based on Formal Concept Analysis. We explain the idea of the algorithm with a toy example and describe our first experiments with real-world datasets.

1 Introduction

The topic of Multiple Classifier Systems (MCSs) is well studied in machine learning community [1]. Such algorithms appear with different names – mixture of experts, committee machines, classifier ensembles, classifier fusion and others.

The underlying idea of all these systems is to train several (base) classifiers on a training set and to combine their predictions in order to classify objects from a test set [1]. This idea probably dates back to as early as the 18^{th} century. The Condorcet's jury theorem, that was formulated in 1785 in [2], claims that if a population makes a group decision and each voter most likely votes correctly, then adding more voters increases the probability that the majority decision is correct. The probability that the majority votes correctly tends to 1 as the number of voters increases. Similarly, if we have multiple weak classifiers (meaning that classifier's error on its training data is less than 50% but greater than 0%), we can combine their predictions and boost the classification accuracy as compared to those of each single base classifier.

Among the most popular MCSs are bagging [3], boosting [7], random forests [9], and stacked generalization (or stacking) [10].

In this paper, we present one more algorithm of such type – Recommender-based Multiple Classifier System (RMCS). Here the underlying proposition is that a classifier is likely to predict the label of the object from a test set correctly if it has correctly classified its neighbors from a training set.

The paper is organized as follows. In chapter 2, we discuss bagging, boosting and stacking. In Section 3, we introduce basic definitions of Formal Concept Analysis (FCA). Section 4 provides an example of execution of the proposed RMCS algorithm on a toy synthetic dataset. Then, Section 5 describes the RMCS

algorithm itself. Further, the results of the experiments with real data are presented. Section 7 concludes the paper.

2 Multiple Classifier Systems

In this chapter, we consider several well-known multiple classier systems.

2.1 Bagging

The *bootstrap sampling* technique has been used in statistics for many years. *Bootstrap aggregating*, or *bagging*, is one of the applications of bootstrap sampling in machine learning. As sufficiently large data sets are often expensive or impossible to obtain, with bootstrap sampling, multiple random samples are created from the source data by sampling with replacement. Samples may overlap or contain duplicate items, yet the combined results are usually more accurate than a single sampling of the entire source data achieves.

In machine learning the bootstrap samples are often used to train classifiers. Each of these classifiers can classify new instances making a prediction; then predictions are combined to obtain a final classification.

The aggregation step of bagging is only helpful if the classifiers are different. This only happens if small changes in the training data can result in large changes in the resulting classifier – that is, if the learning method is unstable [3].

2.2 Boosting

The idea of *boosting* is to iteratively train classifiers with a weak learner (the one with error better than 50% but worse than 0%) [4]. After each classifier is trained, its accuracy is measured, and misclassified instances are emphasized. Then the algorithm trains a new classifier on the modified dataset. At classification time, the boosting classifier combines the results from the individual classifiers it trained.

Boosting was originally proposed by Schapire and Freund [5,6]. In their *Adaptive Boosting*, or *AdaBoost*, algorithm, each of the training instances starts with a weight that tells the base classifier its relative importance [7]. At the initial step the weights of n instances are evenly distributed as $\frac{1}{n}$ The individual classifier training algorithm should take into account these weights, resulting in different classifiers after each round of reweighting and reclassification. Each classifier also receives a weight based on its accuracy; its output at classification time is multiplied by this weight.

Freund and Schapire proved that, if the base classifier used by AdaBoost has an error rate of just slightly less than 50%, the training error of the meta-classifier will approach zero exponentially fast [7]. For a two-class problem the base classifier only needs to be slightly better than chance to achieve this error rate. For problems with more than two classes less than 50% error is harder to achieve. Boosting appears to be vulnerable to overfitting. However, in tests it rarely overfits excessively [8].

2.3 Stacked generalization

In *stacked generalization*, or *stacking*, each individual classifier is called a *level-0 model*. Each may vote, or may have its output sent to a *level-1 model* – another classifier that tries to learn which level-0 models are most reliable. Level-1 models are usually more accurate than simple voting, provided they are given the class probability distributions from the level-0 models and not just the single predicted class [10].

3 Introduction to Formal Concept Analysis

3.1 Main definitions

A *formal context* in FCA is a triple $K = (G, M, I)$, where G is a set of objects, M is a set of attributes, and the binary relation $I \subseteq G \times M$ shows which object possesses which attribute. gIm denotes that object g has attribute m. For subsets of objects and attributes $A \subseteq G$ and $B \subseteq M$ *Galois operators* are defined as follows:

$$A' = \{m \in M \mid gIm \ \forall g \in A\},$$
$$B' = \{g \in G \mid gIm \ \forall m \in B\}.$$

A pair (A, B) such that $A \subseteq G, B \subseteq M, A' = B$ and $B' = A$, is called a *formal concept* of a context K. The sets A and B are closed and called the *extent* and the *intent* of a formal concept (A, B) respectively. For the set of objects A the set of their common attributes A' describes the similarity of objects of the set A and the closed set A'' is a cluster of similar objects (with the set of common attributes A') [11].

The number of formal concepts of a context $K = (G, M, I)$ can be quite large ($2^{min\{|G|,|M|\}}$ in the worst case), and the problem of computing this number is #P-complete [12]. There exist some ways to reduce the number of formal concepts, for instance, choosing concepts by stability, index or extent size [13].

For a context (G, M, I), a concept $X = (A, B)$ is *less general than or equal to* a concept $Y = (C, D)$ (or $X \leq Y$) if $A \subseteq C$ or, equivalently, $D \subseteq B$. For two concepts X and Y such that $X \leq Y$ and there is no concept Z with $Z \neq X, Z \neq Y, X \leq Z \leq Y$, the concept X is called a *lower neighbor* of Y, and Y is called an *upper neighbor* of X. This relationship is denoted by $X \prec Y$. Formal concepts, ordered by this relationship, form a *complete concept lattice* which might be represented by a *Hasse diagram* [14]. Several algorithms for building formal concepts (including *Close by One*) and constructing concept lattices are studied also in [14].

One can address to [11] and [15] to find some examples of formal contexts, concepts and lattices with their applications. Chapter 4 also shows the usage of FCA apparatus in a concrete task.

However, in some applications there is no need to find all formal concepts of a formal context or to build the whole concept lattice. Concept lattices, restricted

to include only concepts with frequent intents, are called *iceberg lattices*. They were shown to serve as a condensed representation of association rules and frequent itemsets in data mining [15].

Here we modified the *Close by One* algorithm slightly in order to obtain only the upper-most concept of a formal context and its lower neighbors. The description of the algorithm and details of its modification is beyond the scope of this paper.

4 A toy example

Let us demonstrate the way RMCS works with a toy synthetic dataset shown in Table 1. We consider a binary classification problem with 8 objects comprising a training set and 2 objects in a test set. Each object has 4 binary attributes and a target attribute (class). Suppose we train 4 classifiers on this data and try to predict labels for objects 9 and 10.

Using FCA terms, we denote by $G = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ — the whole set of objects, $G_{test} = \{9, 10\}$ — the test set, $G_{train} = G \backslash G_{test}$ — the training set, $M = \{m_1, m_2, m_3, m_4\}$ — the attribute set, $C = \{cl_1, cl_2, cl_3, cl_4\}$ — the set of classifiers.

Table 1. A sample data set of 10 objects with 4 attributes and 1 binary target class

G/M	m_1	m_2	m_3	m_4	Label
1	×	×		×	1
2	×			×	1
3		×	×		0
4	×		×	×	1
5	×	×	×		1
6		×	×	×	0
7	×	×	×		1
8			×	×	0
9	×	×	×	×	?
10		×		×	?

Table 2. A classification context

G/C	cl_1	cl_2	cl_3	cl_4
1	×		×	×
2		×	×	
3	×			×
4		×	×	
5	×	×		
6	×	×		×
7		×		×
8		×	×	×

Here we run leave-one-out cross-validation on this training set for 4 classifiers. Further, we fill in Table 2, where a cross for object i and classifier cl_j means that cl_j correctly classifies object i in the process of cross-validation. To clarify, a cross for object 3 and classifier cl_4 means that after being trained on the whole training set but object 3 (i.e. on objects $\{1, 2, 4, 5, 6, 7, 8\}$), classifier cl_4 correctly predicted the label of object 3.

Let us consider Table 2 as a formal context with objects G and attributes C (so now classifiers play the role of attributes). We refer to it as classification context. The concept lattice for this context is presented in Fig. 1.

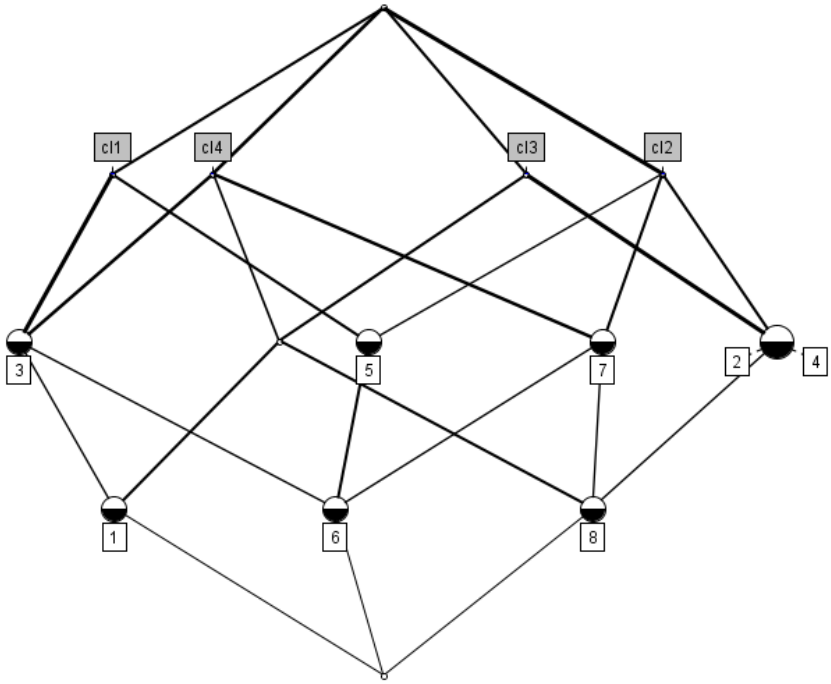


Fig. 1. The concept lattice of the classification context

As it was mentioned, the number of formal concepts of a context $K = (G, M, I)$ can be exponential in the worst case. But for the toy example it is possible to draw the whole lattice diagram. Thankfully, we do not need to build the whole lattice in RMCS algorithm — we only keep track of its top concepts.

Here are these top concepts: $(G, \emptyset)$, $(\{1, 3, 5, 6\}, \{cl_1\})$, $(\{2, 4, 5, 6, 7, 8\}, \{cl_2\})$, $(\{1, 2, 4, 8\}, \{cl_3\})$, $(\{1, 3, 6, 7, 8\}, \{cl_4\})$.

To classify objects from G_{test}, we first find their k nearest neighbors from G_{train} according to some distance metric. In this case, we use $k = 3$ and Hamming distance. In these conditions, we find that three nearest neighbors of object 9 are 4, 5 and 7, while those of object 10 are 1, 6 and 8.

Then, we take these sets of nearest neighbors $Neighb_9 = \{4, 5, 7\}$ and $Nieghb_{10} = \{1, 6, 8\}$, and find maximal intersections of these sets with the extents of formal concepts presented above (ignoring the concept $(G, \emptyset)$). The intents (i.e. classifiers) of the corresponding concepts are given as recommendations for the objects from G_{test}. The procedure is summarized in Table 3.

Finally, the RMCS algorithm predicts the same labels for objects 9 and 10 as classifiers cl_2 and cl_4 do correspondingly.

Lastly, let us make the following remarks:

Table 3. Recommending classifiers for objects from G_{test}

G_{test}	1^{st} nearest neighbor	2^{nd} nearest neighbor	3^{rd} nearest neighbor	Neighbors	Classification concept which extent gives the maximal intersection with the Neighbors set	Recommended classifier
9	4	5	7	$\{4, 5, 7\}$	$(\{2, 4, 5, 6, 7, 8\}, \{cl_2\})$	cl_2
10	1	6	8	$\{1, 6, 8\}$	$(\{1, 3, 6, 7, 8\}, \{cl_4\})$	cl_4

1. We would not have ignored the upper-most concept with extent G if it did not have an empty intent. That is, if we had the top concept of the classification context in a form $(G, \{cl_j\})$ it would mean that cl_j correctly classified all objects from the training set and we would therefore recommend it to the objects from the test set.
2. One more situation might occur that two or more classifiers turn out to be equally good at classifying objects from G_{train}. That would mean that the corresponding columns in classification table are identical and, therefore, the intent of some classification concept is comprised of more than one classifier. In such case, we do not have any argument for preferring one classifier to another and, hence, the final label would be defined as a result of voting procedure among the predicted labels of these classifiers.
3. Here we considered an input dataset with binary attributes and a binary target class. However, the idea of the RMCS algorithm is still applicable for datasets with numeric attributes and multi-class classification problems.

5 Recommender-based Multiple Classifier System

In this section, we discuss the Recommender-based Multiple Classifier System (RMCS). The pseudocode of the RMCS algorithm is presented in the listing Algorithm 1.

The inputs for the algorithm are the following:

1. $\{X_{train}, y_{train}\}$ — is a training set, X_{test} — is a test set;
2. $C = \{cl_1, cl_2, ..., cl_K\}$ — is a set of K base classifiers. The algorithm is intended to perform a classification accuracy exceeding those of base classifiers;
3. $dist(x_1, x_2)$ — is a distance function for objects which is defined in the attribute space. This might be the Minkowski (including Hamming and Euclidean) distance, the distance weighted by attribute importance and others.
4. k, n_fold — are parameters. Their meaning is explained below;
5. $topCbO(context)$ — is a function for building the upper-most concept of a formal context and its lower neighbors. Actually, it is not an input for the algorithm but RMCS uses it.

The algorithm includes the following steps:

1. Cross-validation on the training set. All K classifiers are trained on $n\_folds-1$ folds of X_{train}. Then a classification table (or context) is formed where a cross is put for object i and classifier cl_j if cl_j correctly classifies object i after training on $n\_folds-1$ folds (where object i belongs to the rest fold);
2. Running base classifiers. All K classifiers are trained on the whole X_{train}. Then, a table of predictions is formed where (i, j) position keeps the predicted label for object i from X_{test} by classifier cl_j;
3. Building top formal concepts of the classification context. The *topCbO* algorithm is run in order to build upper formal concepts of a classification context. These concepts have the largest possible number of objects in extents and minimal possible number of classifiers in their intents (not counting the upper-most concept);
4. Finding neighbors of the objects from X_{test}. The objects from the test set are processed one by one. For every object from X_{test} we find its k nearest neighbors from X_{train} according to the selected metric $sim(x_1, x_2)$. Let us say these k objects form a set $Neighbors$. Then, we search for a concept of a classification context which extent yields maximal intersection with the set $Neighbors$. If the intent of the upper-most concept is an empty set (i.e., no classifier correctly predicted the labels of all objects from X_{train}, which is mostly the case), then the upper-most concept $(G, \emptyset)$ is ignored. Thus, we select a classification concept, and its intent is a set of classifiers C_{sel};
5. Classification. If C_{sel} consists of just one classifier, we predict the same label for the current object from X_{test} as this classifier does. If there are several selected classifiers, then the predicted label is defined by majority rule.

6 Experiments

The algorithm, described above, was implemented in Python 2.7.3 and tested on a 2-processor machine (Core i3-370M, 2.4 HGz) with 3.87 GB RAM.

We used four UCI datasets in these experiments - `mushrooms`, `ionosphere`, `digits`, and `nursery`.[1] Each of the datasets was divided into training and test sets in proportion 70:30.

We ran 3 classifiers implemented in `SCIKIT-LEARN` library [2](written in Python) which served as base classifiers for the RMCS algorithm as well. These were a Support Vector Machine with Gaussian kernel (`svm.SVC()` in `Scikit`), logistic regression (`sklearn.linear_model.LogisticRegression()`) and k Nearest Neighbors classifier (`sklearn.neighbors.classification.KNeighborsClassifier()`).

The classification accuracy of each classifier on each dataset is presented in Table 4 along with special settings of parameters. Moreover, for comparison, the results for `Scikit`'s implementation of bagging with SVM as a base classifier and AdaBoost on decision stumps [3] are presented.

[1] `http://archive.ics.uci.edu/ml/datasets`

[2] http://scikit-learn.org

[3] https://github.com/pbharrin/machinelearninginaction/tree/master/Ch07

Algorithm 1 Recommender-based Multiple Classifier System

Input: $\{X_{train}, y_{train}\}, X_{test}$ — are training and test sets, $C = \{cl_1, cl_2, ..., cl_K\}$ — is a set of base classifiers, $topCbO(context, n)$ — is a function for building the uppermost concept of a formal context and its lower neighbors, $dist(x_1, x_2)$ — is a distance function defined in the attribute space, k — is a parameter (the number of neighbors), n_fold — is the number of folds for cross-validation on a training set

Output: y_{test} — are predicted labels for objects from X_{test}

$train\_class\_context = [\,][\,]$ — is a 2-D array
$test\_class\_context = [\,][\,]$ — is a 2-D array
for $i \in 0 \ldots len(X_{train}) - 1$ **do**
  for $cl \in 0 \ldots len(C) - 1$ **do**
    train classifier cl on $(n\_fold - 1)$ folds not including object $X_{train}[i]$
    $pred =$ predicted label for $X_{train}[i]$ by classifier cl
    $train\_class\_context[i][cl] = (pred == y_{train}[i])$
  end for
end for
for $cl \in 0 \ldots len(C) - 1$ **do**
  train classifier cl on the whole X_{train}
  $pred =$ predicted labels for X_{test} by classifier cl
  $test\_class\_context[:][cl] = pred$
end for
$top\_concepts = topCbO(class\_context)$
for $i \in 0 \ldots len(X_{test}) - 1$ **do**
  $Neighbors = k$ nearest neighbors of $X_{test}[i]$ from X_{train} according to $sim(x_1, x_2)$
  $concept = argmax(c.extent \cap Neighbors), c \in top\_concepts$
  $C_{sel} = concept.intent$
  $labels =$ predictions for $X_{test}[i]$ made by classifiers from C_{sel}
  $y_{test}[i] = argmax(count\_freq(labels))$
end for

Table 4. Classification accuracy of 6 algorithms on 4 UCI datasets: mushrooms (1), ionosphere (2), digits (3), and nursery (4)

Data	SVM, RBF kernel (C=1, γ=0.02)	Logit (C=10)	kNN (euclidean, k=3)	RMCS (k=3, n_folds=4)	Bagging SVM (C=1, γ=0.02) 50 estimators	AdaBoost on decision stumps, 50 iterations
1	0.998 t=0.24 sec.	0.996 t=0.17 sec.	0.989 t=1.2*10^{-2} sec.	0.997 t=29.45 sec.	0.998 t=3.35 sec.	0.998 t=44.86 sec.
2	0.906 t=5.7*10^{-3} sec.	0.868 t=10^{-2} sec.	0.858 t=8*10^{-4} sec.	0.933 t=3.63 sec.	0.896 t=0.24 sec.	0.934 t=22.78 sec.
3	0.917 t=0.25 sec.	0.87 t=0.6 sec.	0.857 t=1.1*10^{-2} sec.	0.947 t=34.7 sec.	0.92 t=4.12 sec.	0.889 t=120.34 sec.
4	0.914 t=3.23 sec.	0.766 t=0.3 sec.	0.893 t=3.1*10^{-2} sec.	0.927 t=220.6 sec.	0.913 t=38.52 sec.	0.903 t=1140 sec.

Data	SVM, RBF kernel ($C=10^3$, $\gamma=0.02$)	Logit ($C=10^3$)	kNN (minkowski, p=1, k=5)	RMCS (k=5, n_folds=10)	Bagging SVM ($C=10^3$, $\gamma=0.02$) 50 estimators	AdaBoost on decision stumps, 100 iterations
1	0.998 t=0.16 sec.	0.999 t=0.17 sec.	0.999 t=$1.2*10^{-2}$sec.	0.999 t=29.45 sec.	0.999 t=3.54 sec.	0.998 t=49.56 sec.
2	0.906 t=$4.3*10^{-3}$ sec.	0.868 t=10^{-2} sec.	0.887 t=$8*10^{-4}$ sec.	0.9 t=3.63 sec.	0.925 t=0.23 sec.	0.934 t=31.97 sec.
3	0.937 t=0.22 sec.	0.87 t=0.6 sec.	0.847 t=$1.1*10^{-2}$ sec.	0.951 t=34.7 sec.	0.927 t=4.67 sec.	0.921 t=131.6 sec.
4	0.969 t=2.4 sec.	0.794 t=0.3 sec.	0.945 t=$3*10^{-2}$ sec.	0.973 t=580.2 sec.	0.92 t=85.17 sec.	0.912 t=2484 sec.

As we can see, RMCS outperformed its base classifiers in all cases, while it turned out to be better than bagging only in case of multi-class classification problems (datasets `digits` and `nursery`).

7 Conclusion

In this paper, we described the underlying idea of multiple classifier systems, discussed bagging, boosting and stacking. Then, we proposed a multiple classifier system which turned out to outperform its base classifiers and two particular implementations of bagging and AdaBoost in two multi-class classification problems.

Our further work on the algorithm will continue in the following directions: exploring the impact of different distance metrics (such as the one based on attribute importance or information gain) on the algorithm's performance, experimenting with various types of base classifiers, investigating the conditions preferable for RMCS (in particular, when it outperforms bagging and boosting), improving execution time of the algorithm and analyzing RMCS's overfitting.

Acknowledgements. The author would like to thank his colleagues from Higher School of Economics, Dmitry Ignatov and Sergei Kuznetsov, for their well-timed advice and support during this work. He is also grateful to Jaume Baixeries and Konstantin Vorontsov for their inspirational discussions which directly or implicitly influenced this study.

References

1. Izenman, A. J.: Committee Machines. Modern Multivariate Statistical Techniques. pp. 505—550. Springer New York (2008).
2. Condorcet, M.-J.-A.-N.: Essay on the Application of Analysis to the Probability of Majority Decisions. (1785)
3. Breiman, L.: Bagging predictors. Machine Learning. 24(2), 123–140. (1996)
4. Schapire, R. E.: The Strength of Weak Learnability. Machine Learning. 5, 197—227 (1990)
5. Freund, Y: Boosting a Weak Learning Algorithm by Majority. Information and Computation. 121(2), 256—285 (1995)
6. Freund, Y., Schapire, R. E.: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. Journal of Computer and System Sciences. 55, 119—139 (1997)
7. Freund, Y., Schapire, R. E.: A Short Introduction to Boosting. (1999).
8. Dietterich, T. G.: Ensemble Methods in Machine Learning. Multiple Classifier Systems, LBCS-1857. pp. 1—15. Springer (2000).
9. Breiman, L.: Random Forests. Machine Learning. 45(1), 5—32 (2001)
10. Wolpert, D. H.: Stacked Generalization. Neural Networks, 5, 241—259 (1992)
11. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer-Verlag New York, Inc., Secaucus, NJ, USA (1997).
12. Kuznetsov, S. O.: On Computing the Size of a Lattice and Related Decision Problems. Order. 18(4), 313—321 (2001).
13. Kuznetsov, S. O.: On stability of a formal concept. Annals of Mathematics and Artificial Intelligence. 49(1-4), 101—115 (2007)
14. Kuznetsov, S. O., Obiedkov, S.: Comparing Performance of Algorithms for Generating Concept Lattices. Journal of Experimental and Theoretical Artificial Intelligence. 14, 189—216 (2002).
15. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L.: Intelligent Structuring and Reducing of Association Rules with Formal Concept Analysis. In: Baader, F., Brewka, G., and Eiter, T. (eds.) KI 2001: Advances in Artificial Intelligence. pp. 335—350. Springer Berlin Heidelberg (2001)

Clustering Boolean Tensors

Saskia Metzler and Pauli Miettinen

Max Planck Institut für Informatik
Saarbrücken, Germany
{saskia.metzler, pauli.miettinen}@mpi-inf.mpg.de

Abstract. Tensor factorizations are computationally hard problems, and in particular, often are significantly harder than their matrix counterparts. In case of Boolean tensor factorizations – where the input tensor and all the factors are required to be binary and we use Boolean algebra – much of that hardness comes from the possibility of overlapping components. Yet, in many applications we are perfectly happy to partition at least one of the modes. In this paper we investigate what consequences does this partitioning have on the computational complexity of the Boolean tensor factorizations and present a new algorithm for the resulting clustering problem. While future work aims at further tuning our algorithm for Boolean tensor clustering, it already now can obtain better results than algorithms solving different relaxations of the problem.

1 Introduction

Tensors become increasingly popular data representations in data mining. Ternary (or higher order) relations, for instance, can be represented as binary 3-way (or multi-way) tensors. Given such data, the question is whether there is any underlying structure or regularity in the data. To approach that question, typically tensor decomposition methods are applied. In this work, we restrict ourselves to binary data and also restrict the factors in the decomposition to be binary.

Tensor decompositions with similar restrictions have previously been studied: Cerf et al. [3] present an algorithm for the extraction of noise-tolerant itemsets in binary relations. Erdős and Miettinen [6] propose a scalable algorithm for Boolean CANDECOMP/PARAFAC (CP) and Tucker decompositions, and apply it to information extraction [5].

The novelty of the Boolean tensor decomposition approach we present is the restriction to non-overlapping factors in one mode. This takes apart complexity from the task and also often fits the structure of real-world data. For example in subject–relation–object data, the relations are non-overlapping: While a subject can be linked to multiple objects and vice versa, the relation is a property of the link between them. The algorithm we present has better approximability results, is simpler than previous algorithms, and also outperforms them.

Existing work relates to different aspects of our approach: Jegelka et al. [9] study the problem of clustering simultaneously all modes of a tensor (tensor co-clustering). In the context of formal concept analysis, Belohlavek et al. [2] use

triadic concepts to obtain an optimal decomposition of three-way binary data. That approach is extended to approximate solutions by Ignatov et al. [8]. Huang et al. [7] and Liu et al. [12] (among others) study the problem where only one mode is clustered and the remaining modes are represented using a low-rank approximation. The latter form is closer to what we study in this paper, but the techniques used in the continuous methods do not apply to the binary case.

2 Preliminaries

Throughout this paper, we indicate vectors as bold lower-case letters ($\boldsymbol{v}$), matrices as bold upper-case letters ($\boldsymbol{M}$), and tensors as bold upper-case calligraphic letters ($\boldsymbol{\mathcal{T}}$). Element (i, j, k) of a 3-way tensor $\boldsymbol{\mathcal{X}}$ is denoted as x_{ijk}. A colon in a subscript denotes taking that mode entirely; for example, $\boldsymbol{X}_{::k}$ is the kth *frontal slice* of $\boldsymbol{\mathcal{X}}$ (shorthand $\boldsymbol{X}_k$).

A tensor can be *unfolded* into a matrix by arranging its fibers (i.e. its columns, rows, or tubes in case of a 3-way tensor) as columns of a matrix. For a *mode-n matricization*, mode-n fibers are used as the columns and the result is $\boldsymbol{X}_{(n)}$.

The outer product of vectors is denoted by $\boxtimes$. For vectors $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ of length n, m, and l, $\boldsymbol{\mathcal{X}} = \boldsymbol{a} \boxtimes \boldsymbol{b} \boxtimes \boldsymbol{c}$ is an n-by-m-by-l tensor with $x_{ijk} = a_i b_j c_k$.

The *Boolean tensor sum* of binary tensors $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ is defined as $(\boldsymbol{\mathcal{X}} \vee \boldsymbol{\mathcal{Y}})_{ijk} = x_{ijk} \vee y_{ijk}$. For binary matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ where $\boldsymbol{X}$ has r columns and $\boldsymbol{Y}$ has r rows their *Boolean matrix product*, $\boldsymbol{X} \circ \boldsymbol{Y}$, is defined as $(\boldsymbol{X} \circ \boldsymbol{Y})_{ij} = \bigvee_{k=1}^{r} x_{ik} y_{kj}$. The *Boolean matrix rank* of a binary matrix $\boldsymbol{A}$ is the least r such that there exists a pair of binary matrices $(\boldsymbol{X}, \boldsymbol{Y})$ of inner dimension r with $\boldsymbol{A} = \boldsymbol{X} \circ \boldsymbol{Y}$.

Definition 1 (Boolean tensor rank). *The* Boolean rank *of a 3-way binary tensor $\boldsymbol{\mathcal{X}}$,* $\operatorname{rank}_B(\boldsymbol{\mathcal{X}})$*, is the least integer r such that there exist r triplets of binary vectors ($\boldsymbol{a}_i$, $\boldsymbol{b}_i$, $\boldsymbol{c}_i$) with $\boldsymbol{\mathcal{X}} = \bigvee_{i=1}^{r} \boldsymbol{a}_i \boxtimes \boldsymbol{b}_i \boxtimes \boldsymbol{c}_i$.*

A binary matrix $\boldsymbol{X}$ is a *cluster assignment matrix* if each row of $\boldsymbol{X}$ has exactly one non-zero element. In that case the Boolean matrix product corresponds to the regular matrix product, $\boldsymbol{X} \circ \boldsymbol{Y} = \boldsymbol{X}\boldsymbol{Y}$.

For a tensor $\boldsymbol{\mathcal{X}}$, $|\boldsymbol{\mathcal{X}}|$ denotes its number of non-zero elements. The Frobenius norm of a 3-way tensor $\boldsymbol{\mathcal{X}}$ is $\|\boldsymbol{\mathcal{X}}\| = \sqrt{\sum_{i,j,k} x_{ijk}^2}$. If $\boldsymbol{\mathcal{X}}$ is binary, $|\boldsymbol{\mathcal{X}}| = \|\boldsymbol{\mathcal{X}}\|^2$. The *similarity* between two n-by-m-by-l binary tensors $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ is defined as $\mathsf{sim}(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}) = nml - |\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{Y}}|$.

Let $\boldsymbol{X}$ be n_1-by-m_1 and $\boldsymbol{Y}$ be n_2-by-m_2 matrix. Their *Kronecker (matrix) product*, $\boldsymbol{X} \otimes \boldsymbol{Y}$, is the $n_1 n_2$-by-$m_1 m_2$ matrix defined by

$$\boldsymbol{X} \otimes \boldsymbol{Y} = \begin{pmatrix} x_{11}\boldsymbol{Y} & x_{12}\boldsymbol{Y} & \cdots & x_{1m_1}\boldsymbol{Y} \\ x_{21}\boldsymbol{Y} & x_{22}\boldsymbol{Y} & \cdots & x_{2m_1}\boldsymbol{Y} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n_1 1}\boldsymbol{Y} & x_{n_1 2}\boldsymbol{Y} & \cdots & x_{n_1 m_1}\boldsymbol{Y} \end{pmatrix}.$$

The *Khatri–Rao (matrix) product* of $\boldsymbol{X}$ and $\boldsymbol{Y}$ is defined as 'column-wise Kronecker'. That is, $\boldsymbol{X}$ and $\boldsymbol{Y}$ must have same number of columns ($m_1 = m_2 =$

m), and their Khatri–Rao product $\boldsymbol{X} \odot \boldsymbol{Y}$ is the $n_1 n_2$-by-m matrix defined as $\boldsymbol{X} \odot \boldsymbol{Y} = \big(\boldsymbol{x}_1 \otimes \boldsymbol{y}_1, \boldsymbol{x}_2 \otimes \boldsymbol{y}_2, \ldots, \boldsymbol{x}_m \otimes \boldsymbol{y}_m\big)$. Notice that if $\boldsymbol{X}$ and $\boldsymbol{Y}$ are binary, so are $\boldsymbol{X} \otimes \boldsymbol{Y}$ and $\boldsymbol{X} \odot \boldsymbol{Y}$.

The Boolean tensor CP decomposition mirrors the standard tensor CP decomposition.

Definition 2 (Boolean CP). *Given an n-by-m-by-l binary tensor $\boldsymbol{\mathcal{X}}$ and an integer r, find binary matrices $\boldsymbol{A}$ (n-by-r), $\boldsymbol{B}$ (m-by-r), and $\boldsymbol{C}$ (l-by-r) such that they minimize $|\boldsymbol{\mathcal{X}} - \bigvee_{i=1}^{r} \boldsymbol{a}_i \boxtimes \boldsymbol{b}_i \boxtimes \boldsymbol{c}_i|$.*

Following Kolda and Bader [11], we use $[[\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]]$ to denote the normal 3-way CP and $[[\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]]_B$ for the Boolean CP. We can also write the Boolean CP as matrices using unfolding. The matrix product has to be the Boolean matrix product while the Khatri–Rao product is closed under the Boolean algebra:

$$\boldsymbol{X}_{(1)} = \boldsymbol{A} \circ (\boldsymbol{C} \odot \boldsymbol{B})^T\,,\ \boldsymbol{X}_{(2)} = \boldsymbol{B} \circ (\boldsymbol{C} \odot \boldsymbol{A})^T\,,\ \boldsymbol{X}_{(3)} = \boldsymbol{C} \circ (\boldsymbol{B} \odot \boldsymbol{A})^T\,. \quad (1)$$

Both problems, finding the least error Boolean CP decomposition and deciding the Boolean tensor rank, are NP-hard [13].

3 Problem Definition

We consider the variation of *tensor clustering* where the idea is to cluster one mode of a tensor and potentially reduce the dimensionality of the other modes.

Assuming a 3-way tensor and that we do the clustering in the last mode, we can express the *Boolean CP clustering* (BCPC) problem as follows:

Definition 3 (BCPC). *Given a binary n-by-m-by-l tensor $\boldsymbol{\mathcal{X}}$ and an integer k, find matrices $\boldsymbol{A} \in \{0,1\}^{n \times k}$, $\boldsymbol{B} \in \{0,1\}^{m \times k}$, and $\boldsymbol{C} \in \{0,1\}^{l \times k}$ such that $\boldsymbol{C}$ is a cluster assignment matrix and that the tuple $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ maximizes $\mathsf{sim}(\boldsymbol{\mathcal{X}}, [[\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]]_B)$*

To understand what BCPC does, we use the unfolding rules (1) and write $\boldsymbol{X}_{(3)} \approx \boldsymbol{C}(\boldsymbol{B} \odot \boldsymbol{A})^T$, where we can see that we have restricted the type of cluster centroids: While in a general clustering problem, we would aim to cluster the frontal slices of $\boldsymbol{\mathcal{X}}$ into k clusters each represented by an n-by-m matrix, in this setting each cluster representative has to be of type $(\boldsymbol{b} \otimes \boldsymbol{a})^T$. This restriction on the cluster centroids plays a crucial role in the decomposition, as we shall see shortly.

4 Solving Maximum-Similarity BCPC

Given a tensor $\boldsymbol{\mathcal{X}}$, for the optimal solution to BCPC, we need matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ that maximize $\mathsf{sim}(\boldsymbol{X}_{(3)}, \boldsymbol{C}(\boldsymbol{B} \odot \boldsymbol{A})^T)$. If we replace $\boldsymbol{B} \odot \boldsymbol{A}$ with an arbitrary binary matrix, this would equal the hypercube segmentation problem defined in [1]: Given a set S of l vertices of the discrete d-dimensional cube $\{0,1\}^d$, find k vertices $P_1, \ldots, P_k \in \{0,1\}^d$ and a partition of S into k segments to maximize $\sum_{i=1}^{k} \sum_{c \in S} \mathsf{sim}(P_i, c)$. Therefore we employ an algorithm that resembles those for hypercube segmentation, with the added restrictions to our centroid vectors.

Algorithm 1 `SaBoTeur` algorithm for the BCPC

Input: A 3-way binary tensor $\boldsymbol{\mathcal{X}}$, number of clusters k, number of samples r.
Output: Binary factor matrices $\boldsymbol{A}$ and $\boldsymbol{B}$; cluster assignment matrix $\boldsymbol{C}$.

1: **function** `SaBoTeur`($\boldsymbol{\mathcal{X}}, k, r$)
2: **repeat**
3: Sample k rows of $\boldsymbol{X}_{(3)}$ into matrix $\boldsymbol{Y}$
4: Find binary matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ that maximize $\mathsf{sim}(\boldsymbol{Y}, (\boldsymbol{B} \odot \boldsymbol{A})^T)$
5: Cluster $\boldsymbol{C}$ by assigning each row of $\boldsymbol{X}_{(3)}$ to its closest row of $(\boldsymbol{B} \odot \boldsymbol{A})^T$
6: **until** r resamples are done
7: **return** best $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$
8: **end function**

4.1 The Algorithm

Alon et al. [1] gave an algorithm for the hypercube segmentation problem that obtains similarity within $(1 - \varepsilon)$ of the optimum. The running time of the algorithm is $\mathrm{e}^{O((k^2/\varepsilon^2)\ln k)}nml$ for n-by-m-by-l data. While technically linear in data size, the first term turns the running time unfeasible even for moderate values of k (the number of clusters) and ε. We therefore base our algorithm on the simpler algorithm by Kleinberg et al. [10] that is based on random sampling. This algorithm obtains an approximation ratio of $0.828 - \varepsilon$ with constant probability and running time $O(nmlk(9/\varepsilon)^k \ln(1/\varepsilon))$. While the running time is still exponential in k, it is dominated by the number of samples we do: each sample takes time $O(nmlk)$ for k clusters and n-by-m-by-l data. For practical purposes, we can keep the number of samples constant (with the cost of losing approximation guarantees, though).

Our algorithm `SaBoTeur` (SAmpling for BOolean TEnsor clUsteRing), Algorithm 1, considers only the unfolded tensor $\boldsymbol{X}_{(3)}$. In each iteration, it samples k rows of $\boldsymbol{X}_{(3)}$ as the initial, unrestricted centroids. It then turns these unrestricted centroids into the restricted type in line 4, and then assigns each row of $\boldsymbol{X}_{(3)}$ to its closest restricted centroid. The sampling is repeated multiple times, and in the end, the factors that gave highest similarity are returned.

The algorithm is extremely simple, which is an asset as it gives a very fast algorithm that, as we shall see in Section 5, also performs very well. In line 3 the algorithm samples k rows of the data as its initial centroids. Kleinberg et al. [10] proved that among the rows of $\boldsymbol{X}_{(3)}$ that are clustered into the same optimal cluster, one is a good approximation of the (unrestricted) centroid of the cluster. Intuitively, then, if we sample one row from each cluster, the sample has a high probability of inducing a close-optimal clustering.

4.2 Binary Rank-1 Matrix Decompositions

The final piece of the `SaBoTeur` algorithm is to turn the unrestricted centroids into the restricted format required by the BCPC problem (line 4). We start by showing that this problem is equivalent to finding the maximum-similarity binary rank-1 decomposition of a binary matrix:

Algorithm 2 Approximate maximum-similarity binary rank-1 decompositions

Input: An n-by-m binary matrix $\boldsymbol{X}$.
Output: Binary vectors $\boldsymbol{a}$ and $\boldsymbol{b}$.

```
1: function A(X)
2:     for all rows x_i of X do
3:         Let b = x_i
4:         Find a maximizing sim(X, ab^T)
5:     end for
6:     return best vectors a and b
7: end function
```

Definition 4 (Binary rank-1 decomposition). *Given an n-by-m binary matrix $\boldsymbol{X}$, find an n-dimensional binary vector $\boldsymbol{a}$ and an m-dimensional binary vector $\boldsymbol{b}$ that maximize* $\mathsf{sim}(\boldsymbol{X}, \boldsymbol{a} \boxtimes \boldsymbol{b})$.

Lemma 1. *Given an k-by-nm binary matrix $\boldsymbol{X}$, finding n-by-k and m-by-k binary matrices $\boldsymbol{A}$, $\boldsymbol{B}$ that maximize* $\mathsf{sim}(\boldsymbol{X}, (\boldsymbol{B} \odot \boldsymbol{A})^T)$ *is equivalent to finding the most similar binary rank-1 approximation of each row $\boldsymbol{x}$ of $\boldsymbol{X}$, where the rows are re-shaped as n-by-m binary matrices.*

Proof. If $\boldsymbol{x}_i$ is the ith row of $\boldsymbol{X}$ and $\boldsymbol{z}_i$ is the corresponding row of $(\boldsymbol{B} \odot \boldsymbol{A})^T$, then $\mathsf{sim}(\boldsymbol{X}, (\boldsymbol{B} \odot \boldsymbol{A})^T) = \sum_{i=1}^{k} \mathsf{sim}(\boldsymbol{x}_i, \boldsymbol{z}_i)$, and hence we can solve the problem row-by-row. Let $\boldsymbol{x} = (x_{1,1}, x_{2,1}, \ldots, x_{n,1}, x_{1,2}, \ldots, x_{n,m})$ be a row of $\boldsymbol{X}$. Re-write $\boldsymbol{x}$ as an n-by-m matrix in column major order.

Consider the row of $(\boldsymbol{B} \odot \boldsymbol{A})^T$ that corresponds to $\boldsymbol{x}$, and notice that it can be written as $(\boldsymbol{b} \otimes \boldsymbol{a})^T$, where $\boldsymbol{a}$ and $\boldsymbol{b}$ be the columns of $\boldsymbol{A}$ and $\boldsymbol{B}$ that correspond to $\boldsymbol{x}$. As $(\boldsymbol{b} \otimes \boldsymbol{a})^T = (b_1\boldsymbol{a}^T, b_2\boldsymbol{a}^T, \cdots, b_m\boldsymbol{a}^T)$, re-writing it similarly as $\boldsymbol{x}$ we get $(\boldsymbol{b} \otimes \boldsymbol{a})^T = (a_1b_1, a_2b_1, \ldots, a_nb_1, a_1b_2, \ldots, a_nb_m) = \boldsymbol{a}\boldsymbol{b}^T = \boldsymbol{a} \boxtimes \boldsymbol{b}$. Thus, we obtain $\mathsf{sim}(\boldsymbol{x}, (\boldsymbol{b} \otimes \boldsymbol{a})^T) = \mathsf{sim}(\boldsymbol{Y}, \boldsymbol{a} \boxtimes \boldsymbol{b})$. □

We present a simple, deterministic algorithm that approximates the maximum similarity within 0.828, Algorithm 2. It is similar to the algorithm for hypercube segmentation based on random sampling presented by Kleinberg et al. [10]. The algorithm considers every row of $\boldsymbol{X}$ as a potential vector $\boldsymbol{b}$ and finds the best $\boldsymbol{a}$ given $\boldsymbol{b}$. Using Lemma 3.1 of [10] it is straight forward to show that the algorithm achieves the claimed approximation ratio:

Lemma 2. *Algorithm 2 approximates the optimum similarity within* 0.828 *in time $O(nm \min\{n, m\})$.*

Proof. To prove the approximation ratio, let $\boldsymbol{a}^*(\boldsymbol{b}^*)^T$ be the optimum decomposition. Consider the rows in which $\boldsymbol{a}^*$ has 1. Per Lemma 3.1 of [10], selecting one of these rows, call it $\boldsymbol{b}$, gives us $\mathsf{sim}(\boldsymbol{X}, \boldsymbol{a}^*\boldsymbol{b}^T) \geq (2\sqrt{2} - 2)\mathsf{sim}(\boldsymbol{X}, \boldsymbol{a}^*\boldsymbol{b}^T)$ (notice that $\boldsymbol{a}^*\boldsymbol{b}^T$ agrees with the optimal solution in rows where $\boldsymbol{a}^*$ is zero). Selecting $\boldsymbol{a}$ that maximizes the similarity given $\boldsymbol{b}$ can only improve the result, and the claim follows as we try every row of $\boldsymbol{X}$.

If $n < m$, the time complexity follows as for every candidate $\boldsymbol{b}$ we have to make one sweep over the matrix. If $m < n$, we can operate on the transpose. □

4.3 Discussion

Lemma 1 gives us yet another way of interpreting BCPC, namely, in BCPC each centroid must be a binary rank-1 matrix. One could define a more general variant where the centroids are arbitrary-rank binary matrices. Between these two extrema is a problem where the (Boolean) ranks of the centroids are bounded from above by some constant $r < \min\{n, m\}$. For such a problem, however, finding the centroids is even harder than it is now, as it essentially requires us to solve the Boolean matrix factorization problem which is a hard problem even to approximate [14].

5 Experimental Evaluation

5.1 Other Methods and Evaluation Criteria

We decided to compare `SaBoTeur` to other Boolean tensor CP methods, and for some real-world experiments we also used a continuous CP method.

The Boolean CP methods we used for comparison were `BCP_ALS` [13] and `Walk'n'Merge` [6]. `BCP_ALS` is based on iteratively updating the factor matrices one at a time (similarly to the classical alternating least squares optimizations), while `Walk'n'Merge` is a recent algorithm for highly scalable Boolean tensor factorization in sparse binary tensors. We did not use `Walk'n'Merge` on synthetic data as `BCP_ALS` is expected to perform better on smaller and denser tensors [6] but we used it on larger real-world tensors; `BCP_ALS`, on the other hand, does not scale well to larger tensors and hence we had to omit it from some experiments.

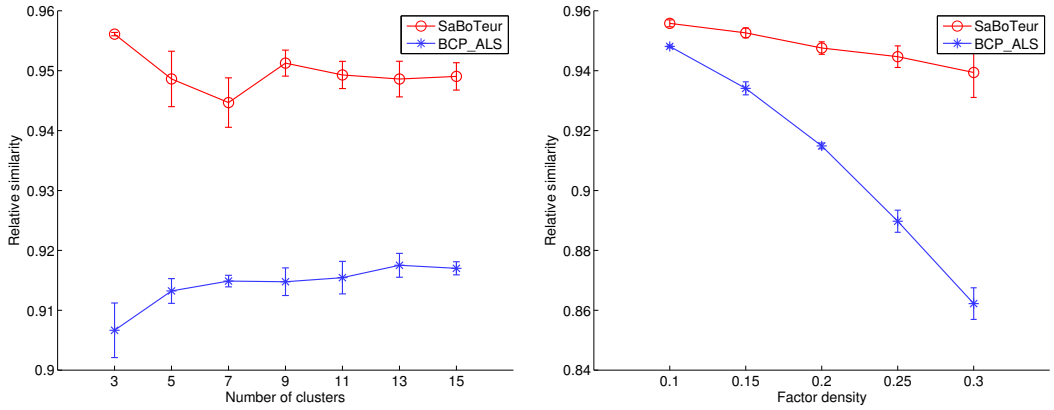
Of the continuous methods we used `CP_APR` [4] (implementation from the Matlab Tensor Toolbox v2.5[1]), an alternating Poisson regression algorithm that is specifically developed for sparse (counting) data.

For synthetic data, we report the relative similarity, that is, the fraction of the elements where the data and the clustering agree. For real-world data, we report the error measured using the squared Frobenius norm. This norm however can help the real-valued methods, as it scales all errors less than 1 down, but at the same time, small errors cumulate unlike with fully binary data. To alleviate this problem, we also rounded the reconstructed tensors from `CP_APR` to binary tensors. From different rounding thresholds between 0 and 1 we selected the one that gave the lowest (Boolean) reconstruction error.

5.2 Synthetic Experiments

To test the `SaBoTeur` algorithm in a controlled environment we created synthetic data sets that measured the algorithm's response to (1) different numbers of clusters, (2) different density of data, and (3) different levels of noise. All tensors were 700-by-500-by-50. All data sets were created by first creating ground-truth binary factor matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$. The default number of clusters was 7 and

[1] `http://www.sandia.gov/~tgkolda/TensorToolbox/`



(a) Varying number of clusters. (b) Varying density of the factors.

Fig. 1. Synthetic experiment results. Markers are at the mean over five random tensors and the width of the errorbars is twice the standard deviation.

the default density of $\boldsymbol{A}$ and $\boldsymbol{B}$ was 0.2. A symmetric random noise was applied to the tensor and flipped 4% of the elements by default.

We varied each of the three features one at a time keeping the others in their default values, and created 5 random copies on each parameter combination. The results we report are mean values over these five random copies. In all experiments, the number of clusters (or factors) was set to the true number of clusters used to create the data. The number of re-samples in `SaBoTeur` was set to $r = 20$ in all experiments. We only used `BCP_ALS` to compare against in the synthetic experiments.

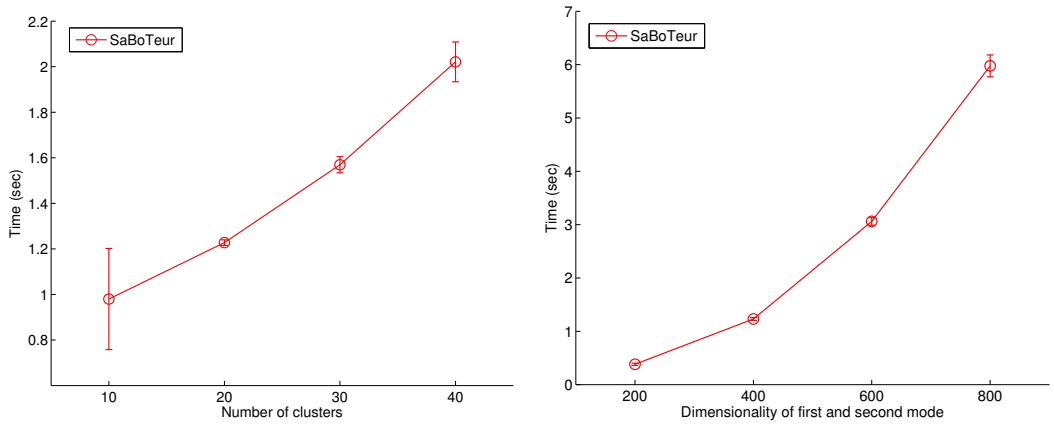
Varying the number of clusters. The number of clusters varied from 3 to 15 with steps of 2. The results are shown in Figure 1a. Perhaps the most surprising result here is how much better `SaBoTeur` is compared to `BCP_ALS`, especially given that `SaBoTeur`'s answer is a valid Boolean CP decomposition.

Varying the density. The density of the factor matrices varied from 10% to 30% with steps of 5%. The results can be seen in Figure 1b. Again `SaBoTeur` is better than `BCP_ALS`: it has gradually declining slope for increased density, whereas `BCP_ALS`'s results dive much faster.

Varying the noise. In the final synthetic experiment, we varied the noise level between 2% and 10% with steps of 2%. As is to be expected, increasing the noise decreases the results of both algorithms. Both algorithms exhibit roughly linear decrease in the similarity w.r.t. noise level, but `SaBoTeur` is again consistently the better of the two (results not shown).

Scalability. `SaBoTeur` is implemented in Matlab[2] and we run the scalability tests on a dedicated machine with 8 Intel Xeon E5530 2.4GHz processors and 48GB

[2] The code is available from `http://www.mpi-inf.mpg.de/~pmiettin/btc/`



(a) Varying the number of clusters. (b) Varying the dimensionality of the data.

Fig. 2. Scalability experiment results. Markers are at the mean over five random tensors and the width of the errorbars is twice the standard deviation.

of main memory. All reported times are wall-clock times. For these experiments, we created a new set of tensors. First, we tested the effect the number of clusters has to the algorithm. The data was 400-by-400-by-80 and the number of clusters varied from 10 to 40 with steps of 10. As can be seen in Figure 2a, we observe that the algorithm scales almost-linearly with the number of clusters. The slight non-linearity is due to the increasing number of non-zeros in the data in higher values of k. For the second experiment, we varied the dimensionality of the first and second mode between 200 and 800 with steps of 200. The results can be seen in Figure 2b, where we observe close-to-quadratic behavior, in line with the theoretical running time of the algorithm.

Discussion. The synthetic experiments confirm that `SaBoTeur` is capable of recovering the latent cluster structure from the synthetic data sets. Arguably the most surprising result of the synthetic experiments was that `SaBoTeur` was consistently better than `BCP_ALS`, even though the latter has more freedom to obtain better solutions.

5.3 Real-World Data

Datasets and earlier experiments. We tested `SaBoTeur` with three real-world data sets: The Resolver data contains entity–relation–entity tuples from the TextRunner open information extraction algorithm[3] [15]. A sample of size 343-by-360-by-200 (entity-by-entity-by-relation) was used for the experiments. The Enron data[4] (146-by-146-by-38) contains information about who sent e-mail to whom (rows and columns) per months (tubes). The TracePort data set[5] (10 266-by-8 622-by-501)

[3] `http://www.cis.temple.edu/~yates/papers/jair-resolver.html`
[4] `http://www.cs.cmu.edu/~enron/`
[5] `http://www.caida.org/data/passive/passive_2009_dataset.xml`

Table 1. Reconstruction errors rounded to the nearest integer. '—' denotes that the experiment was not conducted. Part of the results are from [6] and [13].

Algorithm	Enron	TracePort	Resolver
SaBoTeur	1 765	10 946	1 488
BCP_ALS	1 850	—	1 492
Walk'n'Merge	1 753	10 968	—
CP_APR	1 619	11 069	1 497
CP_APR $_{0/1}$	1 833	11 121	1 543

contains anonymized passive traffic traces (source and destination IP and port numbers) from 2009. With Enron and TracePort data sets, for Walk'n'Merge, CP_APR and ParCube we used the results from [6]. The results for BCP_ALS and Resolver are from [13]. The number of clusters/factors was set to $k = 15$ except for Enron data, for which it was $k = 12$.

Results. The results with the real-world data sets can be seen in Table 1. SaBoTeur continues its impressive results, being roughly on par with the other binary methods and with TracePort even better than the continuous method, CP_APR. In conjunction with what we observed in the synthetic setting, SaBoTeur consistently outperforms BCP_ALS despite solving a more restricted problem.

6 Conclusions and Future Work

We have studied the problem of clustering one mode of a 3-way binary tensor while simultaneously reducing the dimensionality in the two other modes. This problem bears close resemblance to the Boolean CP tensor decomposition, but the additional clustering constraint makes the problem significantly different. The main source of computational complexity, the consideration of overlapping factors in the tensor decomposition, does not play a role in BCPC. This lets us design algorithms with provable approximation guarantees better than what is known for the Boolean matrix and tensor decompositions.

Our experiments show that the algorithm for BCPC, SaBoTeur, is better than the dedicated (Boolean) tensor decomposition algorithms in building a Boolean CP decomposition of a tensor. Sometimes SaBoTeur also outperforms continuous methods for non-Boolean CP decomposition.

The essential piece, the maximum-similarity binary rank-1 approximation achieves an approximation ratio of 0.828 in $O(nm \min\{n, m\})$ time, and with that dominates the running time. Faster algorithms for the rank-1 approximation (with better approximation guarantees) would have an instant impact on SaBoTeur.

For the rank-1 approximation, the running time of $O(nm \min\{n, m\})$ is ascribed to the fact that every row of $\boldsymbol{X}$ (resp. every column if $n < m$) is tried as candidate. Because $\boldsymbol{a}^* \boldsymbol{b}^T$ already agrees with the optimal solution in rows where $\boldsymbol{a}^*$ is zero, it would be enough to take the non-zero elements into account rather than the complete rows. This modification is in particular beneficial to

sparse settings. Further improvement in terms of speed could be gained from parallelization of the algorithm, possibly using a MapReduce model. Each of the two concatenated loops in Algorithm 2 could be executed in parallel. Also early stopping might be advantageous if no rows are left that could improve the result. This might however require an additional step of sorting the rows according to their number of non-zero elements. Future investigations need to show which parallel configuration is most beneficial. Another aspect of refinement concerns the introduction of data structures and operations tailored to Boolean algebra. Storing the data as bit vectors and the use of native bit operations might yield additional speedup. At the same time this reduces the space needed to store the data compared to the representation as vectors of numbers.

Overall, this paper covered an extreme of the Boolean tensor clustering: each centroid was restricted to a rank-1 binary matrix. In future research, we hope to better cover the spectrum between this problem and the other extreme, clustering the frontal slices of the tensor with no reduction on the other modes.

References

1. N. Alon and B. Sudakov. On two segmentation problems. *J. Algorithm*, 33:173–184, 1999.
2. R. Belohlavek, C. Glodeanu, and V. Vychodil. Optimal Factorization of Three-Way Binary Data Using Triadic Concepts. *Order*, 30(2):437–454, Mar. 2012.
3. L. Cerf, J. Besson, K.-N. T. Nguyen, and J.-F. Boulicaut. Closed and noise-tolerant patterns in n-ary relations. *Data Min. Knowl. Discov.*, 26(3):574–619, 2013.
4. E. C. Chi and T. G. Kolda. On Tensors, Sparsity, and Nonnegative Factorizations. *SIAM J. Matrix Anal. Appl.*, 33(4):1272–1299, Dec. 2012.
5. D. Erdős and P. Miettinen. Discovering Facts with Boolean Tensor Tucker Decomposition. In *CIKM '13*, pages 1569–1572, 2013.
6. D. Erdős and P. Miettinen. Walk'n'Merge: A Scalable Algorithm for Boolean Tensor Factorization. In *ICDM '13*, pages 1037–1042, Dec. 2013.
7. H. Huang, C. Ding, D. Luo, and T. Li. Simultaneous tensor subspace selection and clustering: The equivalence of high order svd and k-means clustering. In *KDD '08*, pages 327–335, Aug. 2008.
8. D. I. Ignatov, S. O. Kuznetsov, J. Poelmans, and L. E. Zhukov. Can triconcepts become triclusters? *Int. J. Gen. Syst.*, 42(6):572–593, Aug. 2013.
9. S. Jegelka, S. Sra, and A. Banerjee. Approximation Algorithms for Tensor Clustering. In *ALT '09*, pages 368–383. Springer Berlin Heidelberg, 2009.
10. J. M. Kleinberg, C. H. Papadimitriou, and P. Raghavan. Segmentation problems. *J. ACM*, 51(2):263–280, Mar. 2004.
11. T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009.
12. X. Liu, L. De Lathauwer, F. Janssens, and B. De Moor. Hybrid Clustering of Multiple Information Sources via HOSVD. In *ISNN '10*, pages 337–345. Springer Berlin Heidelberg, 2010.
13. P. Miettinen. Boolean Tensor Factorizations. In *ICDM '11*, pages 447–456, 2011.
14. P. Miettinen, T. Mielikäinen, A. Gionis, G. Das, and H. Mannila. The Discrete Basis Problem. *IEEE Trans. Knowl. Data En.*, 20(10):1348–1362, Oct. 2008.
15. A. Yates and O. Etzioni. Unsupervised methods for determining object and relation synonyms on the web. *J. Artif. Intell. Res.*, 34:255–296, 2009.

On Improving Operational Planning and Control in Public Transportation Networks using Streaming Data: A Machine Learning Approach

Luis Moreira-Matias[1,2],
João Mendes-Moreira[2,3], João Gama[2,4], and Michel Ferreira[1,5]

[1] Instituto de Telecomunicações, 4200-465 Porto, Portugal
[2] LIAAD-INESC TEC, 4200-465 Porto, Portugal
[3] DEI-FEUP, U. Porto, 4200-465 Porto, Portugal
[4] Faculdade de Economia, U. Porto 4200-465 Porto, Portugal
[5] DCC-FCUP, U. Porto, 4169-007 Porto, Portugal
{luis.m.matias, joao.mendes.moreira, jgama}@inescporto.pt,
michel@dcc.fc.up.pt

Abstract. Nowadays, transportation vehicles are equipped with intelligent sensors. Together, they form collaborative networks that broadcast real-time data about mobility patterns in urban areas. Online intelligent transportation systems for taxi dispatching, time-saving route finding or automatic vehicle location are already exploring such information in the taxi/buses transport industries. In this PhD spotlight paper, the authors present two ML applications focused on improving the operation of Public Transportation (PT) systems: 1) Bus Bunching (BB) Online Detection and 2) Taxi-Passenger Demand Prediction. By doing so, we intend to give a brief overview of the type of approaches applicable to these type of problems. Our frameworks are straightforward. By employing online learning frameworks we are able to use both historical and real-time data to update the inference models. The results are promising.

Keywords: GPS data, Public Transportation, Bus Bunching, Taxi-Passenger Demand, Probabilistic Reasoning, Online Learning.

1 Introduction

The increasing number of running road vehicles worldwide is enlarging the complexity of transportation networks - especially of its design and operations. Therefore, it is becoming more difficult to maintain the reliability of this means of transportation, thereby decreasing passenger satisfaction. On the other hand, rising fuel price are increasing its operational costs.

GPS (Global Positioning System) devices are already in place in many of these networks. There are also many **Intelligent Transportation Systems (ITS)** that already successfully explore this kind of data, such as Intelligent Routing [1], Bus Travel Time prediction [2] or efficient taxi dispatching [3].

Despite the intrinsic online characteristics regarding the aforementioned problems, many of the techniques employed are batch learners - which are not prepared to detect the concept drift often introduced by unexpected events, which emerge in the system, such as traffic jams or a massive demand.

In fact, GPS data is mainly an unbounded stream of data. This kind of data is produced continuously at a high speed from multiple locations and time granularities, while its distribution may change over time (e.g. a continuous but speedy vehicular flow on a highway is heavily decreased by a traffic jam). However, state-of-art Machine Learning (ML) algorithms have strong assumptions - such as stationary data distributions and the existence of finite training sets - that make their application to this kind of data inefficient or even useless [4]. On the other hand, techniques that **learn from data streams** have seen great development over the last decade and there has also been an increase in applications used in many sensor networks – such as our own – and which have already proved to be efficient in dealing with these characteristics.

In this PhD thesis, we intend to explore such techniques related to GPS data streams broadcasted by the vehicular networks comprised in Public Transportation (PT) networks, namely, 1) from buses and 2) from taxis. Our goal is to produce ITS applications that will increase the profitability of companies, by providing important information that otherwise would not be possible to mine. The above mentioned networks have common characteristics and synergies that should be explored together. They can be enumerated as follows:

1. Both provide a **continuous stream of data** about the network's behavior, based not only on **vehicle location**, but also on other status variables, such as the number of passengers travelling within or of a mechanical nature.
2. Both enclose **vehicular networks**, and their operations rely on the a) **dependences** and/or b) **correlations** between vehicle behaviors. Some examples of these could be a) the delay propagation effect of a highly frequented bus route, introduced by a vehicle that is failing to fulfill its schedule, or b) the expected distance of a taxi service, departing from a location of interest, given that the last N vehicles, which departed from such a spot, experienced a cruising distance greater (or smaller) than a given time threshold.
3. The **passenger demand** in PT services, provided by such networks, is highly dependent on the **regularities of human behavior**, such as the sleep period at night, or the difference between travel origins and destinations on weekdays/weekends.
4. The **planning** of these networks is highly dependent on **seasonal events exhibited during the year**, as in periods of school holidays or over the Christmas season. Important planning stages of these networks (e.g.: the location of taxi stands in a given urban area, or the planning of bus schedules) are relevant examples of the dependency in place.
5. The **real-time control** of both is highly sensitive to **anomalous demand events** that may unbalance the expected relationship between service offer and demand - and thereby provoke **unexpected disruptions** in such services. Examples of this issue could be overcrowded buses caused by large

scale events (e.g. sporting events, concerts, etc.), which may cause a temporary absence of taxi offers in some locations, due to an exponential increase in passenger demand.

The aforementioned characteristics represent similarities that are reflected in the data provided, namely, 1) by exhibiting the same periodicity of existing regularities (daily, weekly) and 2) common passenger origin/destination matrices; 3) by revealing the existence of anomalous demand events (thereby allowing their detection in both time and space) or 4) even common data distributions of the time series of passenger count. Consequently, such streaming data provides opportunities to improve both the operational planning and control of networks, by exploring methods to learn and therefore identifying these patterns.

In this spotlight paper, we present two ML frameworks for solving two real-world problems: 1) Bus Bunching (BB) Online Detection and the 2) Taxi- Passenger Demand Prediction. By doing so, we intend to give a brief overview of the type of approaches applicable to these type of problems.

2 Case Study

Our case study took place in the city of Porto in Portugal. Two data streams of events from two PT companies operating in Porto were used to evaluate our approaches. This city is the center of a medium-sized urban area, consisting of 1.3 million inhabitants.

To test our BB online detection framework, we used data collected from STCP, the Public Transport Operator in Porto. It describes trips of three distinct lines (A, B, C) during 2010. Each line has two routes - one for each way A1, A2, B1, B2, C1, C2. Line A is a common urban line between *Viso* (an important neighbourhood in Porto) and *Sá da Bandeira*, a downtown bus hub. Line B is also an urban line, although it is an arterial one. It traverses the main interest points in the city by connecting two important street markets: *Bolhão* - located in the downtown area - and *Mercado da Foz*, in the most luxurious neighbourhood in the city. Line C connects the city's downtown area to the farthest large-scale neighborhood in the region (*Maia*).

Concerning the second application case, we focused on the event data stream from a taxi company (which contains 441 running vehicles) operating in Porto, Portugal, between August 2011 and April 2012. This dataset contains information about more than one million fared trips.

3 Bus Bunching Online Detection

It is known that some schedule instability exists, especially in highly frequent routes (10 minutes or less). In these kinds of routes the **headway** (time separation between vehicle arrivals or departures) regularity is more important than the fulfilment of the arrival time at bus stops. In fact, a small delay in a bus'

arrival increases the number of passengers at the next stop. This number increases the dwell time (time period where the bus is stopped at a bus stop). On the other hand, the next bus will have fewer passengers and shorter dwell times without delays. This will continue as a snow ball effect and, at a further point of that route, the two buses will meet at a bus stop, forming a platoon. This phenomenon is denominated as **Bus Bunching(BB)**.

3.1 Related Work

One of the first works to address the BB phenomenon was presented by Powell and Sheffi [5]. After this paper, many other works followed the stability concept (i.e. if we guarantee a stable headway, BB events will never emerge) by constantly introducing corrective actions into the system. Some examples are the work in [6], where each bus is an agent that negotiates with the other buses about which bus is holding up time at each stop or in [7], where the negotiation is cantered around the cruising speed.

The employment of historical data to address this problem is very recent. In [8], a model to determine the optimal holding time at each stop based on real-time location is presented. Delgado et al. [9] also suggested preventing passengers from boarding by establishing maximum holding times to maintain the headway stable. The efficiency of these types of frameworks is usually demonstrated through simulations, assuming 1) stochastic demand or 2) using historical data. Despite their usefulness, all these works do not account for the use of both historical and real-time data simultaneously. Moreover, they have low interpretability because their outputs do not provide any insight on what the best corrective action is. The predictive method presented in the next section is able to deal with the network's stochasticity, regardless of which corrective action we opt to take. Finally, it is important to highlight that the majority of the work described in the literature tries to maintain a stable headway at the cost of some schedule uncertainty (introduced by the constant corrective actions), despite the existing risk of forming a bus platoon at a further stop.

3.2 Methodology

The most important variable in regards to BB events is the headway (i.e. h). Theoretically, the headway between two consecutive trips should be constant. However, due to stochastic events that arise during bus trips, the headway suffers some variability. BB does not only occur when a bus platoon is formed, but occurs as soon as the headway becomes unstable. The headway between two consecutive buses is defined as unstable whenever it is strictly necessary to apply a corrective action in order to recover the headway value to acceptable levels. Such a threshold is usually defined in function of the frequency f = h1 (time between the departure of two consecutive buses) [10].

Let the arrival time be defined as $T_{k,i+1} = T_{k,i} + dw_{k,i} + CTT_{k,i,i+1}$ where $dw_{k,i}$ denotes the dwell time at the stop i and $CTT_{k,i,i+1}$ stands for the Cruise Travel Time between those two consecutive stops. Therefore, it is possible to

anticipate the occurrence of BB events if we are able to predict the value of $dw_{k,i} + CTT_{k,i,i+1}$, which is often denominated by **Link Travel Time** (LTT) [11]. Let the LTT Prediction be defined as an offline **regression** problem where the target variable is the cruising time between two consecutive bus stops. Such predictions are computed on a daily basis (the forecasting horizon) using the θ most recent days (the learning period) to train our model. Consequently, we obtain a set of predictions for all the t trips of the day denoted as $\mathbb{P} = \bigcup_{i=1}^{t} P_i = \{P_{1,1}, P_{1,2}, ..., P_{1,s}, ..., P_{t,s}\}$. $\mathbb{P}$ is then incrementally refined in two steps: 1) trip-based and 2) stop-based. Both steps are based on the Perceptron's Delta Rule [12] by reusing each prediction's **residuals** to improve the further ones.

Let e denote the last trip completed before the current trip starts (i.e. c). The trip-based refinement consists of comparing the predictions of e $P_e = \{P_{e,1}, P_{e,2}, ..., P_{e,s}\}$ with the real times T_e to update P_c. Firstly, we compute the residuals as $R_e = T_e - P_e$ and then its average value as $\nu_e = \sum_{i=1}^{s} \frac{T_{e,i} - P_{e,i}}{s}$. Secondly, an user-defined parameter $0 < \alpha << 1$ is employed to set a threshold th able to identify trips where the error is larger than expected. Consequently, $th = \alpha * f_e$. Three other variables are then defined: $\vartheta_p = 0, \vartheta_n = 0$ and $\beta' = \beta$. The first two are counters that are incremented whenever the prediction error is heading to the same way (positive/negative) in consecutive trips (e.g. if $\mu_e > th$ ϑ_p is incremented; otherwise, $\vartheta_p = 0$). The beta value β' stands for the residual's percentage to be added to P_c (its initial value is user-defined). It is initialized with another user-defined parameter $0 < \beta << 1$ and updated according to a user-defined learning rate $0 < \kappa <= 1$. Consequently, if ϑ_p or ϑ_n are incremented, the P_c and β' are updated as $P'_c = P_c \pm (\beta' \times P_c)$ and $\beta' = \beta' + \vartheta * (1 + \kappa) * \beta$, respectively. If both ϑ stay the same, β' resumes its original value as $\beta' = \beta$. The error tests are always performed over the regression results P_c and not over the updated arrays P'_c. These updates are performed incrementally for every trip.

Given the updated predictions of two consecutive trips (P'_c, P'_{c+1}), it is possible to obtain the predicted headways $E_c = P'_{c+1} - P'_c$ (i.e. an offline prediction). The second refinement uses the headway residuals $HR_c = H_c - E_c$ to update E_c stop-by-stop. Incrementally, we can obtain online headway predictions as $E'_{c,i} = H_{c,i-1} + E_{c,i} - E_{c,i-1}, \forall i \in \{2, s\}$. The problem resides in updating the headway online prediction for the next stop $E'_{c,i}$ given the value of $HR_{c,i-1}$. Let $\gamma' = \gamma$ be the residual's percentage to add to the prediction where its initial value for each trip $(0 < \gamma << 1)$ is an user-defined parameter. $E'_{c,i}$ can be updated as $E''_{c,i} = E'_{c,i} + (HR_{c,i-1} * \gamma')$. Finally, γ' is also updated by comparing the residuals of E_c and E'_c (HR_c and HR'_c, respectively). If $|HR_c| > |HR'_c|$, then $\gamma' = \gamma' * (1 - \gamma)$. Otherwise, $\gamma' = \gamma' * (1 + \gamma)$. The progression of γ' is bounded by an user-defined domain $[\gamma_{min}, \gamma_{max}]$. The value of $E''_{c,i}$ is also used to update the offline predictions for further stops as $E'_{c,j} = E''_{c,j-1} + E_{c,j} - E_{c,j-1} \wedge j = i + 1$ and $E'_{c,j} = E'_{c,j-1} + E_{c,j} - E_{c,j-1}, \forall j \in [i + 2, s]$.

3.3 Experimental Results

In the offline regression problem, a state-of-art algorithm was employed: Random Forest (RF). We did so by following previous work about this topic, which used

data from the same source [13]. The experiments were conducted using the R Software [14]. A sensitivity analysis was conducted on the regression parameters. The best parameter setting was `mtry=3` and `ntrees=750`. The learning period used was $\theta = 7$ days by employing our domain-knowledge. The error threshold to trigger the inter-trip update rule was set to $\alpha = 0.05$ while the initial value for the residual's percentage to be employed is $\beta = 0.01$. The learning rate *kappa* was set to 0.3. The initial residual's percentage employed on the stop-based update rule is $\gamma = 0.1$ while its domain is $\gamma \in [0.005, 0.3]$. Finally, the ρ was set to 360 seconds. All these parameters were set by employing an apriori cross-validation test on some close range of values.

It is possible to divide the evaluation of our framework into two distinct contexts: (i) the Mean Absolute Error (MAE) and (ii) the BB detection accuracy. With the first one, we employed a prequential evaluation [15] by evaluating *just* the prediction made for the LTT performed for the next bus stop. We did so by using the MAE on (1) the offline regression output and then on the (2) inter-trip and (3) intra-trip refinement. In the BB detection context, Accuracy, Precision and Recall were used as evaluation metrics. A weighted Accuracy was also employed, by weighing the trips where a BB event emerged ten times more than the remaining ones. The Average Number of Stops Ahead is also displayed to show which our forecasting horizon is. The results are presented in Table 1. More than only just identifying a problematic link or stop, the BB online detection framework also identifies the vehicle pair where a corrective action must be made. In the current dataset, it was able to detect BB events thirteen stops ahead (on average), which gives more than enough room to perform any of the four possible corrective actions. Despite its achievements, this framework also presents some limitations, namely, with the regression task and with the employed parameters. The regression task was tested using only one algorithm. Even though considering that it presented good results with similar data [13], we do not know if there is another that could perform better, by using a similar computational effort. On the other hand, both the prediction refinements and

Table 1: Experimental results. The times are in seconds. The ALL column contains the average for the first two spans and the sum for the last one.

	A1	**A2**	**B1**	**B2**	**C1**	**C2**	**ALL**
MAE offline regression	1356.96	643.99	1475.22	1871.01	473.61	2776.57	1432.88
MAE inter-trip update	148.85	92.91	124.99	148.85	40.65	123.77	113.34
MAE incremental update	13.21	26.35	22.67	13.21	31.79	27.47	22.45
Accuracy	97.99%	96.34%	97.08%	97.83%	96.63%	93.83%	96.62%
Weighted Accuracy	93.97%	93.57%	94.57%	95.52%	95.73%	91.51%	94.14%
Precision	65.88%	40.85%	41.53%	45.70%	69.44%	51.67%	52.51%
Recall	81.81%	83.18%	83.07%	83.24%	94.48%	87.95%	85.62%
Avg. Nr. of Stops Ahead	11.85	14.78	13.88	15.01	12.96	14.52	13.83
Correct BB Predictions	558	460	363	303	1811	1497	4992
Real BB Events	682	553	437	364	1917	1702	5655

the event detection framework rely on a large set of parameters. To get a fair parameter setting can be a hard task – especially if the user has no expertise with the case study approached.

4 Taxi-Passenger Demand Prediction

Taxi driver mobility intelligence is an important factor to maximize both profit and reliability within every possible scenario. Knowledge about where services will actually emerge can be an advantage for the driver – especially when there is no economic viability to adopt random cruising strategies to find passengers. The GPS historical data is one of the main data sources for this topic because it can reveal underlying running mobility patterns. Multiple works in the literature have already successfully explored this type of data with various applications, such as modelling the spatiotemporal structure of taxi services [16], building passenger-finding strategies [17] or even predicting taxi location through a passenger-perspective [18]. Despite their useful insights, most techniques reported are tested using offline test-beds, discarding some of the main advantages of this type of signal. In other words, they do not provide any live information about the location of a passenger or the best route to pick-up a passenger at the current date/time (i.e. real-time performance).

4.1 Methodology

Let $S = \{s_1, s_2, ..., s_N\}$ be the set of N taxi stands of interest and $D = \{d_1, d_2, ..., d_j\}$ be a set of j possible passenger destinations. Consider $X_k = \{X_{k,0}, X_{k,1}, ..., X_{k,t}\}$ to be a discrete time series (aggregation period of P-minutes) for the number of demanded services at a taxi stand k. Our goal is to build a model which determines the set of service counts $X_{k,t+1}$ for the instant $t + 1$ per each taxi stand $k \in \{1, ..., N\}$. To do so, we propose three distinct short-term prediction models: 1) a Time Varying Poisson Model which handles the long term memory; 2) a Weighted Time Varying Poisson Model, which sets weights to each data point according with its date - where the most recent points weigh more than the older ones and the weights are calculated through an Exponential Smoothing model; 3) an ARIMA model which handles the short-term memory through its high reactibility to bursty changes to the process in place. The output prediction is an Ensemble of the outputs produced by the aforementioned methods. We employed a Sliding Window ensemble that computes and weighs the average of those outputs. The weights are inverse to their error in the *most recent* samples.

This model is deeply described in section III in [19]. Moreover, an extended version of this framework was also presented in [20], where both the Poisson and the ARIMA models were extended to be calculated **incrementally**.

4.2 Experimental Results

The model was programmed using R language [14]. The prediction periodicity was set to 5 minutes. Both the ARIMA model and the weight set were firstly

set (and updated every 24h) through learning the underlying model (i.e. autocorrelation and partial autocorrelation analysis) running on the historical time series curve of each stand, during the last two weeks. To do so, we used an automatic time series function in the `[forecast]` R package - *auto-arima* and the *arima* function from the built-in R package `[stats]`. The Time Varying Poisson averaged models (both weighted and non-weighted) were also updated every 24 hours.A sensibility analysis carried out with data previous to the one used on these experiments determined the optimal values for the parameters α, β and H as 0.4, 0.01 and 4 (i.e. a sliding window of 20 minutes), respectively.

The error measured for each model is presented in Table 2. The results are firstly presented per shift and then globally. The overall performance is good: the maximum value of the error using the ensemble was **25.90%** during the evening shift. The sliding window ensemble is always the best model in every shift and case study considered. The models just present slight discrepancies within the defined shifts. Our model took - on average - 37.92 seconds to build the next prediction about the spatiotemporal distribution the demand by all stands.

4.3 Related Work

Little research regarding the demand prediction problem exists. Kaltenbrunner *et al.* [21] detected the geographic and temporal mobility patterns over data acquired from a bicycle network running in Barcelona. The authors' goal was to forecast the number of bicycles at a station to improve their deployment. Yuan *et. al* presented in [22] a complete work containing methods about a) how to divide the urban area into pick-up zones using spatial clustering; b) how a passenger can find a taxi; and c) which trajectory is the best to pick-up the next passenger. Although its results are promising, it is focused on improving the trajectory of a single driver, disregarding the position of the remaining drivers.

The most similar work to our own is presented by Li *et. al* [16]. The authors present a recommendation system to improve the drivers' mobility intelligence. To do so, data from a taxi network running in Hangzhou, China was used. Firstly, they calculated the city hotspots - urban areas where pick-ups occur more frequently. Secondly, they used ARIMA to forecast the number of pick-ups at these hotspots over periods of 60 minutes. Thirdly, they presented an

Table 2: Error Measured on the Models using $sMAPE$.

Model	**Periods**			
	00h−08h	**08h−16h**	**16h−00h**	**24h**
Poisson Mean	27.67%	24.29%	25.27%	25.32%
W. Poisson Mean	27.27%	24.62%	25.66%	25.28%
ARIMA	28.47%	24.80%	25.60%	26.21%
Ensemble	**24.86%**	**23.14%**	**24.07%**	**23.77%**

improved ARIMA depending both on time and day type. Despite their good results, it just uses the most immediate historical data, discarding the mid and long-term memory of the system; moreover, the proposed approach assumes fixed periods of 60 minutes (i.e. there is just one prediction per hour). Our framework is *incremental*. By doing so, it is able to produce predictions **on-demand**, which is a true advantage facing the real-time characteristics of this type of decisions.

5 Final Remarks

GPS devices are now widespread. Taking full advantage of this rich source of spatiotemporal data to support daily human activities comprises a relevant challenge for the Data Mining community. In this PhD spotlight paper, the authors presented two ML applications focused on improving PT operations. The results are promising. However, there are still many issues to be solved over a mid-term period. The demand prediction must be used to perform commendations in the most profitable urban areas to go to pick-up the drivers' next service. The BB detection framework still requires a large-scale sensitivity analysis of its parameter set. How to select the most adequate corrective action for each situation will also be addressed in our future work. Using both data sources simultaneously on one of these problems can also provide a step forward in this research area. However, further research must be employed to know how beneficial such *shared* knowledge may be.

Acknowledgments

The authors would like to thank Geolink, Lda. and STCP for the data supplied. This work was supported by the VTL: "Virtual Traffic Lights" (PTDC/EIA-CCO/118114/2010), by MAESTRA (ICT-2013-612944), by ERDF - European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and also by the Portuguese Funds through the FCT (Portuguese Foundation for Science and Technology) within project FCOMP-01-0124-FEDER-037281.

References

1. Zhu, W., Boriboonsomsin, K., Barth, M.: Defining a freeway mobility index for roadway navigation. Journal of Intelligent Transportation Systems **14**(1) (2010) 37–50
2. Chien, S., Ding, Y., Wei, C.: Dynamic bus arrival time prediction with artificial neural networks. Journal of Transportation Engineering **128**(5) (2002) 429–438
3. Lee, J., Park, G., Kim, H., Yang, Y., Kim, P., Kim, S. In: A Telematics Service System Based on the Linux Cluster. Volume 4490 of LNCS. Springer Berlin / Heidelberg (2007) 660–667
4. Gama, J.: Knowledge discovery from data streams. Chapman and Hall/CRC (2010)

5. Powell, W., Sheffi, Y.: A probabilistic model of bus route performance. Transportation Science **17**(4) (1983) 376–404
6. Gershenson, C., Pineda, L.: Why does public transport not arrive on time? the pervasiveness of equal headway instability. PloS one **4**(10) (2009) 72–92
7. Daganzo, C., Pilachowski, J.: Reducing bunching with bus-to-bus cooperation. Transportation Research Part B: Methodological **45**(1) (2011) 267–277
8. Daganzo, C.: A headway-based approach to eliminate bus bunching. Transportation Research Part B **43**(10) (2009) 913–921
9. Delgado, F., Muñoz, J.C., Giesen, R., Cipriano, A.: Real-time control of buses in a transit corridor based on vehicle holding and boarding limits. Transportation Research Record: Journal of the Transportation Research Board **2090**(1) (2009) 59–67
10. Moreira-Matias, L., Ferreira, C., Gama, J., Mendes-Moreira, J., de Sousa, J.: Bus bunching detection by mining sequences of headway deviations. In: Advances in Data Mining. Applications and Theoretical Aspects. Volume 7377 of LNCS., Springer. (2012) 77–91
11. Chen, G., Yang, X., An, J., Zhang, D.: Bus-arrival-time prediction models: Link-based and section-based. Journal of Transportation Engineering **138**(1) (2011) 60–66
12. Rosenblatt, F.: The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review **65**(6) (1958) 386
13. Mendes-Moreira, J., Jorge, A., de Sousa, J., Soares, C.: Comparing state-of-the-art regression methods for long term travel time prediction. Intelligent Data Analysis **16**(3) (2012) 427–449
14. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. (2012)
15. Dawid, A.: Present position and potential developments: Some personal views: Statistical theory: The prequential approach. Journal of the Royal Statistical Society. Series A (General) **147** (1984) 278–292
16. Li, X., Pan, G., Wu, Z., Qi, G., Li, S., Zhang, D., Zhang, W., Wang, Z.: Prediction of urban human mobility using large-scale taxi traces and its applications. Frontiers of Computer Science in China **6**(1) (2012) 111–121
17. Li, B., Zhang, D., Sun, L., Chen, C., Li, S., Qi, G., Yang, Q.: Hunting or waiting? discovering passenger-finding strategies from a large-scale real-world taxi dataset. In: 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops). (March 2011) 63–68
18. Yuan, J., Zheng, Y., Xie, X., Sun, G.: Driving with knowledge from the physical world. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2011) 316–324
19. Moreira-Matias, L., Gama, J., Ferreira, M., Mendes-Moreira, J., Damas, L.: Predicting taxi-passenger demand using streaming data. IEEE Transactions on Intelligent Transportation Systems **14**(3) (2013) 1393–1402
20. Moreira-Matias, L., Gama, J., Ferreira, M., Mendes-Moreira, J., Damas, L.: On predicting the taxi-passenger demand: A real-time approach. In: Progress in Artificial Intelligence. Volume 8154 of LNCS. Springer (2013) 54–65
21. Kaltenbrunner, A., Meza, R., Grivolla, J., Codina, J., Banchs, R.: Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system. Pervasive and Mobile Computing **6**(4) (2010) 455–466
22. Yuan, J., Zheng, Y., Zhang, L., Xie, X., Sun, G.: Where to find my next passenger? In: 13th ACM International Conference on Ubiquitous Computing (UbiComp). (2011)

Inference of Switched Biochemical Reaction Networks Using Sparse Bayesian Learning

Wei Pan[1], Ye Yuan[2], Aivar Sootla[1], and Guy-Bart Stan[1]

[1]Centre for Synthetic Biology and Innovation and Department of Bioengineering, Imperial College London, {w.pan11, a.sootla, g.stan}@ imperial.ac.uk
[2]Department of Engineering, University of Cambridge, yy311@cam.ac.uk

Abstract. This paper proposes an algorithm to identify biochemical reaction networks with time-varying kinetics. We formulate the problem as a nonconvex optimisation problem casted in a sparse Bayesian learning framework. The nonconvex problem can be efficiently solved using Convex-Concave programming. We test the effectiveness of the method on a simulated example of DNA circuit realising a switched chaotic Lorenz oscillator.

Keywords: Switched system, Sparse Bayesian Learning, Convex-Concave programming.

1 Motivation, background and related work

Identification of switched systems, which are characterised by the interaction of both continuous and discrete dynamics, is widely used in many different fields such as systems/synthetic biology, econometrics, finance, biochemical engineering, social networks, etc. In this paper, we are interested in the identification of switched biochemical reaction networks. Biochemical processes can go through different phases in time; for example, a cell cycle in bacteria or diurnal alternations in plants. These switches are typically triggered by time dependent processes or by some external force. Therefore, the dynamics of biochemical reactions can be modelled as a collection of submodels amongst which switches occur over time. For biochemical reaction networks, the submodels are typically nonlinear due to mass action kinetics.

In classical switched system identification, the submodels are typically assumed to be linear or of the switch affine type [1], which is often used to approximate nonlinear dynamics. In [2], the structure of submodels is fixed and a minimal number of switches between submodels is inferred. However, these techniques are not generally applicable to biochemical kinetics due to highly nonlinear terms and model complexity. In the nonlinear case, there is an additional problem of nonlinear basis selection, which is fixed and predefined in the

linear case. Unlike the linear case, the number of nonlinear basis functions can be infinite and one might have to use complicated nonlinear functions to model the dynamics without any switches. In practice, if one is interested in obtaining the least number of switches, the number of nonlinear basis functions will typically grow, and vice versa, a small number of nonlinear basis functions will result in many switches. Hence there are two different and competing minimisation criteria: the number of switches between submodels and the number of basis functions in each submodel.

In this paper, we cast the problem of identification of switched biochemical reaction networks as a linear regression problem by defining a set of nonlinear basis functions based on mass action kinetics. Minimising the number of switches and/or the number of basis functions is typically addressed in such problems by an ℓ_1 or ℓ_2 regularisation approach. In this paper, however, we take a sparse Bayesian learning approach, which is shown to promote sparsity better than to ℓ_1 methods [3–5]. By specifying sparse priors on the number of parameters and the number of switches in this sparse Bayesian learning framework, the identification problem is formulated as a nonconvex optimisation problem. By exploiting the structure of the nonconvex optimisation problem, one can use Convex-Concave programming techniques to solve the problem efficiently. One illustrative example from DNA computation is used to show the effectiveness of the proposed method.

2 Preliminaries on biochemical reaction networks

Consider a biochemical system with n species $X_1, \ldots, X_n$. We denote the concentration of species X_j as x_j. Let $\mathcal{U}$ be the set of uni-species reactions and $\mathcal{B}$ be the set of bi-species reactions. A uni-species reaction $i \in \mathcal{U}$ is defined by the index $r_i \in \{1, \ldots, n\}$ of its single reactant species, the associated real-valued rate constant $k_i > 0$, and the integer product coefficients for each species $c_{i,j} \geq 0$: $m_i X_{r_i} \stackrel{k_i}{\rightarrow} c_{i,1}X_1 + \ldots + c_{i,n}X_n$. A bi-species reaction $i \in \mathcal{B}$ is defined by the indices $r_{i,1}, r_{i,2} \in \{1, \ldots, n\}$ of its two reactant species, the real-valued rate constant $k_i > 0$, and the integer product coefficients for each species $c_{i,j} \geq 0$: $m_i X_{r_{i,1}} + n_i X_{r_{i,2}} \stackrel{k_i}{\rightarrow} c_{i,1}X_1 + \ldots + c_{i,n}X_n$. Using the law of mass action, the dynamics of the concentrations $x_j \geq 0$ of species X_j are given according to the ordinary differential equations

$$\begin{aligned} \dot{x}_j = &- \sum_{i \in \mathcal{U} | r_i = j} k_i x_j^{m_i} - \sum_{i \in \mathcal{B} | r_{i,1} = j} k_i x_j^{m_i} x_{r_{i,2}}^{n_i} - \sum_{i \in \mathcal{B} | r_{i,2} = j} k_i x_{r_{i,1}}^{m_i} x_j^{n_i} \\ &+ \sum_{i \in \mathcal{U}} c_{i,j} k_i x_{r_i} + \sum_{i \in \mathcal{B}} c_{i,j} k_i x_{r_{i,1}} x_{r_{i,2}}, \end{aligned} \tag{2.1}$$

We can expand (2.1) for more than two species, though this can be rarely found in reality due to highly improbable simultaneous three-species molecular collision mechanisms.

Eq. (2.1) can be modelled using the general form: $\dot{\mathbf{x}} = \mathbf{S}\mathbf{v}(\mathbf{x})$, where $\mathbf{x}$ is the vector of species whose elements are x_j, $\mathbf{S}$ is the stoichiometry matrix and $\mathbf{v}(\mathbf{x})$ is a vector of propensity functions. The matrix $\mathbf{S}$ and the propensity vector $\mathbf{v}(\mathbf{x})$ can be built based on the biochemical reactions and their rates. Hence, without loss of generality we can assume that $\mathbf{S}$ is a matrix whose elements are real constants and $\mathbf{v}(\mathbf{x})$ is a vector whose elements are nonlinear functions of $\mathbf{x}$ as in (2.1). Biochemical processes can go through different phases in time; for example, a cell cycle in bacteria or diurnal alternations in plants. These switches, which are typically triggered by time dependent processes or by some external force, can be fitted into our model as follows: $\dot{\mathbf{x}} = \mathbf{S}^{\alpha(t)}\mathbf{v}(\mathbf{x})$, where $\alpha(t)$ is a sequence of integers in a bounded set and $\mathbf{S}^{\alpha(t)}$ takes values from an unknown set $\{\mathbf{S}_1, \ldots, \mathbf{S}_{N_{modes}}\}$ depending on time.

In what follows, we consider the system dynamics expressed in discrete-time and subjected to additive i.i.d. Gaussian noise $\xi(k)$ with known statistics.

$$\mathbf{x}(k+1) = \mathbf{S}^{\alpha(k)}\mathbf{v}(\mathbf{x}(k)) + \xi(k). \tag{2.2}$$

3 Problem formulation

3.1 Linear Regression Problem Formulation

Taking the transpose of both sides of (2.2) and considering the i^{th} state variable x_i of (2.2), we can obtain

$$\begin{aligned} x_i(k+1) &= v_i^\top(x(k)) \cdot \left(\mathbf{S}_{i,:}^{\alpha(k)}\right)^\top + \xi_i(k), \\ &= \big(f_{i1}(\mathbf{x}(k)) \ldots f_{iN}(\mathbf{x}(k)\big) \cdot \mathbf{w}_i(k) + \xi_i(k), \end{aligned} \tag{3.1}$$

where $\mathbf{S}_{i,:}^{\alpha(k)}$ represent the i^{th} row of $\mathbf{S}^{\alpha(k)}$; and f_{ij} represent the basis functions we use to reconstruct the model. The form of these functions can be any of those described in (2.1). In (3.1), $\mathbf{w}_i(k) = [w_{i1}(k), \ldots, w_{iN}(k)]^\top$, and the noise $\xi_i(k)$ is assumed to be i.i.d. Gaussian distributed: $\xi_i(k) \sim \mathcal{N}(0, \sigma_i^2)$, with $\mathbb{E}(\xi_i(p)) = 0$, $\mathbb{E}(\xi_i(p)\xi_i(q)) = \sigma_i^2\delta_{pq}$, with $\delta_{pq} = \begin{cases} 1, p = q \\ 0, p \neq q \end{cases}$. Now, let's assume that time-series measurements from a biochemical network are collected in a vector $\mathbf{y}_i$, where $\mathbf{y}_i = \big(x_i(2) \ldots x_i(M+1)\big)^\top$. We state the problem as identifying the system (2.2) based on these measurements. That is, our goal is to find all matrices $\mathbf{S}_1, \ldots, \mathbf{S}_{N_{modes}}$ and the switching sequence $\alpha(k)$ from the measurements stored in $\mathbf{y}_i$. Since the formulation in (3.1) is similar for all the state variables x_i, $i = 1, \ldots, N$, in what follows we drop the subscript i to ease the notation.

By defining the following block matrix and vectors

$$\begin{aligned}
\mathbf{A} &\triangleq \left[\begin{array}{ccc|c|ccc} f_1(\mathbf{x}(1)) & & & \cdot & & f_N(\mathbf{x}(1)) & \\ & \ddots & & \ddots & & \ddots & \\ & & f_1(\mathbf{x}(M)) & \cdot & & & f_N(\mathbf{x}(M)) \end{array}\right] \\
&= \left[A_1 \middle| \ldots \middle| A_N \right] \in \mathbb{R}^{M \times MN}, \\
\mathbf{w} &\triangleq \left[w_1(1), \ldots, w_1(M) \middle| \ldots \middle| w_N(1), \ldots, w_N(M) \right]^\top \\
&= \left[\mathbf{w}_1^\top \middle| \ldots \middle| \mathbf{w}_N^\top \right]^\top \in \mathbb{R}^{MN}, \\
\boldsymbol{\Xi} &\triangleq [\xi(1), \ldots, \xi(M)]^\top \in \mathbb{R}^M.
\end{aligned} \tag{3.2}$$

we can reformulate the linear regression equations in (3.1) as

$$\mathbf{y} = \mathbf{A}\mathbf{w} + \boldsymbol{\Xi}. \tag{3.3}$$

There are two issues that needs to be considered at this stage. First, each block $\mathbf{w}_i = [w_1(1), \ldots, w_1(M)]$ is associated only with certain basis function. The solution $\mathbf{w}$ to (3.3) is therefore typically going to be block sparse, which is mainly due to the potential introduction of non-relevant and/or non-independent basis functions in $\mathbf{A}$. Second, in the switched case, we have to penalise the number of switches from t_1 to t_M and/or the number of modes N_{modes}, which can be fixed in advance or set equal to M. Clearly such a problem has an infinite number of solutions, especially in the noisy setting. Therefore, we refine the problem statement to identify the system (2.2) with the least number of non-zero blocks in $\mathbf{w}$ and the least number of switches in the sequence $\alpha(k)$.

These are actually two different and competing criteria: if we want the least number of switches, the number of non-zero parameters in $\mathbf{w}$ will grow, and vice versa, a small number of non-zero parameters in $\mathbf{w}$ will result in many switches.

3.2 Minimising the Number of Switches

To limit the number of switches, we need to ensure that $S^{\alpha(k)}$ stays the same from time k to time $k+1$. Hence we need to add a condition maximising the sparsity of $S^{\alpha(k+1)} - S^{\alpha(k)}$ for all k. This leads to the following problem statement:

Problem 1. Given $\mathbf{y}$ and $\mathbf{A}$ and the block partitions formulated in (3.3), find a $\mathbf{w}$ that can explain the data with the minimal number of switches and the minimal number of non-zero blocks in $\mathbf{w}$.

If we index the vector $\mathbf{w}$ appropriately, the problem of minimising the number of switches can be formulated by enforcing $\mathbf{D}_j\mathbf{w}_j$ sparse, where the matrix $\mathbf{D}_j$ is defined as follows:

$$\mathbf{D}_j \triangleq \begin{bmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \end{bmatrix} \in \mathbb{R}^{(M-1)\times M}. \tag{3.4}$$

If we further define

$$\mathbf{B}_j \triangleq \begin{bmatrix} I \\ \rho D_j \end{bmatrix} \in \mathbb{R}^{(2M-1)\times M,}, \mathbf{B} \triangleq \begin{bmatrix} \mathbf{B}_1 & & \\ & \ddots & \\ & & \mathbf{B}_N \end{bmatrix} \in \mathbb{R}^{N(2M-1)\times MN}, \tag{3.5}$$

Problem 1 can be formulated as follows

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{w}\|_2^2 + \lambda\|\mathbf{B}\mathbf{w}\|_{\ell_0}, \tag{3.6}$$

where ρ in (3.5) is a trade-off parameter between the number of switches and the number of non-zero parameters, while λ in (3.6) is known as a regularisation parameter in penalised linear regression problems. Using the specially designed matrix $\mathbf{B}$ defined in (3.5), we can penalise a) the number of switches that occur and b) the number of non-zero element in every identified model.

For matrices $\mathbf{B}$ with the special form given in (3.5), algorithms minimising the number of non-zero elements and the number of switches belong to the class of so-called *fused LASSO algorithms* [6]. For general $\mathbf{B}$ matrices, the problem defined in (3.6) would be solved using *generalised LASSO algorithms* [7].

Overall, instead of employing LASSO-type algorithms to obtain an approximated solution, we are going to tackle the problem from a sparse Bayesian learning perspective [3, 4] as this gives much sparser solutions.

4 Sparse Bayesian Learning

In order to estimate $\mathcal{P}(\mathbf{w}|\mathbf{y})$, firstly the prior distribution over $\mathbf{w}$ should be specified. In problem (3.6), we not only want to minimise the number of basis functions but also the number of switches. Therefore, sparsity promoting priors should be specified for $\mathcal{P}(\mathbf{B}_{j,:}\mathbf{w}_j)$, $\forall j$, where $\mathbf{B}_{j,:}$ is the j^{th} row of $\mathbf{B}$. These priors can be chosen as *super-Gaussian* [8]. It means that for every parameter $\mathbf{B}_{j,:}\mathbf{w}_j$, we define a hyper-parameter γ_j such that $\mathcal{P}(\mathbf{B}_{j,:}\mathbf{w}_j) = \max_{\gamma_j>0} \mathcal{N}(\mathbf{B}_{j,:}\mathbf{w}_j|0,\gamma_j)\varphi(\gamma_j)$. In this case the priors $\mathcal{P}(\mathbf{B}\mathbf{w})$ can be computed as follows:

$$\mathcal{P}(\mathbf{B}\mathbf{w}) = \max_{\boldsymbol{\gamma}>\mathbf{0}} \prod_j \mathcal{N}(\mathbf{B}_{j,:}\mathbf{w}_j|0,\gamma_j)\varphi(\gamma_j). \tag{4.1}$$

where $\boldsymbol{\gamma}$ is a vector of γ_j and $\varphi(\cdot)$ is a nonnegative function of the hyperparameters, which can be given depending on a selection specific sparsity promoting distribution, such as a Laplace distribution, a Student's t distribution, etc. Note that, if the parameter vector $\boldsymbol{\gamma}$ is known, we can estimate $\mathcal{P}(\mathbf{B}\mathbf{w}|\mathbf{y};\boldsymbol{\gamma})$ instead of computing $\mathcal{P}(\mathbf{B}\mathbf{w}|\mathbf{y})$. Therefore, the problem should be recasted in terms of finding the most appropriate hyperparameters of the priors: $\hat{\boldsymbol{\gamma}}$. A good way of selecting $\hat{\boldsymbol{\gamma}}$ is to choose it as the minimiser of the sum of the misaligned probability

mass, e.g.,

$$\begin{aligned}\hat{\boldsymbol{\gamma}} &= \underset{\boldsymbol{\gamma}>\mathbf{0}}{\operatorname{argmin}} \int \mathcal{P}(\mathbf{y}|\mathbf{w})\,|\mathcal{P}(\mathbf{B}\mathbf{w}) - \mathcal{P}(\mathbf{w};\boldsymbol{\gamma})|\,d\mathbf{w} \\ &= \underset{\boldsymbol{\gamma}>\mathbf{0}}{\operatorname{argmax}} \int \mathcal{P}(\mathbf{y}|\mathbf{w})\mathcal{P}(\mathbf{B}\mathbf{w};\boldsymbol{\gamma})d\mathbf{w}.\end{aligned} \tag{4.2}$$

The procedure in (4.2) is referred to as evidence/marginal likelihood maximisation [3, 4]. It means that the marginal likelihood can be maximised by selecting the most probable hyperparameters able to explain the observed data. Defining $\boldsymbol{\Gamma}$ as a diagonal matrix with diagonal entries γ_j, the parameters $\mathbf{w}$ and $\boldsymbol{\gamma}$ can be estimated by solving the optimisation problem in Proposition 1:

Proposition 1. *The optimisation problem in* (4.2) *is equivalent to the following non-convex problem*

$$\begin{aligned}\min_{\boldsymbol{\gamma}>\mathbf{0},\mathbf{w}} \{ &\frac{1}{\sigma^2}\|\mathbf{y} - \mathbf{A}\mathbf{w}\|_2^2 + \mathbf{w}^\top \mathbf{B}^\top \boldsymbol{\Gamma}^{-1}\mathbf{B}\mathbf{w} \\ &+ \log|\boldsymbol{\Gamma}| + \log|\mathbf{B}^\top\boldsymbol{\Gamma}^{-1}\mathbf{B} + \sigma^{-2}\mathbf{A}^\top\mathbf{A}| + \sum_{j=1}^{N} p(\gamma_j)\}\end{aligned} \tag{4.3}$$

where $\boldsymbol{\Gamma}$ *is a diagonal matrix with entries* $\boldsymbol{\gamma}$ *on the diagonal and* $p(\cdot) = \log(\varphi(\cdot))$.

Proof. The proof is similar to that derived in [4, 5]. Therefore, we omit it due the space limitation.

We approach the solution to this problem by separating the objective function into the following parts:

$$\begin{aligned}f(\mathbf{w},\boldsymbol{\gamma}) &= \frac{1}{\sigma^2}\|\mathbf{y} - \mathbf{A}\mathbf{w}\|_2^2 + \mathbf{w}^\top \mathbf{B}^\top \boldsymbol{\Gamma}^{-1}\mathbf{B}\mathbf{w} \\ g(\boldsymbol{\gamma}) &= \log|\boldsymbol{\Gamma}| + \log|\mathbf{B}^\top\boldsymbol{\Gamma}^{-1}\mathbf{B} + \sigma^{-2}\mathbf{A}^\top\mathbf{A}| + \sum_{j=1}^{N} p(\gamma_j).\end{aligned}$$

Proposition 2. *The function* $f(\mathbf{w},\boldsymbol{\gamma})$ *is jointly convex in* $\mathbf{w}$ *and* $\boldsymbol{\gamma}$*, while the function* $g(\boldsymbol{\gamma})$ *is concave.*

Proof. It is easy to verify the first part of this proposition. A proof on concavity of the sum of log-determinant functions in the second part for general matrices $\mathbf{B}$ can be found in [5, Theorem 3.1 (3)].

Proposition 2 allows us to use Convex-Concave Programming [9] in order to find a stationary point, which results in:

$$\left(\mathbf{w}^{k+1}, \boldsymbol{\gamma}^{k+1}\right) = \underset{\mathbf{w},\boldsymbol{\gamma}>\mathbf{0}}{\operatorname{argmin}} f(\mathbf{w},\boldsymbol{\gamma}) + \nabla_{\boldsymbol{\gamma}}(g(\boldsymbol{\gamma}^k))^\top\boldsymbol{\gamma} \tag{4.4}$$

In order to make the algorithm more transparent we also separate the minimisation into separate minimisation programmes over $\mathbf{w}$ and $\boldsymbol{\gamma}$: By defining $\boldsymbol{\epsilon}^{k+1} \triangleq \nabla_{\boldsymbol{\gamma}}(g(\boldsymbol{\gamma}^k))$, the optimal solution in (4.4) over $\boldsymbol{\gamma}$ can be computed analytically as $\gamma_j = \mathbf{B}_{j,:}\mathbf{w}/\sqrt{\epsilon_j}$, $\forall j$ and for every fixed $\mathbf{w}$. Now we only need to minimise in (4.4) over $\mathbf{w}$ as follows:

$$\mathbf{w}^{k+1} = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{A}\mathbf{w} - \mathbf{y}\|_2^2 + \sigma^2 \sum_j \|\epsilon_j^k \cdot \mathbf{B}_{j,:}\mathbf{w}\|_1,$$

while the hyperparameters are updated as $\gamma_j^{k+1} = \mathbf{B}_{j,:}\mathbf{w}^{k+1}/\sqrt{\epsilon_j}$, $\forall j$. To summarise the algorithm, one can initialise γ_j^0 at any positive real scalar. Some additional insight can be obtained by initialising $\epsilon_j^0 = 1$, $\forall j$ instead. In that case, the first iteration becomes a linear regression problem with ℓ_1 penalty on the parameters $\mathbf{B}\mathbf{w}$:

$$\mathbf{w}^1 = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{A}\mathbf{w} - \mathbf{y}\|_2^2 + \sigma^2 \|\mathbf{B}\mathbf{w}\|_{\ell_1}.$$

Then we update γ_j^1 using $\gamma_j^1 = \mathbf{B}_{j,:}\mathbf{w}^1/\sqrt{\epsilon_j^0}$. Using this initialisation, we provably get results at least not worse than the generalised LASSO algorithm. Algorithm 1 summarises this approach, which converges to a stationary point in $\mathbf{w}$ and $\boldsymbol{\gamma}$ [9]. Algorithm 1 can be seen as a particular version of the reweighted LASSO approach with a Bayesian update on the weights. The program (4.4) is

Algorithm 1 Switched Systems Identification Algorithm

1: Initialise $\epsilon_j^0 = 1,\ \forall j = 1, \ldots, N(2M-1)$,
2: **for** $k = 0, \ldots, k_{\max}$ **do**
3: Update the parameters as follows:

$$\mathbf{w}^{k+1} = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{A}\mathbf{w} - \mathbf{y}\|_2^2 + \sigma^2 \sum_j \|\epsilon_j^k \cdot \mathbf{B}_{j,:}\mathbf{w}\|_1$$

$$\boldsymbol{\gamma}_j^{k+1} = \frac{\mathbf{B}_{j,:}\mathbf{w}^{k+1}}{\sqrt{\epsilon_j^k}}$$

$$\epsilon_j^{k+1} = -\frac{\mathbf{B}_{j,:}(\mathbf{B}^\top(\boldsymbol{\Gamma}^{k+1})^{-1}\mathbf{B} + \rho^k \mathbf{A}^\top\mathbf{A})^{-1}\mathbf{B}_{j,:}^\top}{(\gamma_j^{k+1})^2} + \frac{1}{\gamma_j^{k+1}}$$

4: **if** a stopping criterion is satisfied **then**
5: Break
6: **end if**
7: **end for**

convex, quadratic and uncostrained; however, the size of the problem can be extremely large. Two techniques to speed-up the solution can be adopted: pruning

the parameter $\mathbf{w}$ space after each iteration as in [10], and/or using distributed computation methods such as ADMM, e.g. [11]).

5 Results

In this section, we consider time-series data obtained from a chaotic Lorenz Oscillator implemented *in vitro* using DNA computations [12]. From the associated biochemical reactions, a polynomial ODE can be derived using the law of mass action. We artificially generate data using this oscillator model but change certain parameters at certain time. This can be realised *in vitro* by changing experiment conditions or enzyme concentrations. The Lorenz oscillator can be described by the discretised differential equations

$$\begin{aligned}&\left[\frac{y_1(k+1)-y_1(k)}{\delta t}, \frac{y_2(k+1)-y_2(k)}{\delta t}, \frac{y_3(k+1)-y_3(k)}{\delta t}\right]\\ =&\left[p_1(k)(y_2(k)-y_1(k)), y_1(k)(p_2(k)-y_2(k)), y_1(k)y_2(k)-k_2(k)y_3(k)\right].\end{aligned}$$

where we the fix sampling time to $\delta t = 0.02$ (arbitrary units).

Initially ("Mode 1"), the parameters are $p_1 = 10, p_2 = 30, p_3 = 8/3$. From $k = 201$ to $k = 400$ ("Mode 2"), the parameters are changed to $p_1 = 10, p_2 = 30, p_3 = 4$. For the kinetics of y_1 and y_3, the nonlinear dynamics change after switching from Mode 1 to Mode 2. For y_2, the parameters do not switch. We construct the basis functions in (3.1) as

$$\left(y_1^0(k)y_2^0(k)y_3^0(k),\, y_1^0(k)y_2^0(k)y_3^1(k),\, \ldots,\, y_1^{n_1}y_2^{n_2}(k)y_3^{n_3}(k)\right).$$

We index the parameter vector $\mathbf{w}(k)$ as $[w^{000}(k), w^{001}(k), \ldots, w^{n_1n_2n_3}(k)]$, choose $\lambda = 1$ and $\rho = 100$ and set the initial condition to $[y_1(1), y_2(1), y_3(1)] = [0.2444, -2.217, 2.314]$. Finally, we set $n_1 = 1$, $n_2 = 1$ and $n_3 = 1$. The true and estimated parameters' evolution over time are shown in Figure1.

6 Conclusion

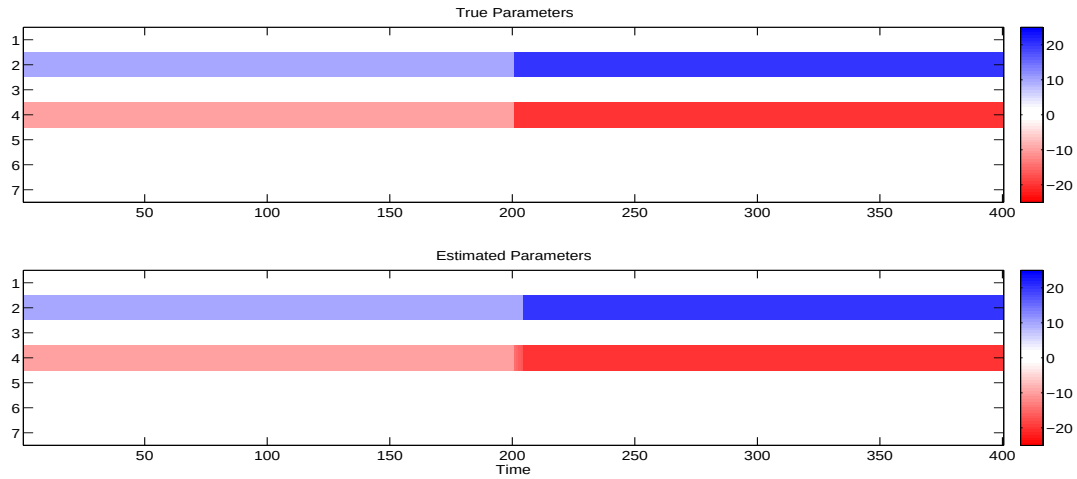
In this paper we proposed an efficient way to solve the switched system identification problem for biochemical reaction networks. For this purpose, an efficient framework based on sparse Bayesian learning has been proposed to solve this problem by specifying sparse priors on the number of parameters and the number of switches. Future work lies in the identifiability of such switching systems and how to design proper excitation signals to guarantee identifiability of the switching systems from output data.

7 Acknowledgement

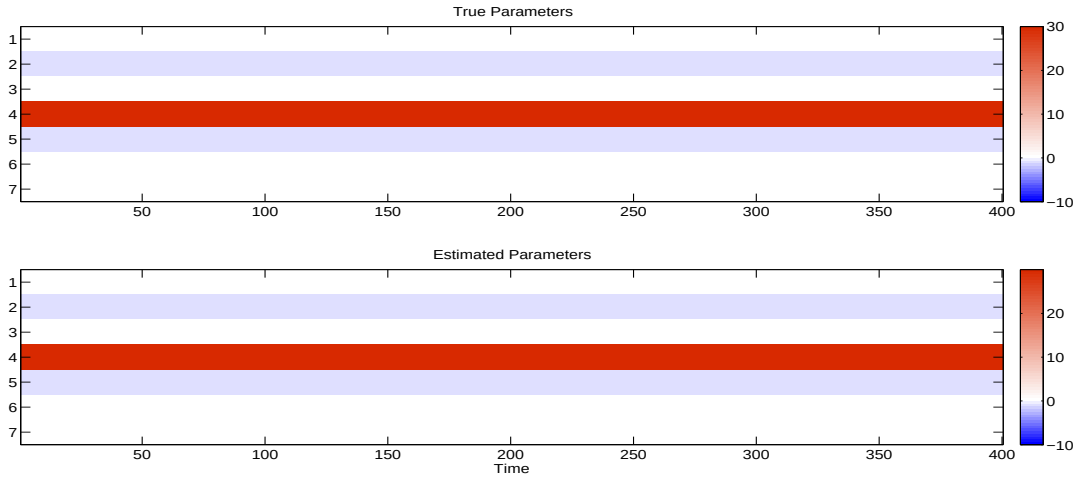
Mr W. Pan gratefully acknowledges the support of Microsoft Research through the PhD Scholarship Program. Dr A. Sootla and Dr G.-B. Stan acknowledge the support of EPSRC through the project EP/J014214/1 and the EPSRC Science and Innovation Award EP/G036004/1. Dr Y. Yuan acknowledge the support from EPSRC (project EP/I03210X/1).

References

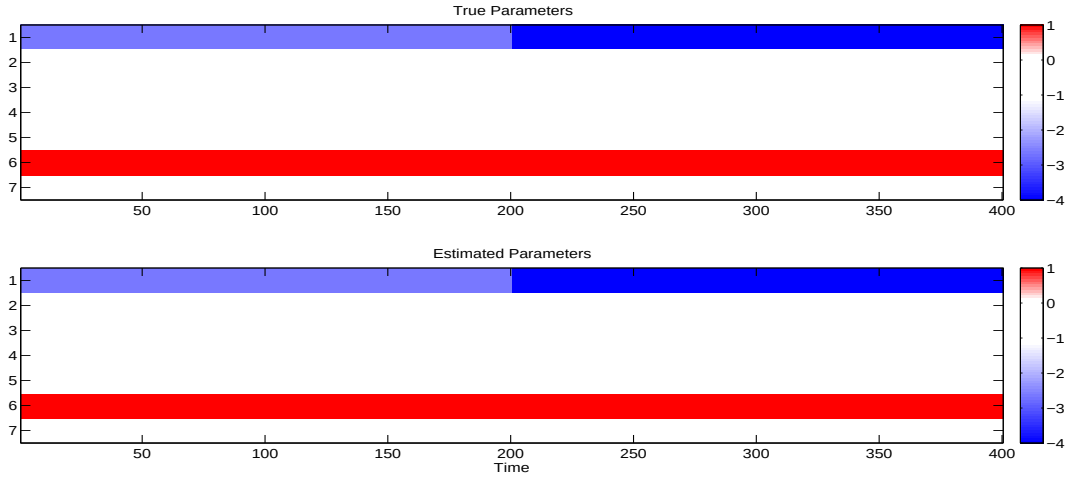
1. S. Paoletti, A. L. Juloski, G. Ferrari-Trecate, and R. Vidal, "Identification of hybrid systems a tutorial," *European journal of control*, vol. 13, no. 2, pp. 242–260, 2007.
2. N. Ozay, M. Sznaier, C. M. Lagoa, and O. I. Camps, "A sparsification approach to set membership identification of switched affine systems," *Automatic Control, IEEE Transactions on*, vol. 57, no. 3, pp. 634–648, 2012.
3. M. Tipping, "Sparse bayesian learning and the relevance vector machine," *The Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001.
4. D. Wipf, B. Rao, and S. Nagarajan, "Latent variable bayesian models for promoting sparsity," *Information Theory, IEEE Transactions on*, vol. 57, no. 9, pp. 6236–6255, 2011.
5. M. W. Seeger and H. Nickisch, "Large scale bayesian inference and experimental design for sparse linear models," *SIAM Journal on Imaging Sciences*, vol. 4, no. 1, pp. 166–199, 2011.
6. R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, "Sparsity and smoothness via the fused lasso," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 1, pp. 91–108, 2005.
7. R. J. Tibshirani, *The solution path of the generalized lasso*. Stanford University, 2011.
8. J. Palmer, K. Kreutz-delgado, B. D. Rao, and D. P. Wipf, "Variational em algorithms for non-gaussian latent variable models," in *Advances in neural information processing systems*, 2005, pp. 1059–1066.
9. B. K. Sriperumbudur and G. R. Lanckriet, "On the convergence of the concave-convex procedure." in *NIPS*, vol. 9, 2009, pp. 1759–1767.
10. W. Pan, Y. Yuan, J. Gonçalves, and G.-B. Stan, "Bayesian approaches to nonlinear network reconstruction," *submitted*, 2013.
11. S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.
12. D. Soloveichik, G. Seelig, and E. Winfree, "Dna as a universal substrate for chemical kinetics," *Proceedings of the National Academy of Sciences*, vol. 107, no. 12, pp. 5393–5398, 2010.



(a) True and estimated parameters for y_1.



(b) True and estimated parameters for y_2.



(c) True and estimated parameters for y_3.

Fig. 1: True (upper panel) and estimated (lower panel) parameters' evolution over time. The horizontal axis represents time, whereas the vertical axis represents the estimated coefficients. From top to bottom, the index goes from 001 to 111.

Heterogeneous Bayes Filters with Sparse Bayesian Models: Application to state estimation in robotics

Alexandre Ravet[1,2] and Simon Lacroix[1,3]

[1] CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France
[2] Univ de Toulouse, INSA, LAAS, F-31400 Toulouse, France
[3] Univ de Toulouse, LAAS, F-31400 Toulouse, France

Abstract. This study introduces a new augmented Bayes filter model for time-varying, context-dependent emission noise. The envisaged application, robust state estimation for a robot, motivates the use of the Relevance Vector Machine to model the emission noise, because it provides sparsity and fast inference capabilities. Besides the introduction of this new model, this work also aims at comparing the final filter performance when discriminative training is used instead of the prevalent generative training. The theoretical foundations for training and running inference over the model are proposed.

1 Introduction

Bayes filters (BF) have been widely applied to many areas. They are notably a workhorse of robotics, where recursive filtering [6], a fast and simple inference procedure, has provided the most common and reliable method for real-time state estimation over the past decades. BF used for state estimation usually rely on some optimistic assumptions for two main reasons: the real physical system is actually too complex to be perfectly described through a tractable model, and physical exactitude is neglected for the sake of computational efficiency. As a consequence, filter models designed for state estimation usually rely on a minimum system state representing the robot variables required for achieving a specific task. Any unmodeled aspect of the system, among which the effects of the current environment over the filter performance, are then encompassed within additional noise terms.

In practice, when used for robot state estimation, Bayes filters require a substantial tuning phase to provide acceptable performances. This is because capturing all unspecified aspects of the system through the sole introduction of noise often results in a trade-off between output optimality (accuracy of the state estimate) and robustness (to the different unmodeled aspects). An illustration of this problem, and the core motivation of this work, is the case of an autonomous robot navigating through different environments, in which one has to deal with a whole range of alterations in sensor readings, going from average (optimal) observation noise to complete failure (unreliable data). This is especially true

when sensor performances are strongly affected by the different environment characteristics, such as luminosity, texture and materials of surroundings objects, ground and obstacles...

The classical state-space model is defined by:

$$\begin{aligned} x_t &= f(x_{t-1}) + \gamma \\ y_t &= g(x_t) + \nu \end{aligned} \tag{1}$$

where x_t is the latent state at time t with the associated observation y_t, f and g the transition and emission functions respectively, $\gamma \sim \mathcal{N}(0, \Sigma_\gamma)$ the system noise and $\nu \sim \mathcal{N}(0, \Sigma_\nu)$ the observation noise. f and g can either be linear functions (linear dynamical system) or nonlinear (nonlinear dynamical system). Most often, no fixed noise values Σ_ν that would yield an optimal output for a large variety of operating conditions (or environments) can be determined. Other unmodeled effects producing the strongest measurement alterations are often compensated with a rejection scheme, usually relying on a model self-consistency check. In other words, designing a Bayes filter emission model consists in finding the best distribution modeling the emission process for the nominal cases, and reject all the data that does not fit this distribution [13]. One strong consequence of this approach is that the system might converge to erroneous but model-consistent state values [14, 12]. When such divergences are observed, additional parameter tuning is required to improve global robustness, then detrimentally affecting the state estimation performance.

This classical state-space model, whose graphical representation is shown in Fig.1, is known as (time) homogeneous. It can be enhanced by making the model parameters vary in time: the trade-off usually required when tuning the parameters is then no longer needed, and the resulting model can handle the whole spectrum of alterations over measurements. As the research context motivating this study concerns robust and adaptive perception for autonomous robots, this work focuses on the emission distribution – even though the proposed approach can be straightforwardly applied to the prediction distribution of model (1).

This paper aims at developing a model able to compensate for the assumptions made by describing a system through the simplified emission distribution with simple Gaussian noise Σ_ν – while maintaining high computational efficiency. It results in an enhanced Kalman filter capable of dealing with both moderate alterations and outliers, without requiring the implementation of rejection rules. This is done by training an additional model for the emission noise, which relies on contextual information input. One particularly appealing consequence of this approach lies in the introduction of a second order knowledge over measurements reliability, where basic rejection schemes rely on some knowledge about the data properties. The proposed approach is consequently less likely to diverge.

Discriminative training being known to help in compensating some of the mis-modeled aspects of a system [1], we also aim at analyzing the impact of a discriminative learning method versus a generative one over the performance of the resulting filter.

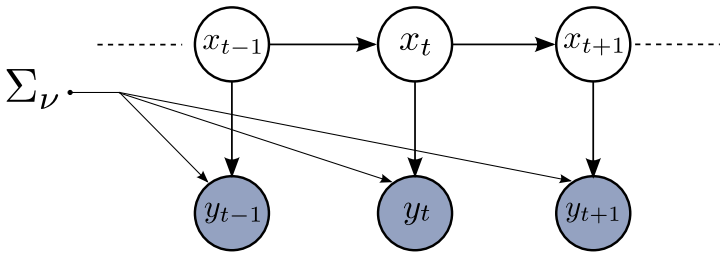


Fig. 1. Graphical model of a Bayes filter with homogeneous emission distributions.

The next section depicts the model basic principle. Sections 3 and 4 respectively describe generative and discriminative training of the model. Inference methods are then provided in section 5, and a discussion concludes the paper.

2 Heterogeneous BF with sparse Bayesian models

2.1 Background

Unless the state of simple models such as (1) encompasses all the exogenous phenomena likely to alter the system behavior, BF are by nature unable to model time varying emission and prediction processes. Recently, extensions have been proposed in order to compensate this unability. Nonparametric models such as Gaussian Processes (GP) have been integrated for modeling transition and emission distributions [3], and extended to fully state-dependent models in [9], through the introduction of a heteroscedastic observation noise. If nonparametric models improve the filter robustness compared to parametric functions, they are usually designed as state-dependent models, and as such are unable to handle contextual influence over the measurement process. For the heteroscedastic observation model proposed in [9], the presence of outliers in the training set is then critical: based on the sole state information, the system is unable to discern the contribution of the noise free model ($g(x_t)$ in (1)) from the noise model (which is then written $\nu(x_t)$) within measurements. An other disadvantage of these approaches is that one has to turn to more complex sparse GP techniques when using a large training set if the system is intended to be used in real time.

From a different perspective, optimizing the parameters of a BF has always been mostly considered as a tuning task, achieved with the intuitive goal of providing the most accurate state estimation. This allows to take into consideration some mis-modeled aspects of the system, even if they are never explicitly described, neither understood. Surprisingly, methods for learning the parameters of a BF appeared quite recently in the literature, and mostly rely on maximum likelihood. Since BF are generative models, this implies that the parameters are not determined with respect to the system ultimate performance, but so as to get the best model for the underlying prediction and emission processes. Conversely, discriminative training is similar to manual tuning, in the sense that

the parameters are optimized with respect to filter performance. But this latter training approach remains uncommon, although it proved to outperform manual tuning and maximum likelihood as well [1]. To our knowledge, combining an augmented model with discriminative training remains untreated, while both approaches serve a similar purpose, *i.e* compensating for unmodeled aspects of the real system.

2.2 Heterogeneous Bayes filters with sparse Bayesian model

To overcome BF unability to deal with context influence, an augmented model is introduced, whose particularity is to explicit the context repercussion over the measurement process. It relies on an additional observation variable c_t relating to the perception context. As suggested in previous work [12, 11], this additional observation can consist in the joint set – or subset – of sensor measurement values y_t possibly extended with any relevant contextual information i_t (any other sensor measurement, robot internal data, or any information that might influence the measurement emission process). It is assumed that this joint set of measurements defines a proper representation space for the contextual influence over measurement noise, *i.e* there exists a mapping from the context input space to the observation noise level.

To further avoid ambiguities in the contribution of two distinct models g and ν in the measurement process, we assume that the noise free component g of the emission model is known and homogeneous in time, since it can generally be obtained directly through physical considerations about the nominal measurement generation process. This results in an emission model of the form:

$$y_t = g(x_t) + \nu(c_t)$$

Reminding the goal of this model is to enhance a Kalman filter, $\nu(c_t)$ is then a zero mean Gaussian noise distribution with context-dependent variance:

$$\nu(c_t) \sim \mathcal{N}\big(0, r(c_t)\big)$$

To avoid making any assumption over the functional form for the variance model, and keep computational efficiency, $r(c_t)$ is modeled with the Relevance Vector Machine (RVM) framework [15], naturally providing sparsity thanks to the *automatic relevance determination* mechanism. For a given training set of T observations $\{c_i\}_{i=1}^{T}$, we define $r(c_t) = exp(z_t)$ to ensure variance positivity where z_t is given by

$$z_t = \sum_{i=1}^{T} w_i K(c_t, c_i) + w_0 + \epsilon \tag{2}$$

with w_0 a bias parameter, K the chosen kernel function, and $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$. To foster sparsification, a zero-mean Gaussian prior is placed over the weight vector $\mathbf{w} = (w_0, w_1, .., w_N)^T$:

$$p(\mathbf{w}|\alpha) = \prod_{i=0}^{N} \mathcal{N}(w_i|0, \alpha_i^{-1})$$

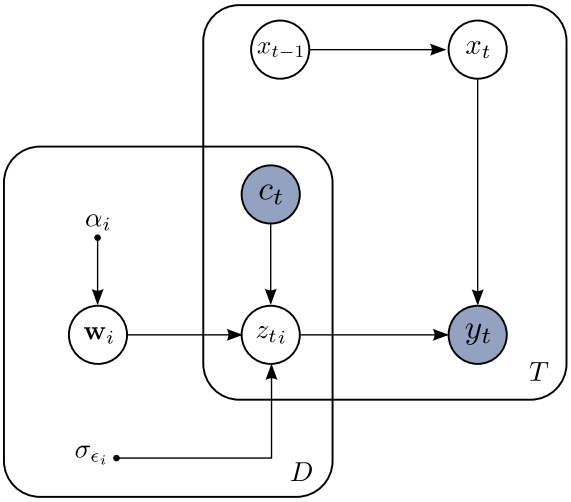


Fig. 2. Bayes Filter with heterogeneous emission noise.

where we define uniform hyperparameters priors over $\alpha = (\alpha_0, \alpha_1, ..., \alpha_N)^T$.

So far, the model has been depicted for a one-dimensional observation space. Real applications however require to consider the multi-dimensional case. For an observation variable $y_t \in \mathbb{R}^D$, D distinct RVM models are then used to model each component of the noise covariance matrix $\nu(c_t)$. For clarity, the next sections only consider the one-dimensional case, the extension to higher dimensions being straightforward. The resulting graphical model of this augmented model is depicted Fig.2.

3 Generative training

Bayes filter parameter optimization is usually done by minimizing the likelihood of the training set [4, 2], considering the latent state variable remains unobserved. In this work, we assume that the training set also contains ground truth data, *i.e* accurate values of the state variables $\mathbf{x} = \{x_1, ..., x_T\}$. The issue of training the emission model then turns to be analogous to the regression task, and more specifically to the heteroscedastic regression task with nonparametric models [5, 7, 10, 8, 9]. Note however that the training task is here a bit simpler since the observation function g is fixed and only the model $\nu(c_t)$ has to be determined. If sampling and variational approximation can be also used, the chosen approach relies on hard-assignment Expectation Maximization (EM) as suggested in [7]. By using hard-assignment EM we iteratively estimate the RVM parameters and predicted log noise level $\mathbf{z}$ at original inputs $C = \{c_i\}_{i=1}^T$. Thanks to this approximation, we are able to make direct use of classical RVM optimization and prediction equations, providing in this context the fastest solution for a real time application.

Following Kersting et al. approach [7], the hard E-step consists in empirical estimation of the noise variance. Based on real observations $\mathbf{y} = \{y_1, ..., y_T\}$ and

samples y_t^k provided by the current observation model (using the parameters α and σ_ϵ found after last EM iteration), the set of values y_t and $\{y_t^k\}_{k=1}^K$ are seen as independent noisy observations of $g(x_t)$. Empirical estimation of the noise variance at x_t is then provided by the mean

$$var_t = \frac{1}{2.K} \sum_{k=1}^{K} (y_t - y_t^k)^2$$

In the subsequent M-step the RVM model is trained with the new training set $D = \{c_t, log(var_t)\}_{t=1}^T$ with a classical optimization procedure [15].

In other words, the optimization process considers the noise variance as the hidden variable of the model, and iteratively optimize the parameters of the RVM model based on a hard assignment of the estimated noise. This method requires to use a substantial number of samples to empirically estimate the noise variance and, as any hard-assignment EM, is prone to oscillating, requiring to monitor the likelihood of the model over the training set after each algorithm iteration. It however brings an important advantage, since the optimized model, in association with the last noise variance estimation, can be readily used for prediction using classical RVM equations.

4 Discriminative training

The previous learning approach aims at minimizing a loss function corresponding to the emission likelihood. In other words the optimization step finds model parameters explaining at best the measurement generation process. As suggested in [1], it is however better to optimize the parameters with respect to the ultimate system performance, *i.e* the accuracy of filter estimates. Training the model then consists in finding α_{max} and $\sigma_{\epsilon max}$ such that:

$$\langle \alpha_{max}, \sigma_{\epsilon max} \rangle = \arg\max_{\alpha, \sigma_\epsilon} \sum_{t=1}^{T} log(p(x_t|y_{1:t}))$$

where $p(x_t|y_{1:t})$ is provided by Kalman equations. Considering f and g are linear functions corresponding to the matrices F and G respectively (classical Kalman filter case), we have:

$$p(x_t|y_{1:t}) = \mathcal{N}(x_t|\mu_t, \sigma_t)$$

with

$$\begin{aligned} \mu_t &= F\mu_{t-1} + K_t(y_t - GF\mu_{t-1}) \\ \sigma_t &= (I - K_t G)P_{t-1} \end{aligned}$$

$$\begin{aligned} P_{t-1} &= F\sigma_{t-1}F^t + \Sigma_\gamma \\ K_t &= P_{t-1}G^t(GP_{t-1}G^t + \Sigma_{\nu(c_t)}) \end{aligned}$$

Since this distribution requires the evaluation of two latent variables: the log noise level and the RVM weight parameter, a procedure similar to Type II maximum likelihood is employed. z_t and $\mathbf{w}$ are then marginalized out and we now seek for parameters α_{max} and $\sigma_{\epsilon max}$ maximizing:

$$\mathcal{L}_{discr} = \sum_{t=1}^{T} log\Big(\int \int p(x_t|y_{1:t}, z_t) p(z_t|c_t, \mathbf{w}, \sigma_\epsilon) p(\mathbf{w}|\alpha) \, \mathrm{d}\mathbf{w} \, \mathrm{d}z_t \Big)$$

Since $p(z_t|c_t, \mathbf{w}, \sigma_\epsilon)$ and $p(\mathbf{w}|\alpha)$ are both Gaussian, the integral with respect to $\mathbf{w}$ is readily evaluated to give:

$$\mathcal{L}_{discr} = \sum_{t=1}^{T} log\Big(\int p(x_t|y_{1:t}, z_t) \mathcal{N}(z_t|0, D_t) \mathrm{d}z_t \Big)$$

where $D_t = \sigma_\epsilon + K^T \alpha^{-1} K$, with K the vector of kernel functions such that $K_i = K(c_t, c_i)$ as defined in (2), and $A = \mathrm{diag}(\alpha)$. The last equation is analytically intractable and is then approximated with integration by substitution and Gauss-Hermite quadrature.

α_{max} and $\sigma_{\epsilon max}$ are subsequently found by conjugate gradient ascent over $\mathcal{L}_{discr}$. Note that classical optimization of the RVM model requires the computation of a *design matrix* containing all kernel elements evaluated at all original locations $\{c_i\}_{i=1}^T$. Here, the optimization is done separately for each kernel vector evaluated at c_i, through their influence over ultimate filter performance. This different form of training (by comparison to the one in [7]) then requires to foster sparsity by thresholding of the α_i values during the optimization process.

5 Inference

5.1 Generative training case

For the classical model of Fig.1, filtering consists in evaluating normalized marginal distributions $\hat{\alpha}(x_t) = p(x_t|y_1, .., y_t)$ with the recursion equation of the form:

$$\eta_t \hat{\alpha}(x_t) = p(y_t|x_t) \int \hat{\alpha}(x_{t-1}) p(x_t|x_{t-1}) \mathrm{d}x_{t-1} \tag{3}$$

where $\eta_t = p(y_t|y_1, ..., y_{t-1})$ the scaling factor and $p(y_t|x_t)$ and $p(x_t|x_{t-1})$ the emission and prediction distribution considered as Gaussian for kalman filtering. Since in the new model the noise level is considered as an additional latent variable, the emission distribution required for the evaluation of (3) is now given by:

$$p(y_t|x_t, c_t) = \int \mathcal{N}\big(y_t|g(x_t), exp(z_t)\big) p(z_t|c_t, C, \mathbf{z}, \alpha, \sigma_\epsilon) \mathrm{d}z_t \tag{4}$$

where $\mathbf{z}$ is the predicted log noise level at original inputs $\{c_i\}_{i=1}^T$, and $p(z_t|c_t, C, \mathbf{z}, \alpha, \sigma_\epsilon)$ the predictive distribution given by:

$$p(z_t|c_t, C, \mathbf{z}, \alpha, \sigma_\epsilon) = \int p(z_t|c_t, \mathbf{w}, \sigma_\epsilon) p(\mathbf{w}|C, \mathbf{z}, \alpha, \sigma_\epsilon) \mathrm{d}\mathbf{w} \tag{5}$$

This familiar predictive distribution [15] is also Gaussian. The evaluation of the integral (4) is hence analytically intractable, and requires approximation. The fastest approach is the most likely approximation, where we replace the integral by $\mathcal{N}\big(y_t|g(x_t), exp(z_t^*)\big)$ with $z_t^* = \arg\max_{z_t} p(z_t|c_t, C, \mathbf{z}, \alpha, \sigma_\epsilon)$. Note that this approximation allows $p(y_t|x_t, c_t)$ to be a Gaussian distribution, a necessary condition for using Kalman recursive equations.

5.2 Discriminative training case

Since discriminative training did not involve evaluation of the posterior distribution of the noise level $\mathbf{z}$, we can not make straightforward use of the RVM prediction equation. We then replace equation (5) by:

$$p(z_t|c_t, C, \alpha, \sigma_\epsilon) = \int\int p(z_t|c_t, \mathbf{w}, \sigma_\epsilon) p(\mathbf{w}|C, \mathbf{z}, \alpha, \sigma_\epsilon) p(\mathbf{z}|C, \alpha, \sigma_\epsilon) \mathrm{d}\mathbf{w}\mathrm{d}\mathbf{z} \quad (6)$$

We turn to MCMC sampling in order to evaluate at first the posterior $p(\mathbf{w}|C, \alpha, \sigma_\epsilon)$ reminding that

$$p(\mathbf{w}|C, \alpha, \sigma_\epsilon) = \int p(\mathbf{w}|C, \mathbf{z}, \alpha, \sigma_\epsilon) p(\mathbf{z}|C, \alpha, \sigma_\epsilon) \mathrm{d}\mathbf{z}$$

and as described in [15],

$$p(\mathbf{w}|C, \mathbf{z}, \alpha, \sigma_\epsilon) = \mathcal{N}(\mathbf{w}|m, \Sigma)$$
$$p(\mathbf{z}|C, \alpha, \sigma_\epsilon) = \mathcal{N}(\mathbf{z}|0, E)$$

where $m = \sigma_\epsilon \Sigma \Phi^T \mathbf{z}$, $\Sigma = (A + \sigma_\epsilon \Phi^T \Phi)^{-1}$ and $E = \sigma_\epsilon^{-1} I + \Phi A^{-1} \Phi^T$, with Φ the *design matrix*. For computational efficiency, evaluation of the predictive distribution over the noise level for a new input c_t is then done by replacing the integral over $\mathbf{w}$ in (6) by the evaluation of $p(z_t|c_t, \mathbf{w}, \sigma_\epsilon)$ at the point estimate value of $\mathbf{w}$ provided by the sampling procedure, which is done offline. The remaining of the inference is then similar to the one depicted in the generative training case.

6 Discussion

The aim of this work is to define a new Bayes filter model able to encompass a variety of contexts, and to analyse different training approaches and their expected consequences over the system performance. Its specificities rely in the introduction of an additional observation used for context identification, and in the use of a sparse RVM model for context-dependent observation noise prediction. The theoretical foundations have been presented and system performances are currently being investigated. Note that the idea of introducing an additional context variable has been tested in our previous work [12, 11] and proved to be relevant for the simple task of context-dependent sensor selection (equivalent to rejection). However experiments conducted so far concerned the simple task of

altitude estimation for an UAV, and the approach has still to be tested on more complex scenarios, involving numerous sensors and a broad range of contexts.

While augmenting Bayes filters with time-varying noise model plays a central role in trying to compensate the optimistic assumptions usually made by the classical model, the training method might also have major consequences over the system performance. As such, discriminative training seems promising in that it requires to *run* the filter during optimization while generative training focuses on the underlying emission and prediction processes. Discriminative training nevertheless brings some particular issues, since at first, it does not allow to use classical training (and sparsification) method for the RVM model, but also because the optimization process of $\mathcal{L}_{discr}$ is much more complex. Indeed, each term of the discriminative loss function is strongly related to the preceding one as a direct consequence of the recursive equations used for state estimation. Classical RVM models already require the optimization of a nonconvex function, and we still need to study the consequences of the additional complexity introduced along with this specific loss function. Besides this particular issue, the use of discriminative training also ends up with some additional approximations during inference. Experiments will provide a better insight on how expected benefits and inherent drawbacks of the discriminative method impact the system performance, by comparison to the much simpler generative approach.

References

1. Abbeel, P., Coates, A., Montemerlo, M., Ng, A.Y., Thrun, S.: Discriminative training of kalman filters. In: Proceedings of Robotics: Science and Systems. Cambridge, USA (June 2005)
2. Bishop, C.: Pattern recognition and machine learning, vol. 4. Springer New York (2006)
3. Deisenroth, M.P., Turner, R.D., Huber, M.F., Hanebeck, U.D., Rasmussen, C.E.: Robust filtering and smoothing with gaussian processes. CoRR abs/1203.4345 (2012)
4. Ghahramani, Z., Hinton, G.E.: Parameter estimation for linear dynamical systems. Tech. rep. (1996)
5. Goldberg, P.W., Williams, C.K.I., Bishop, C.M.: Regression with input-dependent noise: A gaussian process treatment. In: In Advances in Neural Information Processing Systems 10. pp. 493–499. MIT Press (1998)
6. H. E. Rauch, F. Tung, C.T.S.: Maximum likelihood estimates of linear dynamic systems, (1965)
7. Kersting, K., Plagemann, C., Pfaff, P., Burgard, W.: Most likely heteroscedastic gaussian process regression. In: Proceedings of the 24th International Conference on Machine Learning. pp. 393–400. ICML '07, ACM, New York, NY, USA (2007)
8. Khashabi, D., Ziyadi, M., Liang, F.: Heteroscedastic relevance vector machine. CoRR abs/1301.2015 (2013)
9. Ko, J., Fox, D.: Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models. Autonomous Robots 27(1), 75–90 (2009)
10. Lzaro-gredilla, M., Titsias, M.K.: Variational heteroscedastic gaussian process regression. In: In 28th International Conference on Machine Learning (ICML-11. pp. 841–848. ACM (2011)

11. Ravet, A., Lacroix, S., Hattenberger, G.: Context-dependent bayesian filtering: an approach based on measurement selection and supervised learning. In: To be published in Intelligent Robots and Systems (IROS), 2014 IEEE/RSJ International Conference on (2014)
12. Ravet, A., Lacroix, S., Hattenberger, G., Vandeportaele, B.: Learning to combine multi-sensor information for context dependent state estimation. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on. pp. 5221–5226 (2013)
13. Sivia, D., Skilling, J.: Data Analysis: A Bayesian Tutorial. Oxford science publications, OUP Oxford (2006), http://books.google.fr/books?id=zN-yliq6eZ4C
14. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The MIT Press (2005)
15. Tipping, M.E.: Sparse bayesian learning and the relevance vector machine. J. Mach. Learn. Res. 1, 211–244 (Sep 2001)

Be In The Know: Connecting News Articles to Relevant Twitter Conversations

Bichen Shi, Georgiana Ifrim, and Neil Hurley

Insight Centre for Data Analytics
University College Dublin
Dublin, Ireland
{bichen.shi,georgiana.ifrim,neil.hurley}@insight-centre.org

Abstract. In this paper we propose a framework for tracking and automatically connecting news articles to Twitter conversations as captured by Twitter hashtags. For example, such a system could alert journalists about news that get a lot of Twitter reactions, so they can investigate those conversations for new developments in the story, promote their article to a set of interested consumers, or discover general sentiment towards the story. Mapping articles to hashtags is nevertheless challenging, due to different language styles of articles versus tweets, the streaming aspect, and user behavior when marking tweet-terms as hashtags. We track the IrishTimes RSS-feed and a focused Twitter stream over a two months period, and present a system that assigns hashtags to each article, based on its Twitter echo. We propose a machine learning approach for classifying article-hashtag pairs. Our empirical study shows that our system delivers high precision and recall for this task.

Keywords: news tracking, social media, Twitter, hashtag recommendation

1 Introduction

Since its start in 2006, Twitter has established itself as an alternative media source. Increasingly, Twitter conversations and calls to action that mobilize masses have dedicated hashtags, as showcased by recent world events, e.g., #ArabSpring, #Syria, #freethe7. Twitter hashtags thus lead to the formation of ad hoc publics around specific themes and topics without the need for the users to be otherwise explicitly connected [2].

Hashtags can convey information about the community that uses them or the sentiment of the messages they group. For an outsider, or even for an insider that doesn't continuously track the massive Twitter activity, it is close to impossible to stay in the know when it comes to the right hashtags or users to follow, for current and developing news stories. Nevertheless for journalists in particular, it is vital to get to the right hashtags quickly, in order to be able to follow new developments on topics of interest. Data analytics techniques can provide tools that link news stories to the relevant Twitter conversations.

Automatically mapping news articles to appropriate hashtags (where a hashtag is seen as a group of tweets forming a conversation around it) can be very challenging. This is due to different language styles used in the two types of data (e.g., clean, long

Table 1. A news article and initially retrieved hashtags (before learning algorithm is applied).

News Article	Retrieved Hashtags	Hashtag Category
Headline: *FG fears day of reckoning over Enda Kennys Seanad gamble*	#seanad, #enda, #irelandsaysno	Relevant (Specific)
Sub-headline: *There is deep concern within the Fine Gael ranks that its populist referendum campaign misfired so badly*	#ireland, #rtept, #news	Relevant (General)
	#caughtrotten, #whip	Relevant (Abusive)
	#mentalhealth	Irrelevant

articles versus messy, short tweets), the fast paced streaming aspect of both news and tweets (matching two streams moving at different speeds), as well as user behaviour when coining certain tweet-terms as hashtags. To showcase the third issue, in Table 1 we present an example news article and the categories we identified for the hashtags retrieved for it, in an initial pre-processing stage. The article is about Irish politics: the 2013 referendum to adopt a unicameral parliamentary system by abolishing one of the current two houses of parliament, the Seanad. The hashtags retrieved for this article in an initial pre-processing step, range from highly specific and relevant, to general but still relevant, to abusive but potentially relevant, to irrelevant. We can see from this example that an approach that can accurately filter irrelevant hashtags and rank relevant hashtags can deliver value by connecting to the right Twitter conversations.

In this paper we propose a large-scale, real-time framework for connecting news articles from mainstream media to their echo on the Twitter stream. We discuss the data collection process for continuously gathering, processing and connecting a stream of news articles and a focused Twitter stream relevant to the tracked news stories. We analyze relevant features and propose a machine learning approach for ranking hashtags for a given news story. Our experiments show that our system can achieve high precision and recall on this task. The rest of the paper is organized as follows. Section 2 discusses related work and our contributions. In Section 3 we explain the data collection process, while in Section 4 we describe the process of modeling hashtag ranking as a learning problem. In Section 5 we discuss our results and Section 6 concludes with directions for future work.

2 Related Work

Recent years have seen an explosion of research work analyzing social media (e.g., most prominently the micro-blog Twitter) and the connection between traditional media and this new form of reporting. Among the diverse investigations of Twitter data, two categories are most relevant to this paper.

Hashtag Recommendation. Tag recommendation for tagging systems such as Last.fm and Delicious has been studied in a number of works such as [7] that applies topic modelling using Latent Dirichlet Allocation (LDA) to the problem. Focusing in particular on hashtag retrieval over a Twitter corpus, in [5], language modelling is used to find hashtags given a keyword query. A model of each hashtag is learned from the set of tweets that contain the tag as a multinomial distribution over terms. Hashtags are ranked according to the Kullback Leibler divergence of their corresponding model to the query model. In [4] the issue of recommending hashtags to untagged tweets is addressed. An

LDA topic model is used to categorise tweets into topics and a translation probability maps topics to hashtags. The method is modified in [3] by replacing standard LDA with the topic model of [14].

News and Tweets. Work that investigates the connection between news and Twitter includes [11]. Given a set of tweets that specifically mention the URL of a given article, this work focuses on a method to filter this set into a subset of most interesting tweets. The authors use four indicators of interestingness, namely informativeness, opinionatedness, popularity and authority to filter the initial set. TweetMogaz [8], a system for microblog search and filtering, aims to find tweets relevant to regional news. It relies on a curated list of *key players* from which to collect an initial set of relevant tweets. The initial set is augmented, by firstly extracting a set of keywords from news sites and searching for tweets containing these keywords. The *keyword* tweets are filtered by training a classifier using the *key player* tweets as positive examples and a set of random tweets as negative examples. Other works investigated automatic news detection from tweets [12], recommending news articles using tweets [9], forecasting the popularity of news using Twitter [1], or enhancing news articles with information extracted from Twitter, such as *comment tweets* [6].

Our work differs from the above research in a number of ways. In particular, we address hashtag recommendation in a streaming context, with a requirement that the model be updated on a daily basis. Rather than applying topic modeling on a large, static Twitter corpus, containing potentially many diverse topics, we attempt to filter irrelevant tweets directly by using the news articles to be hashtagged in order to focus the data collection from the Twitter stream. Nevertheless, unlike other work on connecting articles and microblogs, we avoid seeding our data collection with a curated user group or with tweets that specifically mention the articles in question (via the URL). As discussed later, our automatic-keyword Twitter stream allows for a wide set of tweets to be gathered, while ensuring that the collection contains relevant tweets with high probability. Our search strategy provides sufficient breadth to allow high recall in gathering relevant hashtags, while avoiding being drowned in a vast sea of Twitter noise. We alternate this high recall with a high precision oriented step, by using a learning approach to rank the retrieved hashtags for each article.

Our contributions are as follows: (1) we propose a focussed Twitter data collection strategy based on automatic keyword extraction from news articles; (2) we formulate a large-scale, real-time learning process for assigning hashtags to news articles; (3) we deliver a system for matching a daily news stream and a relevant Twitter conversation stream [10].

3 Data Collection

We collect data from two sources between October 7, 2013 and November 30, 2013, RSS Feeds of news articles and a focused Twitter stream.

News Articles from RSS Feeds. We gathered the news articles streamed on the Irish Times RSS feeds, by polling the RSS feeds every 5 minutes, yielding a total of 4,862 unique articles, around 170 articles per day. The Irish Times is an Irish mainstream media outlet, that covers Irish news and high impact world news. Each article has a

Table 2. Processed keyword set by permutation.

Original keywords/phrases	Final keywords/phrases
enda kenny	enda kenny
fine gael	fine gael
fg	fg fears
fears	fg seanad
seanad	fears seanad

headline, a one paragraph description that summarises the article (sub-headline), and the article body.

We extract representative keywords for each downloaded article, by parsing the headline and sub-headline, part-of-speech tagging this text, and extracting nouns and named entities using shallow parsing techniques and heuristics (e.g., we extract Aer Lingus, Enda Kenny, etc.). We do not use the article-body for keyword extraction, since it poses risks of topic drift and noise. For example, for the news article in Table 1 we extract the keywords *enda kenny, fine gael, fg, fears, seanad*.

Focused Twitter Stream. Since we are interested in continuously streaming news and corresponding tweets, we use the Twitter Streaming API[1], which can be employed with either keywords (words or phrases), geographical boundary boxes or user ID. We gather Twitter streaming data by using a dynamic[2] set of keywords extracted from the stream of RSS news articles every 30 mins each day. Each article's keywords are streamed for 24 hours.

Additionally, we noticed that in order to get relevant tweets, it helps if we constrain each tweet returned by the Twitter API to contain at least two article keywords. We achieve this by splitting our original keyword set, into individual keywords, and creating all possible permutation pairs as our final keyword set, with the constraint that we freeze named entities. For example, for the article in Table 1, we process the keyword set *enda kenny, fine gael, fg, fears, seanad* by keeping the named entities and permuting the single keywords to form pairs, as shown in Table 2. We apply this process every 30 minutes to *all* the RSS articles downloaded up to that point in time, pool the keywords together, and re-connect with the Streaming API using the updated keyword list.

Through this process we aim to retrieve a large set of relevant tweets while not being restricted to a set of manually curated user lists, locations or keywords. The problem of retrieving relevant tweets to a set of news has been pointed out in recent research [6] with ad-hoc retrieval techniques achieving low Recall (0.5). Prior work relies mostly on tweets where the url of the article is explicitly provided, therefore obtaining a clean but potentially small set of tweets. Our initial tweet-retrieval process gathers a large set of potentially relevant tweets (23.3 million unique tweets), which we carefully filter in the following machine learning process.

[1] https://dev.twitter.com/docs/streaming-apis

[2] Dynamic refers to the list of extracted keywords being updated every 30 mins with the new keywords of incoming articles

4 Learning Algorithm for Scoring Hashtags

In this section we discuss the process of modelling hashtag selection as a learning problem. We parse the stream of news articles and the Twitter stream in a 24 hours timewindow, in order to extract and rank hashtags for each news article. For tweet-processing, we remove stop words, punctuation, URLs and user names, and apply stemming to the remaining terms. For each day, and each news article, we separate the tweets of the corresponding Twitter stream *per article*, based on a shallow matching of tweet keywords and article-keywords (as extracted for the Streaming API and showcased in Table 2). This results in a local tweet-bag per article, that can be analyzed for extracting hashtags and hashtag information, e.g., frequency, tf.idf profile describing the hashtag as reflected in its tweet-bag. Next, we form article-hashtag pairs, and compute features of each pair useful for discriminating whether a hashtag is relevant to a given article.

Features. We extract four features for each article-hashtag pair: two features that characterize the local hashtag profile (based on tweets in the article-tweet bag), while the other two characterize the global hashtag profile (based on tweets in the entire Twitter stream that are retrieved till that time point), useful for describing specific versus general hashtags.

One of the first features we select is the *local cosine similarity* between the tf.idf profile of the article and the local tf.idf profile of hashtag (as extracted from the tweets mentioning that hashtag in the article tweet-bag, by considering tweets as an *articles* and calculate tf.idf weight of them). To avoid noise in the article tf.idf profile, rather than selecting terms from the full article-body, we only select them from the headline and sub-headline, but compute their tf.idf weight using the entire article (stop words removed, stemming). Additionally, we extract the *local frequency* of the hashtag (i.e., the number of tweets in the article tweet-bag, mentioning that hashtag). The hashtag frequency captures whether a hashtag is actively used.

We compute the *global cosine similairy* between the local and the global hashtag tf.idf profile (as extracted from tweets in the entire historical Twitter stream[3], mentioning that hashtag), to asses how much does the global hashtag profile diverge from the local profile. Note that globally, the same hashtag may refer to different events, or a hashtag may be preferred over a time window to refer to a certain event, and then slowly discarded or outweighed by other hashtags. Therefore, using local and global features for each hashtag, addresses the issue of time-of-use and scope of a hashtag. we also extract the *global frequency* of the hashtag (i.e., the number of tweets in the entire Twitter stream, mentioning that hashtag).

For each article-hashtag pair, we now have four features describing how relevant a hashtag may be for a given article. We normalize all four features to the $[0, 1]$ interval. Next, we discuss how to use these features and a set of manually labeled article-hashtag pairs for learning to identify relevant hashtags.

Labeled Data. In order to build a classification algorithm for recognizing relevant hashtags, we need labeled article-hashtag pairs. We selected two days at random from the two month dataset, October 23, and November 23, 2013, extracted all the article hashtag pairs and their features as described above, and asked two annotators to manually

[3] Historical refers to the time the article was published

label each pair. We asked the annotators to decide which of the three scenarios applies to each pair: (1) a hashtag is *specific and relevant* to the topic of the news article, (2) *general and relevant*, or (3) *irrelevant*. For abusive hashtags, the annotators were advised to decide which of the three scenarios the hashtag belongs to depending on the local context. For the purpose of our experiments, we merged the first two classes into simply relevant (a positive example in binary classification) or irrelevant (negative example). The inter-annotator agreement was 80%. We used the subset of examples where both annotators agreed for training/testing a classification algorithm.

Besides of manually labeled data, we also gathered tweets containing both the Irish Times article's URL and hashtags. These tweets naturally form Article-User Hashtags pairs and can be used as a form of ground truth, by assuming all user assigned hashtags are relevant to the article.

Classification Algorithm We train and test our approach by employing a series of Weka [13] classification algorithms using default settings, including Logistic, kNN, Decision Tree, Multilayer Perceptron etc. The algorithm only sees the examples as described by the four features, and can learn thresholding strategies on the provided features. For example, to classify a hashtag as relevant for a given article, a classification algorithm may learn (from the training set) that the cosine feature should be higher than 0.5 and the hashtag frequency should be close to 1. Additionally, most classifiers provide a score describing the likelihood that a hashtag is relevant to the article. We use this classification score to rank hashtags for each article.

5 Evaluation

In order to evaluate our overall strategy for retrieving, learning, and ranking hashtags, we evaluate classifiers using three different settings: **Small**, **Medium** and **Large**. The evaluation settings and results are shown in Table 3.

5.1 Results: Small Experiment

For the small experimental setting, we use the manual labelling of article-hashtag pairs for two random days: October 23, 2013 (874 examples) for training, and November 23, 2013 (1,122 examples), for testing.

Baselines. We first evaluate two simple baseline techniques. On the test set (November 23, 2013), we select the top-3 hashtags per article (257 pairs out of 1,122), using the *highest local hashtag frequency* and the *highest local cosine similarity*. We evaluate the precision of these two baseline. Since we only take the top-3, recall is not applicable in this case.

Learning Approach. We now evaluate the classifier's ability to retrieve all the hashtags deemed relevant by our annotators as well as its ability to rank them before the irrelevant ones. We experimented with a series of Weka classifiers, with default parameter settings. MultilayerPerceptron, Logistic (regularised logistic regression) and Kstar (K-nearest neighbours with entropy-based-distance) delivered the best results, as shown in Table 3. We note that all three classifiers have high precision (0.85), recall (0.80) and AUC (0.92), showing that the classifier ranks relevant hashtags before irrelevant ones.

Table 3. The evaluation results of **Small**, **Medium** and **Large** experiment settings.

Settings	Training Set		Testing Set		Results					
						Baseline		Learning Approach		
	Dataset	Size	Dataset	Size		Most Frequent Top3	Highest Cosine Top3	Multilayer Perceptron	Logistic	Kstar
Small	Oct 23	847	Nov23	1122	Accuracy	-	-	**84.6%**	84.4%	83.9%
					Precision	0.548	0.634	0.850	**0.876**	0.861
					Recall	-	-	**0.807**	0.770	0.774
					F1	-	-	**0.846**	0.844	0.839
					AUC	-	-	0.921	**0.924**	0.911
Medium	Oct 23	847	Article-User Hashtags	1147	Recall	0.503	0.644	**0.781**	0.756	0.704
	Nov 23	1122						0.792	**0.808**	0.787
	Oct 23 & Nov 23	1996						**0.845**	0.797	0.776
Large	Oct 23 & Nov23 & Article-User Hashtags	3143	Randomly Selected Article-Hashtag Pairs	1029	Precision	-	-	-	0.869	-
					Precision@1	-	-	-	0.900 ([0.871, 0.929], $p < 2.2e-16$)	-
					NDCG@3	-	-	-	0.877 ([0.850, 0.904], $p < 2.2e-16$)	-

The AUC is particularly important, since ultimately it is useful to rank the hashtags of each document, from most relevant to least relevant. Additionally, the Logistic classifier is a linear model that can be easily interpreted, scales to large data and its classification scores are true probabilities. The Logistic model deemed all four features as important (non-zero weights), with the local cosine feature getting the highest weight, followed by the frequency based features, and ending with the global cosine.

5.2 Results: Medium Experiment

In this setting, we use Article-User Hashtags data, which is gathered from tweets containing both the Irish Times articles' URL and hashtags and can be used as a form of ground truth, by assuming all user assigned hashtags are relevant to the article. The articles with user assigned hashtags are a subset of the total streaming articles set. Since we assume all the test examples are relevant (belongs to the positive class), in this setting the Accuracy is the same as Recall, and Precision is 1 by default.

Baselines. We still employ the two baseline techniques: On the test set (Article-User-Hashtags), we select the top-3 hashtags per article (257 pairs out of 1,122), using the *highest local hashtag frequency* and the *highest local cosine similarity*. We evaluate the recall of these two baseline.

Learning Approach. As training data we analyze three settings, October 23, 2013, November 23, or both days together as training, and Article-User-Hashtag pairs as test (1,147 test examples). Note that we assume that all the user-assigned hashtags are relevant, which may not necessarily be the case, since sometimes users also assign spurious hashtags, e.g., #annoying #omg. Our algorithm may consider such hashtags as irrelevant.

Table 3 shows Recall over all article-user-hashtags retrieved by our learning algorithm as being relevant (classification score above 0.5). We note that when we increase the amount of training data, by combining the October and November examples, the accuracy of MultilayerPerceptron stands at 84.5%, a similar value to that of the small setting experiment.

5.3 Results: Large Experiment

In this setting, we use the labeled examples of October 23, November 23, and the Article-User Hashtags data as training data, and around 1000 randomly selected article-hashtag pairs extracted from the RSS and Twitter streams (no labels) as testing data.
Baselines. No baselines are employed in this setting due to the expensive cost of human evaluation.
Learning Approach. Due to cost of manual evaluation, we only use the Logistic classifier in this setting, for the reason that it performs well in the previous two settings, and it runs in linear time, so the scalability would not be an issue.

We apply the trained Logistic classifier to all the article-hashtag pairs. We randomly select 422 articles that get at least one relevant hashtag (based on classification score above 0.5), and manually asses the relevant article-hashtag pairs (1,029 pairs), using 0, 1 and 2 relevancy scores. As shown in Table 3, we evaluate both the filtering quality, i.e., the classification across all article-hashtag pairs (to asses the Precision over the pairs classified as relevant), as well as the hashtag *ranking quality per article*, using information retrieval metrics. For the article oriented metrics, we use Precision@1 and NDCG@3 and average them across all articles.

We note that the precision for the filtering step (binary classification into relevant/irrelevant) is fairly high (Precision 0.86), and similar to what we have seen in the previous experiments. When we evaluate the quality of ranking of hashtags for each article, we see a similar result: the Precision@1 is 0.9, while the NDCG@3 which penalizes relevant hashtags ranked at low ranks, is 0.87.

5.4 Discussion

In order to make the whole methodology more explicit, in Table 4 we show some example articles from our annotated sample of the **Large** setting, their extracted keywords, their (up to top-3) ranked hashtags, together with the features extracted for the corresponding pair, the classifier score and the annotator relevance score. We observe that for local as well as international news (first 3 articles), the hashtags assigned and ranked by classifier score are relevant and quite specific (e.g., #ecb, #walshwhiskeydistillery).

We found three main reasons why an article does not get any (relevant) hashtag: the article-keyword extraction process is faulty (due to part-of-speech tagging errors or due to the fact that the extracted keywords are too generic); there is no discussion on Twitter about that particular news story; the tweets relevant to an article do not contain any hashtags. The aspect of assigning noisy or irrelevant hashtags can be mitigated to some extent by tuning the classifier threshold (here we used the default classification score of 0.5). Additionally, the four features describing each article-hashtag pair could

Table 4. Example results from our two-month annotated sample.

Article Headline	Article Keywords	Hashtag	LFr	LCo	GFr	GCo	ClassifScor	RelScor
Tech titans in town for Dublin Web Summit	dublin, dubstarts, summit, tech, web	#websummit	1.00	0.35	0.58	0.82	0.92	2
		#tech	0.23	0.45	0.70	0.37	0.89	1
		#web	0.42	0.41	0.40	0.52	0.82	2
Whiskey distillery to create 55 jobs for Co Carlow	carlow, co, walsh, distillery, whiskey	#whiskey	1.00	0.73	0.16	0.56	0.99	2
		#carlow	0.90	0.64	0.16	0.53	0.99	2
		#walshwhiskey-distillery	0.66	0.61	0	1	0.97	2
ECB's Draghi moves to ease fears on interest rates	banks, draghi, ecb	#ecb	1.00	0.50	0.39	0.42	0.97	2
		#draghi	0.54	0.58	0.19	0.69	0.96	2
		#news	0.00	0.46	0.89	0.29	0.90	1
Climate change watchdog must be robust and independent, says report	advisory, climate, expert, fiscal	#delleir	1.00	0.27	0.00	1.00	0.60	0
		#job	0.00	0.30	0.80	0.35	0.53	0
		#delcfe	0.89	0.26	0.00	1.00	0.51	0
Europe bank payouts capped as capital bar keeps rising	capital, europe	#europe	0.79	0.35	0.55	0.02	0.84	1
		#travel	0.66	0.27	0.68	0.38	0.72	0

be enhanced, e.g., using user authority to re-weight tweets, filtering spammy hashtags (e.g., #ff, #followback). Regarding the lack of hashtags in the tweet-bag of an article, in such cases we could employ recent techniques for extracting informative tweets [11], or adapt our approach for the problem of assigning Twitter users (rather than hashtags) relevant to a given news article. The manual annotation for the learning approach is also potentially noisy, since at times it is quite difficult to decide whether a hashtag is relevant or not, without considerable background knowledge. In this respect we plan to employ crowd sourcing platforms such as Crowdflower, in order to obtain larger and possibly cleaner labeled datasets, but even then, the annotators require considerable background knowledge for labelling (e.g, political climate in Ireland).

6 Conclusion

In this work we present a framework for connecting news articles to their relevant Twitter conversations, as semantically grouped by Twitter hashtags. We discuss the aspect of continuously tracking a stream of news and tweets, and present an approach for obtaining a large focused Twitter stream, automatically seeded by a dynamic keyword set extracted from the articles. Furthermore, we model the problem of hashtag assignment as a classification problem, and analyze a framework for hashtag retrieval and appropriate features and data for building a hashtag classifier. We evaluate our methods and show that our approach achieves high precision for this task.

Future Work. We plan to improve our approach by avoiding using manually labelled data (which is expensive) and instead using Article-User Hashtag data (which is free to obtain) as training data. This means changing the two class classification problem into one class (without negative examples), or two class with noisy/random negative data. The Article-User Hashtag data collected through the real-time Twitter stream could help re-training the classifier and keeping it up-to-date, to avoid potential concept drift.

We plan to extend our study to track several RSS news feeds and Twitter conversations, and test a prototype with journalists. We also intend to investigate applications of our methods to clustering of articles in hashtag space, story tracking and event detection.

An early prototype Insight4News system[4] integrates the techniques described here, and demonstrates the real-time, large-scale nature of our proposed hashtag recommendation process. Unlike the 24 hours time window used in this paper, the Insight4News system updates hashtag recommendation for articles every 15min, and processes around 300 articles pre day.

Acknowledgements This work was supported by Science Foundation Ireland under grant 07/CE/I1147.

References

1. Roja Bandari, Sitaram Asur, and Bernardo A Huberman. The pulse of news in social media: Forecasting popularity. In *ICWSM*, 2012.
2. Axel Bruns and Jean E Burgess. The use of twitter hashtags in the formation of ad hoc publics. 2011.
3. Zhuoye Ding, Xipeng Qiu, Qi Zhang, and Xuanjing Huang. Learning topical translation model for microblog hashtag suggestion. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 2078–2084. AAAI Press, 2013.
4. Zhuoye Ding, Qi Zhang, and Xuanjing Huang. Automatic hashtag recommendation for microblogs using topic-specific translation model. In Martin Kay and Christian Boitet, editors, *COLING (Posters)*, pages 265–274. Indian Institute of Technology Bombay, 2012.
5. Miles Efron. Hashtag retrieval in a microblogging environment. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 787–788. ACM, 2010.
6. Alok Kothari, Walid Magdy, Ahmed Mourad Kareem Darwish, and Ahmed Taei. Detecting comments on news articles in microblogs. *ICWSM 2013*, 2013.
7. Ralf Krestel, Peter Fankhauser, and Wolfgang Nejdl. Latent dirichlet allocation for tag recommendation. In *Proceedings of the third ACM conference on Recommender systems*, pages 61–68. ACM, 2009.
8. Walid Magdy. Tweetmogaz: A news portal of tweets. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 1095–1096, New York, NY, USA, 2013. ACM.
9. Owen Phelan, Kevin McCarthy, Mike Bennett, and Barry Smyth. Terms of a feather: Content-based news recommendation and discovery using twitter. In *Advances in Information Retrieval*, pages 448–459. Springer, 2011.
10. Bichen Shi, Georgiana Ifrim, and Neil Hurley. Insight4news: Connecting news to relevant social conversations. In *ECML/PKDD*, 2014.
11. Tadej Štajner, Bart Thomee, Ana-Maria Popescu, Marco Pennacchiotti, and Alejandro Jaimes. Automatic selection of social media responses to news. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 50–58. ACM, 2013.
12. Ilija Subašić and Bettina Berendt. Peddling or creating? investigating the role of twitter in news reporting. In *Advances in Information Retrieval*, pages 207–213. Springer, 2011.
13. Ian H Witten, Eibe Frank, and Mark A Hall. *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*. Elsevier, 2011.
14. Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. Comparing twitter and traditional media using topic models. In *Advances in Information Retrieval*, pages 338–349. Springer, 2011.

[4] `http://insight4news.ucd.ie/insight4news/`

Evaluating Collaborative Filtering: Methods within a Binary Purchase Setting

Stijn Geuens, Kristof Coussement, Koen W. De Bock
IESEG School of Management – Université Catholique de Lille (LEM, UMR CNRS 8179), Lille, France
{s.geuens,k.coussement,k.debock}@ieseg.fr

Abstract. The study of recommender systems based upon implicit binary purchase data constitutes an under-investigated area. The objective of this study is to evaluate configuration options of memory-based collaborative filtering (CF) for generating recommendations based upon online binary purchase data with different characteristics. First different algorithm configurations are identified. More specifically, three important algorithm parameters are investigated: the data reduction technique, the user- versus item-based CF and the similarity measure. Results on synthetic datasets show that these three factors influence the accuracy results of an algorithm. In a second analysis, extended experiments are set up to gain more insight into the influence of input data characteristics on the relative success of the CF configuration options. In particular, three input characteristics, sparsity level, item purchase distribution and item/user ratio are manipulated to analyze the impact on the algorithm's best configuration. Results show that the best performing algorithm is consistent, independently from the input data characteristics.

Keywords: Collaborative filtering, purchase data, synthetic data, evaluation metrics, data reduction, similarly measure

1 Introduction

In a typical E-commerce setting a customer receives an overload of information. In these situations, it is impossible for a customer to make optimal or even good product evaluations and purchase decisions. To cope with this overload of information, recommendation systems are designed. Besides the clear advantage for the customers, E-tailors also benefit from these systems, because they increase sales, revenue and loyalty [1].

A common used technique in recommendation systems is collaborative filtering (CF). CF systems propose a personalized set of items based on the customer's past behavior and activities of its peers. A good overview of the main existing algorithms is given in [2] and [3]. In a vast part of the literature, recommendations are based upon explicit customers' ratings [4]. These systems base product propositions upon ratings explicitly given by customers and its peers on a rating scale. Although these ratings clearly represent the customer's preference, it demands the user's effort, time and cost [5]. Additionally, results can be biased because customers have difficulties expressing their interest, leading to arbitrarily given or incorrect ratings. In most systems only a small fraction of the products purchased is rated, resulting in only a partial view of a customer when using explicit data [6].

To overcome these problems, implicit ratings can be used [6]. In contrast to explicit ratings, implicit ratings do not require user feedback, but derive affinity with items from actual user behavior. In an online retail setting characterized by a broad, deep and fast-changing product offering, explicit feedback is often hard to (sufficiently) collect. The collection of implicit feedback, on the other hand, is objective and non-intrusive. Implicit ratings come in all kinds of forms. For an overview we refer to Palanivel and Sivakumar [6]. In this study binary purchase data is used as a recommendation basis [7, 8]. In contrast to recommender systems based on explicit user feedback, systems using implicit purchase information remain under-investigated. Purchases are an important KPI for every commercial company, which makes it interesting to investigate a recommender based on purchase data.

The proposed study investigates recommendation systems based on implicit binary purchase data. In particular, different variations of the memory-based collaborative filtering algorithm are tested on binary purchase datasets having distinct input characteristics. Based on this setup, an experimental design is constructed to analyze the impact of input characteristics of a purchase dataset on the optimal algorithm configuration.

The remainder of this paper is organized as follows. The next section discusses some related work to better grasp the background of this study. A third section describes the setup of the experimental design. A fourth section presents the accuracy results based on 54 synthetic datasets. Finally, section five highlights the conclusions and next steps of the project.

2 Related Research

Recommendation systems use available customer information to create relevant personalized product suggestions. These recommendations can be based on different kinds of data. A typical classification of algorithms is presented by Bobadilla et al. [3]. First, *content-based* systems use past buying behavior of a customer to link these purchases to similar products based on characteristics. Second, *demographic-based* systems exploit socio-demographic data to predict relevant recommendations based on similarity in customers' characteristics. Third, *collaborative filtering* algorithms use past behavior of a customer, but, in contrast to content-based systems, they do not use product characteristics. Collaborative systems identify customers exhibiting the same behavior. Based on actions of these peers, products are suggested. Fourth, by combining different algorithms, *hybrid solutions* can be created, using different kinds of data to optimize the performance of the algorithms.

This study uses collaborative filtering algorithms based upon implicit binary purchase datasets pre-processed by a data reduction technique, as further elaborated in this related research.

2.1 Implicit Binary Purchase Data

The collaborative filtering literature typically refers to situations in which a customer expresses a preference by giving an explicit rating to a certain item [4]. In these cases recommendations are based upon the ratings explicitly given by a customer. As discussed in the introduction, using explicit data can have some flaws.

This study focuses though on implicit data and in particular on binary purchase data [7, 8]. This particular information, directly related to purchase behavior remains under-investigated in literature. As in the general case of recommender systems, algorithms possibly suffer from problems related to the input characteristics of the binary purchase matrix [9]. This study investigates sparsity, purchase distribution and item/user ratio.

Sparsity. A common recommender problem is the curse of dimensionality [9]. Typically an input matrix is very sparse, since a customer only buys a limited number of products and so products are only purchased by a limited number of customers. CF configurations tend to have difficulties with this scalability and sparsity, possibly influencing the model performance. A sparser input dataset leads in many cases to a decrease in accuracy and coverage of the proposed algorithm [9]. Typically datasets are very sparse, indicating this characteristic could indeed be an important factor to bear in mind.

Purchase Distribution. A second possible input characteristic problem is the distribution of purchases. Typically some items are very popular, but most products are only bought a few times. CF has the tendency to be less accurate towards the long tail and promoting the popular products [10].

Item/User Ratio. A third factor influencing the performance of CF is the item/user ratio. In settings with a low item/user ratio, it might be beneficial to use item-based algorithms since they are less computationally expensive. Moreover a user-based algorithm could be preferable in a setting with a high item/users ratio [11]. On top can the prediction accuracy of an algorithm depend on the ratio between items and users.

2.2 Data Reduction

Data reduction has as indirect advantage that sparsity is reduced [9]. One of the possibilities is using data reduction in the pre-processing phase of the algorithm's building process [5]. Although very popular in an explicit ratings context, reduction techniques are less used on implicit binary data [8, 12,13], while other fields of research frequently use binary reduction techniques [4, 14].

In this study reduction techniques are only used as pre-possessing step of the collaborative filtering procedure to gain efficiency and memory. Although direct imputation methods based on decomposed matrices are used in the past, this technique is not replicated in this study. The main reason for not using the direct method is the structure of the input matrix. Since direct imputations have as goal to estimate blanks in the original matrix and our input matrix only contains 0/1's, no purchase/purchase, and no missing values, direct imputation is less useful.

Four data reduction techniques are used in the pre-processing phase of the algorithm building process. Three popular reduction techniques in literature are singular value decomposition (SVD) [5], nonnegative matrix factorization (NMF) [4] and logistic principal component analysis (LPCA) [15]. These methods will be used in this study. Additionally a fourth reduction technique, correspondence analysis (CA), is applied. This method is conceptually similar to PCA, but can only be applied to binary data [16]. CA is never used in recommendation settings. Next to the reduced input matrices, the non-reduced purchase matrix is used as input.

2.3 Memory Based Collaborative Filtering

This paper focuses on memory-based collaborative filtering based on binary purchase data. Memory based CF is one of the two main distinctions in CF. In contrast to model-based CF [2], memory-based CF does not estimate a model to make recommendations. It only uses the user-item input matrix to calculate recommendations [17].

Until today memory-based CF remains one of the most popular algorithms in literature. The fact that it is only using the user-item matrix as input is a big advantage, since no extra data has to be gathered. The algorithm uses actual customer behavior as input for making the recom-

mendations. Based on the user-item matrix, similarity is calculated and predictions are often based on the k-nearest neighbor algorithm [17].

CF Methods. Two possible distinctions exist in terms of the used CF method. An algorithm can be user- or item-based. The former type of system calculates similarity between customers. Products are proposed to users based on the behavior of their nearest neighbors [17]. In contrast, an item-based system calculates similarity between products and proposes similar items compared to the items purchased by a customer.

Similarity Measure. To identify nearest neighbors, similarity has to be calculated. In literature, many measures are considered [18]. In binary data settings cosine, Pearson correlation [13] and Jaccard's similarity [19] are often used. In contrast to cosine and Pearson correlation similarity, Jaccard's measure is based on set theory and can only be used on binary data.

$$\text{Cosine Similarity}(x,y) = \frac{\sum_{i \in I_{xy}} p_{x,i}\, p_{y,i}}{\sqrt{\sum_{i \in I_x} p_{x,i}^2}\sqrt{\sum_{i \in I_y} p_{y,i}^2}} = \frac{p_{x,i} \cdot p_{y,i}}{\sqrt{\sum_{i \in I_x} p_{x,i}}\sqrt{\sum_{i \in I_y} p_{y,i}}}. \quad (1)$$

$$\text{Pearson Correlation Similarity}(x,y) = \frac{\sum_{i \in I_{xy}} (p_{x,i} - \overline{p_x})\,(p_{y,i} - \overline{p_y})}{\sqrt{\sum_{i \in I_{xy}} (p_{x,i} - \overline{p_x})^2 \sum_{i \in I_{xy}} (p_{y,i} - \overline{p_y})^2}}. \quad (2)$$

$$\text{Jaccard Similairty}(x,y) = \frac{x \cap y}{x \cup y}. \quad (3)$$

Equations (1) and (2) represent respectively the cosine and Pearson correlation similarity measures. In these formulas $p_{x,i}$ and $p_{y,i}$ represent the purchase of item i by respectively customer x and y in a user-based setting. In an item-based setting, $p_{x,i}$ and $p_{y,i}$ represent respectively the purchase of product x and y by customer i. $\overline{p_x}$ and $\overline{p_y}$ represent respectively the mean purchase rate of customer x and customer y in a user-based setting. In an item-based setting, $\overline{p_x}$ and $\overline{p_y}$ refer to the mean purchase rate of product x and product y. I_{xy}, I_x and I_y represent the set of products bought by respectively customer x and y, customer x and customer y.

Equation (3) represents Jaccard's formula for calculating similarity between two binary purchase vectors. In this formula $x \cap y$ is the number of products bought by both customer x and y in a user-based setting. In an item-based setting, $x \cap y$ represents the number of customers purchased both product x and y. $x \cup y$ represents the number of products bought by at least one of both customers in a user-based setting. In an item-based setting, $x \cup y$ represents the number of customers who purchased at least one of the products x and/or y.

3 Setup of an Experimental Design

To analyze the link between input characteristics of the binary purchase input matrix and algorithm variations, a 5 x 2 x 3 between-subjects experimental design is constructed. The conditions of the experiment are the algorithm variations as discussed in related research. All algorithm variations are tested on 54 synthetic datasets with different input characteristics [20, 21], which are discussed in paragraph 3.1. Afterwards, results are calculated and algorithms are

compared to obtain robust results identifying the best algorithm combination for given input characteristics.

3.1 Input Data Characteristics

Every binary implicit data matrix is unique and has its own characteristics. These characteristics can influence the optimal algorithm configuration, discussed in paragraph 3.2 and the final results of the model. In this proposed study, three important data characteristics are investigated on synthetic datasets: sparsity level, purchase distribution over products and item/user ratio.

Synthetic Data Characteristics. In order to mimic real-life situations, 54 synthetic datasets, characterized by six levels of sparsity, three different purchase distributions and three distinct item/user ratios, are created.

Sparsity levels. Sparsity levels are artificially created to mimic possible real-life situations. Since a typical recommendation setting consists of very sparse input matrices, six sparsity levels are variable between 95% and 99.50% [20, 22, 23].

Purchase Distribution. Purchases typically show a long tailed distribution over products. Some popular items are bought frequently, but most items are purchased only a few times [10]. In the study, the input is varied from a logarithmic distribution, having a very long tail over a linear distribution with a moderate tail to a uniform distribution.

Item/User Ratio. A last manipulated input characteristic is the item/user ratio. By simply varying the number of rows of the binary input matrix, the ratio adjusts [20]. The synthetic datasets consist of 1 000 items, but the number of customers is adjusted. The number of users is set to 500, 1 000 and 2 000, resulting in item/user ratios of respectively 2, 1 and 0.5.

Synthetic Data Generation. Although the aim of the datasets is to be generic and generalizable, a certain structure needs to be inherently present in the data to be able to discover patterns. To create this structure, the correlation matrix of a binary purchase dataset of a European E-tailor is mimicked in the synthetic datasets.

$$r_{ij} = \frac{p_{ij} - p_i p_j}{\sqrt{p_i q_i p_j q_j}} \Leftrightarrow p_{ij} = r_{ij}\sqrt{p_i q_i p_j q_j} + p_i p_j, \text{ where } i, j \in \{1, \dots, I\}\,. \quad (4)$$

Equation 4 shows the relationship between the correlation, from the test dataset, the marginal probabilities and pairwise joint probabilities between two item vectors i and j [24]. r_{ij} indicates the correlation between vector i and j, p_i and p_j represent the marginal probabilities of item i and j, q_i and q_j denote $(1 - p_i)$ and $(1 - p_j)$ and p_{ij} represents the pairwise joint probabilities between vector i and j. Based on equation 4 purchase probabilities of each synthetic product (item vector) can be generated for a certain number synthetic customers. For each synthetic product i a purchase probability vector $P_i = [p_{i1} \dots p_{iC}]$, where C stand for the number of customers, can be generated.

Next to taking into account the correlation structure, the described step gives the ability to control the item/user ratio and sparsity level. By setting the number of customers (C), the item/user ratio is regulated. By adjusting the marginal purchase probabilities by a constant, sparsity is set. The same constant should be applied to every item to not to disrupt the purchase probability structure.

Purchase probabilities are continuous elements in [0, 1], whereas a binary input matrix should be created. Transforming the purchase probability vector P_i to a binary purchase vector $B_i = [b_{i1} \dots b_{iC}]$ can be done by applying a threshold. Every element b_{ic} is defined by equation 5.

$$b_{ic} = \begin{cases} 1, p_{ic} \geq f(i) \\ 0, p_{ic} < f(i) \end{cases}, \text{where } c \in \{1, \dots, C\}. \quad (5)$$

Implementing equations 4 and 5 gives the possibility to create binary item purchase vectors accounting for item/user ratio and sparsity. To be able to vary the purchase distribution, functional thresholds f(i), as described in equations 6, 7 and 8, are applied. Each of these equations results in different structures of the binary input matrix.

Firstly, a logistic function is applied as threshold to create a logarithmic distribution of purchase frequencies of each item. The resulting distribution is characterized by a limited number of products frequently purchased and a lot of products only purchased a few time.

$$f(i) = \beta_1 \log_{(C+\gamma_1)}(i+1), \text{where } i \in \{1, \dots, I\}. \quad (6)$$

Secondly, a linear function is imposed to give the purchase distribution a linear structure.

$$f(i) = \beta_2 \frac{i}{(C+\gamma_2)}, \text{where } i \in \{1, \dots, I\}. \quad (7)$$

Finally, a constant is set as threshold to create a uniform purchase distribution.

$$f(i) = \beta_3 \quad (8)$$

3.2 Experimental Setup

To get a better notion which algorithms perform best in combination with certain input characteristics, different algorithm configurations are calculated on input datasets discussed above. The algorithms are based on memory-based CF, but differ in dimension reduction method, CF method (item- vs. user-based) and similarity measure, as discussed in the related research section.

The used data reduction methods are SVD, NMF, LPCA and CA. Together with the none reduced purchase matrix, five different input matrices for the collaborative filtering algorithm are created. Item- and user based CF are used as CF-method and cosine, Pearson correlation and Jaccard's similarity constitute the three measures for similarity calculation.

Combining the discussed algorithm configuration elements gives the opportunity to create a 5 x 2 x 3 between-subjects experimental design. Different experimental conditions are constructed by combining each time one of the five proposed reduction techniques with user- or item-based CF and one of the three discussed similarity measures. To make the results generalizable and valid, tests are run on the 54 synthetic datasets**.** In total 1620 individual runs with different input characteristic – algorithm configuration combinations are executed.

3.3 Evaluation Metrics

Results of different CF configurations on different binary input matrices can be assessed using accuracy measures. This allows comparing the performance of the used algorithm variations on input matrices with a variety of specific characteristics.

To evaluate the algorithms, Top-N recommendations consisting of 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150 and 200 items are considered. The accuracy of the predictions is expressed in terms of F1-measure [11], calculated on a random test sample consisting of 20% of the input dataset.

4 Results

Firstly, this study investigates the best algorithm configuration (data reduction technique, CF-method and similarity measure) in general, without taking dataset characteristics into account. This will result in an evaluation of the best overall configuration. Secondly, the best algorithm configuration for each dataset separately is investigated. By comparing different champion models and good alternatives, for datasets with distinct input characteristics, conclusions can be draw with respect to the best input characteristic – algorithm variation combination.

Method. To evaluate the effect of the different algorithm configurations on the different synthetic datasets an ANCOVA analysis is run.

$$\begin{aligned} F1_i = \beta_{0i} + \beta_{1i}RM + \beta_{2i}CFM + \beta_{3i}SM + \beta_{4i}RM * CFM + \beta_{5i}RM * SM + \\ \beta_{6i}CFM * SM + \beta_{7i}RM * CFM * SM + \beta_{8i}SS + \varepsilon_i \ . \end{aligned} \quad (9)$$

Equation 9 presents the ANCOVA model that allows analyzing the main and interaction effects of reduction technique (RM), CF-method (CFM) and similarity measure (SM). Selection size (SS) is included as a covariate to control for different top-N selections. The inclusion of the covariate makes it possible to evaluate the F1-measures for every selection size in one analysis instead of repeating single ANOVA's for each top-N.

Best overall configuration. Analyzing the main effects of reduction technique, CF-method and similarity measure results in an evaluation of the best overall setting of each of the three algorithm variation parameters. Results show reduction method ($F_{4, 252}$ = 121.95, p = < .0001), CF-method ($F_{1, 252}$ = 154.59, p = < .0001) and similarity measure ($F_{2, 252}$ =368.09, p = < .0001) have a significant impact on the F1 performance of the model.

CA is the significantly best performing data reduction method, followed by NMF and SVD, which do not significantly differ from each another. LPCA performs significantly worse than mentioned reduction techniques, but significantly outperforms models based on the none reduced matrix. In terms of CF-method, item-based CF significantly outperforms the user-based method. Similarity measures also show a significant difference. Correlation and cosine similarity are performing similar, while both measures significantly outperform Jaccard's similarity.

Best configuration within datasets. The ANCOVA analysis is run separately for each of the 54 synthetic datasets with 299 different model configurations (observations). The analysis results in a champion model, based on F1 performance, for each synthetic dataset with different input characteristics. Tables 1, 2 and 3 visualize the results. Beneath the tables, models not significantly differing from the champions are listed.

CF-Method. In the tables, only item-based algorithms are listed as champions. This indicates that, regardless the characteristics, item-based CF outperforms user-based CF for each dataset.

Similarity Measure. The absence of models using Jaccard's measure in the tables indicates that these algorithms perform significantly worse than models with cosine and Pearson correlation similarity measures. For each dataset having a model with cosine (correlation) as champion model, the correlation (cosine) alternative does not significantly differ. This observation indicates that cosine and correlation similarity are statistically interchangeable. For 24 datasets, correlation constitutes the champion model. In 30 cases, cosine delivers the best results.

Table 1. Champion models for datasets with an **item/user ratio of 2** (= 500 users) in function of sparsity and distribution

	Distribution		
Sparsity	**Logistic**	**Linear**	**Uniform**
0.95	CA / Item / Corr [1]	CA / Item / Cos [1/3/4/5]	CA / Item / Cos [2/3/4/5/6]
0.96	CA / Item / Corr [1]	CA / Item / Cos [1/3/4/5]	CA / Item / Cos [2]
0.97	CA / Item / Corr [1]	CA / Item / Corr [1]	CA / Item / Corr [1/3]
0.98	CA / Item / Corr [1]	CA / Item / Corr [1/3]	CA / Item / Corr [1]
0.99	CA / Item / Corr [1/3]	CA / Item / Cos [2/3]	CA / Item / Corr [1/3]
0.995	CA / Item / Corr [1/3]	CA / Item / Corr [1/3/4]	CA / Item / Cos [2/3]

[1] CA / Item / Cos
[2] CA / Item / Corr
[3] NMF / Item / Cos, Corr
[4] None / Item / Cos, Corr
[5] SVD / Item / Cos, Corr
[6] LPCA / Item / Cos, Corr

Table 2. Champion models for datasets with an **item/user ratio of 1** (=1 000 users) in function of sparsity and distribution

	Distribution		
Sparsity	**Logistic**	**Linear**	**Uniform**
0.95	CA / Item / Corr [1/5]	CA / Item / Cos [2/4/5]	CA / Item / Cos [2/3/4/5/6]
0.96	CA / Item / Cos [2]	CA / Item / Cos [2/4/5]	CA / Item / Cos [2]
0.97	CA / Item / Cos [2]	CA / Item / Corr [1/5]	CA / Item / Corr [2]
0.98	CA / Item / Cos [2]	CA / Item / Cos [2]	CA / Item / Cos [2]
0.99	CA / Item / Corr [1]	CA / Item / Cos [2]	CA / Item / Corr [2/3]
0.995	CA / Item / Corr [1]	CA / Item / Cos [2/3]	CA / Item / Cos [2/3/4]

[1] CA / Item / Cos
[2] CA / Item / Corr
[3] NMF / Item / Cos, Corr
[4] None / Item / Cos, Corr
[5] SVD / Item / Cos, Corr
[6] LPCA / Item / Cos, Corr

Table 3. Champion models for datasets with an **item/user ratio of 0.5** (=2 000 users) in function of sparsity and distribution

	Distribution		
Sparsity	**Logistic**	**Linear**	**Uniform**
0.95	SVD / Item / Cos [1/2/5]	SVD / Item / Corr [1/2/4/5]	CA / Item / Cos [2/3/4/5/6]
0.96	CA / Item / Cos [2/5]	SVD / Item / Cos [1/2/4/5]	CA / Item / Cos [2/3/4/5/6]
0.97	CA / Item / Corr [1/5]	CA / Item / Cos [2/5]	CA / Item / Cos [2/5]
0.98	CA / Item / Cos [2]	CA / Item / Cos [2/5]	CA / Item / Corr [1/3]
0.99	CA / Item / Cos [2]	CA / Item / Cos [2]	CA / Item / Corr [1]
0.995	CA / Item / Corr [1]	CA / Item / Cos [2]	CA / Item / Corr [1/3]

[1] CA / Item / Cos
[2] CA / Item / Corr
[3] NMF / Item / Cos, Corr
[4] None / Item / Cos, Corr
[5] SVD / Item / Cos, Corr
[6] LPCA / Item / Cos, Corr

Data Reduction Method. In 51 out of 54 datasets the champion model is based on a CA reduced matrix, indicating this is the overall best reduction technique. For three datasets an algorithm based on the SVD decomposed matrix gives the best results. For these models CA configurations are not performing significantly worse, indicating that CA configurations are good alternatives for the SVD based models in these three cases. This logic also goes the other way round. Although CA reduction is the clear champion, in most cases other configurations are performing statistically similar, meaning these models can serve as good alternatives.

Item-based models using cosine or correlation similarity based on the *NMF reduced matrix* serve as good alternatives for CA configurations in 19 cases. Especially in datasets with a high

item/user ratio (2), NMF seems a good alternative. For 11 out of 18 datasets having an item/user ratio of 2, NMF is not performing significantly worse than CA.

Creating a model based on the *SVD reduced matrix* is a statistically good input basis for 18 out of 54 datasets. Lower sparsity, gives a higher chance that SVD is a good alternative for CA, especially in cases with a lower item/user ratio. The good performance of SVD in these cases is emphasized by being the champion in datasets with sparsity 0.95 (and 0.96 only for linear distribution) and an item/user ratio of 0.5.

Using the the full *none reduced matrix* as input is a good alternative in 12 cases. All cases represent linear or uniform distributions with mainly low sparsity levels (0.95 – 0.96). Watch out that for datasets with these characteristics most configurations based on item-based CF and cosine or correlation are statistically good alternatives.

LPCA reduced input matrices serve only a good alternative in 4 cases. For these 4 datasets, being cases with low sparsity and a linear or uniform distribution, all models using item-based CF and cosine or correlation similarity are performing significantly similar. This indicates that LPCA can be seen as the data reduction technique resulting in the least good accuracy results.

5 Discussion and Future Work

The presented prediction accuracy results, measured by the F1-statistic, indicate that item-based CF outperforms user-based CF and cosine and Pearson correlation give the same results but beat Jaccard based models. As data reduction technique CA gives the best results, but in some specific input characteristic cases NMF, SVD or the none reduced matrix can serve a good alternative.

Results presented in this study only take the F1-measure into account. To broaden the scope of the analysis, extra evaluation metrics will be analyzed in the next steps of the project. For prediction accuracy recall, precision and ROC-curve will be considered next to the F1-measure. Ranking accuracy will be measured using Kendalls Tau-C. Next to accuracy, item coverage, computation time and diversity (Intra-List Similarity) will be analyzed to get a complete outlook of each input characteristic – algorithm configuration combination.

To validate the results of this study, the discussed experimental design will be replicated on 10 real-life datasets of a leading European E-tailor.

References

1. Dias, M. B., Locher, D., Li, M., El-Deredy, W., Lisboa, P. J. G.: The Value of Personalised Recommender Systems to E-Business: A Case Study. In: Proceedings of the 2008 ACM Conference on Recommender Systems, pp. 291-294, ACM, Lausanne, Switzerland
2. Adomavicius, G., Tuzhilin, A.: Toward the Next Generation of Recommender Systems: A Survey of yhe State-of-the-Art and Possible Extensions. IEEE Trans. Knowl. Data Eng. 17, 6, 734-749 (2005)
3. Bobadilla, J., Ortega, F., Hernando, A., Gutierrez, A.: Recommender Systems Survey. Knowledge-Based Syst. 46, 109-132 (2013)
4. Su, X., Khoshgoftaar, T. M.: A Survey of Collaborative Filtering Techniques. Adv. in Artif. Intell. 2-21 (2009)
5. Kellar, M., Watters, C., Duffy, J.: Effect of Task on Time Spent Reading as an Implicit Measure of Interest. In: Bryans, J. B. (eds.): Asist 2004: Proceedings of the 67th Asis&T Annual Meeting: Managing and Enhancing Information: Cultures and Conflicts, Vol 41, pp. 168-175. Information Today Inc, Medford (2004)

6. Palanivel, K., Sivakumar, R.: A Study on Implicit Feedback in Multicriteria E-Commerce Recommender System. J. Electron. Commer. Res. 11, 2, 140-156 (2010)
7. Bodapati, A. V.: Recommendation Systems with Purchase Data. J. Mark. Res. 45, 77-93 (2008)
8. Pradel, B., Sean, S., Delporte, J., Guérif, S., Rouveirol, C., Usunier, N., Fogelman-Souli, F., Dufau-Joel, F.: A Case Study in a Recommender System Based on Purchase Data. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 377-385, ACM, San Diego, California, USA (2011)
9. Sarwar, B., Karypis, G., Kostan, J., Riedl, J. T.: Application of Dimensionality Reduction in Recommender Systems - A Case Study. In: WebKDD Workshop (2000)
10. Steck, H.: Item Popularity and Recommendation Accuracy. In: Proceedings of the Fifth ACM Conference on Recommender Systems, pp. 125-132, ACM, Chicago, Illinois, USA (2011)
11. Herlocker, J. L., Konstan, J. A., Terveen, K., Riedl, J. T.: Evaluating Collaborative Filtering Recommender Systems. ACM Trans. Inf. Syst. 22, 1, 5-53 (2004)
12. Khanna, R., Zhang, L., Agarwal, D., Chen, B. C.: Parallel Matrix Factorization for Binary Response. IEEE, New York (2013)
13. Breese, J. S., Heckerman, D., Kadie, C.: Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In: Proceedings of the Fourteenth conference on Uncertainty in Artificial Intelligence, pp. 43-52, Madison, Wisconsin, USA (1998)
14. Belohlavek, R., Vychodil, V.: Discovery of Optimal Factors in Binary Data via a Novel Method of Matrix Decomposition. J. of Comp. and Sys. Sc. 76, 1, 3-20 (2010)
15. Lee, S., Huang, J. Z., Hu, J. H.: Sparse Logistic Principal Components Analysis For Binary Data. Ann. Appl. Stat. 4, 3, 1579-1601 (2010)
16. Greenacre, M.: Correspondence Analysis in Practice. In Series: Keiding, N., Morgan, B., Speed, T., van der Heijden, P. (eds.): Interdisciplinary Statistics Series, Edition 2, Chapman & Hall/CRC, Boca Raton (2007)
17. Papagelis, M., Plexousakis, D.: Qualitative Analysis of User-Based and Item-Based Prediction Algorithms for Recommendation Agents. Eng. Appl. Artif. Intell. 18, 7, 781-789 (2005)
18. Choi, S.-S., Cha, S.-H., Tappert, C.: A Survey of Binary Similaruity and Distance Measures. J. on Systemics, Cybermetics and Informatics 8, 1, 5 (2010)
19. Candillier, L., Meyer, F., Fessant, F.: Designing Specific Weighted Similarity Measures to Improve Collaborative Filtering Systems. In: Proceedings of the 8th Industrial Conference on Advances in Data Mining: Medical Applications, E-Commerce, Marketing, and Theoretical Aspects, pp. 242-255, Leipzig, Germany (2008)
20. Aggarwal, C. C., Wolf, J. L., Wu, K.-L., Yu, P. S.: Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering. In: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 201-212, ACM, San Diego, California, USA (1999)
21. Tso, K. H. L., Schmidt-Thieme, L.: Empirical Analysis of Attribute-Aware Recommendation Algorithms with Variable Synthetic Data. Springer-Verlag, Berlin (2006)
22. Deshpande, M., Karypis, G.: Item-Based Top-N Recommendation Algorithms. ACM Trans. Inf. Syst. 22, 1, 143-177 (2004)
23. Popescul, A., Ungar, L. H., Pennock, D. M., Lawrence, S.: Probabilistic Models for Unified Collaborative and Content-Based Recommendation in Sparse-Data Environments. In: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, pp. 437-444, Seatle, Washington, USA (2001)
24. Leisch, F., Weingessel, A., Hornik, K.: On the Generation of Correlated Artificial Binary Data. Working Paper Series: Adaptive Information Systems and Modelling in Economics and Management Science, Vienna (1998)

An opinion mining Partial Least Square Path Modeling for football betting

Mohamed El Hamdaoui, Jean-Valère Cossu

LIA/Université d'Avignon et des Pays de Vaucluse
** 39 chemin des Meinajaries, Agroparc BP 91228, 84911 Avignon cedex 9, France
firstname.name@alumni.univ-avignon.fr

Abstract. In the last few years, football betting had known a large expansion in the world, using different ways to try to guess and predict the unknown in the sport. Every time, people try to prognosticate the results of matches using probabilistic, statistic and other methods to get the maximum benefits, especially with the emerging of betting websites. In this paper, we present an alternative approach, to state of the art probabilistic models, based on Partial Least Square Path Modeling (PLS-PM). We first show that the simple PLS model containing only statistical resources about each team are efficient to predict the team ranking at d+1 and this gives a state of the art prediction of match outcomes. We then take advantage of PLS ability of integrating complex and heterogeneous data to reach a practical model by including textual data, taken from tweets related to teams, that we previously classify by polarity using robust sentiment analysis in multiple languages. Another learning of our experiment is the role of the inner model in PLS when used for prediction purpose. Unlikely Bayesian networks, the latent variable used in the prediction need to be deeply inside the inner model and not considered as marginal outcomes, this to allow back and forth retro-propagation from multiple types of data. The main purpose of our work is to show that PLS-PM can be surprisingly efficient in predicting tournament outcomes for which temporal statistics and social network data are available if inference is based on central inner latent variables.

1 Introduction

Football is one of the most famous sport in the world, where betting on results is very popular. But it is not as easy as it looks, because even the football experts are not expert in prognostication as shown by [3]. Almost all systems and offices of betting use the probabilistic model to calculate odds linked to each part of bet. [5] gives in "Statistica Neerlandica", an example how scores are obtained using Poisson goals distribution. We want to prove, by these experiments, that probabilistic model is not the only way that allows to get efficient prognostications and intend to explore if a betting system based on correlation analysis of multiple and sparse data can be improved using different Partial Least Squares

** http://lia.univ-avignon.fr/

Path Modeling (PLS-PM). The remainder of the paper is structured as follows. First we present a simple model, based only on the ranking of teams. Then, we will use a model based on a few variables before combining these two solutions, and we will compare it with a probabilistic model, which is the rating used in different sport betting game. Finally, we will try to improve our PLS-PM model using text mining over twitter. Our work focus on on the leagues of France "League 1", England "Premier League", Spain "La Liga", and Italy "Serie A", each league is composed by 20 teams, and data for the first experimentation were obtained from the 14th, until the 23rd day of the league, and from the last 6 days for the model that contains textual data.

2 PLS-PM models

PLS-PM is a statistical method that allows studying and modeling complex relationships between observed (manifest) and latent variables. Data is analyzed like a structure made of blocks of manifest variables, and each block is summarized by a latent variable. This approach was developed by Herman WOLD during the 70s of the last century, when he presented PLS for the first time in 1979. But its popularity just started recently to increase in different domains. PLS-PM is formally represented by two sets of linear equation, the inner model and the outer one. The first model represents the relationships between latent variables, while the other model represents the relationships between a latent variable and its manifest variables. PLS-PM is a way to estimate parameters, it is used to find complex linear regressions, based on the latent and manifest variables, by calculating the solution of the general underlying model of multivariate PLS. For more details on how to manipulate manifest variables and these relations we refer the reader to [1] and [7]. We try to use different models to demonstrate the ability of PLS-PM concerning prognostication on football games. By these experiments we want to find the success indicator (SI) of each team. This will allow us to compare two teams and then, prognosticate which one will win. We tried to combine in one hand the method used in [6] based on simple statistics concerning number of goals scored and conceded, and on the other hand, the method used in [5], where he makes the difference between matches played at home and away. This way, we had to duplicate our PLS-PM model, in order to calculate both at home and away SI. Before starting, let us show you the type of data that we use, it is a table containing information for each team (in table 1).

Table 1. Resume of the 23rd day of league 1

Team	GSH	GSA	PSH	PSA	GCH	GCA	PCH	PCA	WH	WA	Rank
PSG	35	19	1.00	0.91	-5	-10	0.67	0.27	9	7	20
ASMonaco	19	19	0.91	0.92	-6	-10	0.54	0.42	8	6	19
Lille	16	8	0.91	0.50	-6	-8	0.64	0.58	8	4	18

This table shows some statistics on the top 3 ranked teams, collected after crawling and scraping some football websites.
GS (H/A): Number of goals scored at home (H) or away (A).
PS (H/A): Percentage of game where the team scored goal(s).
GC (H/A): Number of goals conceded.
PC (H/A): Percentage of game where the team conceded goal(s).
WM (H/A): Number of won games.

2.1 Model based on Ranking

We started by an inner model that contains three latent variables: Attack, Defense, and Success. This first experiment consists in realizing a baseline model, including only the ranking of the league as a manifest variable, which reflects the latent variable Success (as shown in figure 1). Keep in mind that we used that model twice respectively for away and at home Success Index (SI).

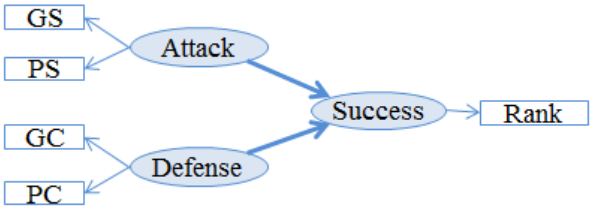


Fig. 1. Simple model based on Rank

Figure 1 represents the relation that exists between Attack, Defense, and Success, and our model is based on how each variable impacts other variables, so we can express our model in the following equation.

$$\text{Success}_{rank} = f(\text{Attack}, \text{Defense})$$

2.2 Model based on won matches

In this second experiment, we replaced the variable Ranking, by other latent variables like number of wins (as shown in figure 2).

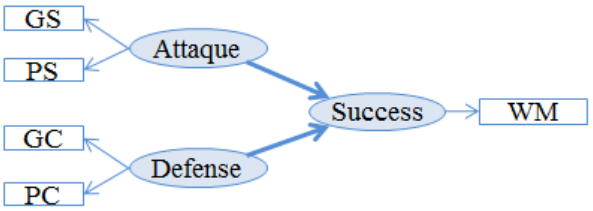


Fig. 2. Simple model based on number of won matches

In figure 2, Success is based this time on number of won games, so we can express it like:

$$\text{Success}_{won} = f(\text{Attack}, \text{Defense})$$

2.3 Model based on Ranking and won matches

The next experiment consists in mixing the previous models, to get this PLS-PM model (as described in figure 3):

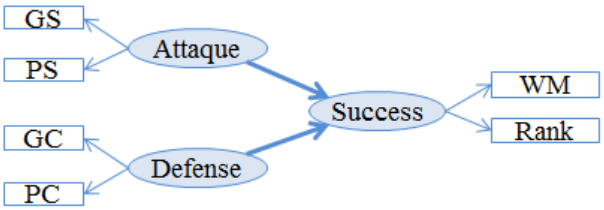


Fig. 3. PLS-PM model based on number of won matches and Rank

$$\text{Success}_{(rank,won)} = f(\text{Attack}, \text{Defense})$$

In each experiment, we used our results to verify the efficiency of the model, by prognosticating the results of matches, and we got results that gives the number of the right prognostication in ten games by day (figure 4).

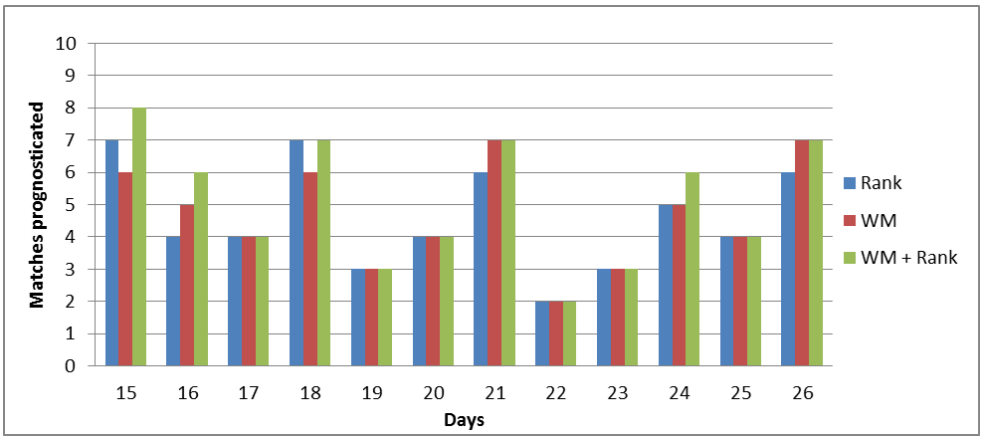


Fig. 4. Comparison between the previous models

Figure 4 compares the number of matches prognosticated by each model, from the 15^{th} week, until the 26^{th} one, it shows how, by using the model that combines ranking and won games, the number of correct matches prognosticated is greater than using models separately.

$$\text{Success day}_i(\text{WM} + \text{Rank}) \geq Max(\text{Success day}_i(\text{WM}), \text{Success day}_i(\text{Rank}))$$

As an interesting result it should be important to notice that the lowest values of correct matches prognosticated in one day is mostly due to the abundance of draws, which are difficult to predict (e.g. 22^{nd} week, there were 5 draws, which explain that our model predicted only 2 matches). So, we see that we can improve our prognostications by adding more manifest variables, such as number of points obtained in the last matches, as well as systems and websites of betting do based on previous matches.

2.4 Probabilistic model

Betting systems consider three values for each bet as it is represented in the following table (in table 2). Here, we consider the prognostication true if the result was a victory for Monaco, (1) means victory of the host team, because it had the highest probability of winning, but unluckily, this prognostication was false because the game finished by a draw which its odd was 3.34. Conversely, the prognostication in the second example was true, because Lyon, which has the lowest odd, won that game. So we had follow this way to calculate the results of other matches.

Table 2. Example for probabilistic model based on Odds

Host	Result	Visitor	(1)	(x)	(2)
Monaco	1 - 1	Lille	1.88	3.34	4.35
Nice	0 - 1	Lyon	4.48	3.54	1.82

(1): Host wins, (x): Draw, (2): visitor wins

Notice that odds are the inverse of probabilities values.

$$\text{Odds} = \frac{1}{p(\omega)}, \omega \in \Omega\{1, x, 2\}$$

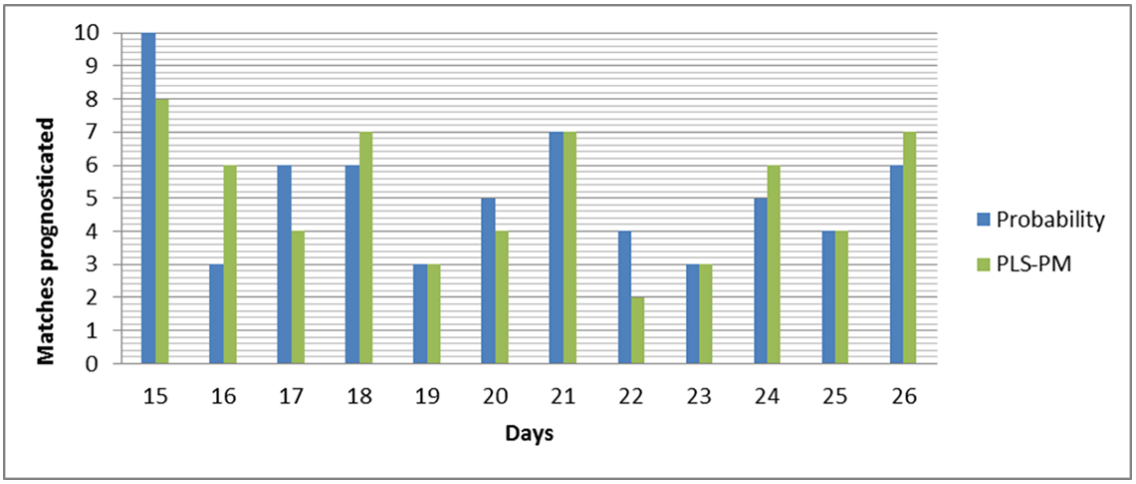


Fig. 5. Comparison between PLS-PM and probabilistic model

Figure 5 compares the probabilistic model, which is the state of the art, with our last PLS-PM model which assembles ranking and number of games won. These statistics between each model, prove how our PLS-PM model as efficient as the probabilistic model.

2.5 PLSPM with Twitter

The most interesting feature in PLS-PM, is that it can deal we a large variety of heterogeneous variables, provided that the correct model is set. Our next experiment consists in adding textual data in the model. As example we took the

reputation of each team on twitter, using twitteR package. It allows to collect tweets concerning each team. Then we used sentiment package in order to perform a 3-valued (positive, negative and neutral classes) polarity classification of these tweets using multiple languages opinion lexicon [2] [4]. As we mentioned, the difficulty relies in setting the correct model. For example, in a first trial we used a model which considers the reputation of a team as an element of its success (as described in figure 6). This first model downgraded all results as shown in (table 3).

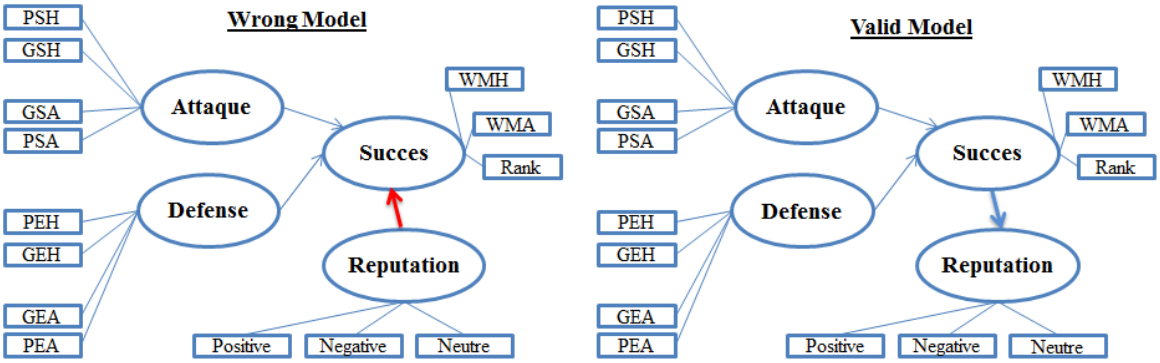


Fig. 6. The importance of choosing the right sense of relation between variables

Table 3. Result of wrong Model

Team	Success H	Success A
PSG	-1.7766	-1.7766
ASMonaco	0.8521	0.8521
Lille	-0.7359	-0.7359
OM	-0.3059	-0.3059
StdReims	2.2199	2.2199

In fact, it is not the reputation which affects success, but the opposite, so we changed the path matrix of the last model as described in figure 6.

$$\begin{cases} \text{Success} = f(\text{Attack}, \text{Defense}) \\ \text{Success} = f^{-1}(\text{Reputation}) \end{cases}$$

By next, we compared the result obtained by the probabilistic model with the improved PLS-PM model, and both are closer (table 4).

Figure 7 summarizes the last table, where we see that the difference between the number of matches predicted by probabilistic model, and PLS-PM model including text data (Twitter) is only 1 match in a total of 240, and 9 matches more than our basic PLS-PM model. This is an encouraging result, knowing that Probabilistic model, based on Poisson distribution, has a high performance of prediction because it integrates time series.

Table 4. Number of match predicted by each model

	Games	Results of matches	Model	Predictions	%
FR	70	1 - 30	Probability	38	0,54
	70	N - 19	PLSPM	33	0,47
	70	2 - 21	Twitter	37	0,53
ES	60	1 - 29	Probability	30	0,50
	60	N - 16	PLSPM	28	0,47
	60	2 - 15	Twitter	28	0,47
EN	50	1 - 23	Probability	30	0,60
	50	N - 8	PLSPM	27	0,54
	50	2 - 19	Twitter	29	0,58
IT	60	1 - 29	Probability	35	0,58
	60	N - 12	PLSPM	36	0,60
	60	2 - 19	Twitter	38	0,63

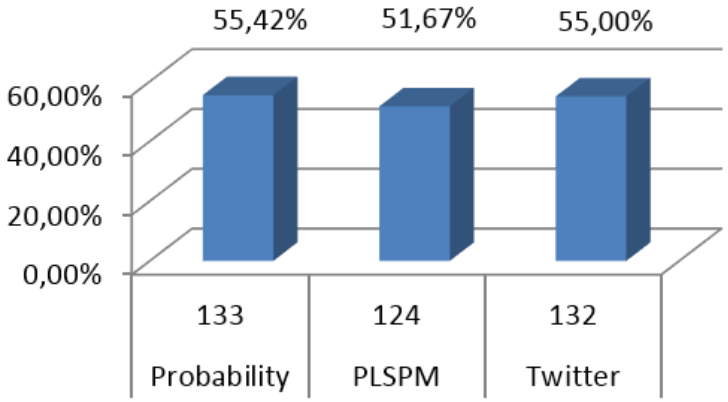


Fig. 7. Percentage of match predicted by each model

The most important thing to notice, concerning PLS-PM model including text data, is that it improves the number of predicted drawn matches, and this is what explains how it outperforms PSL-PM basic model, especially in the case of teams with similar rankings. The method considering that a game will finish by a draw is resumed in the next formula :

$$\text{Draw} \Leftrightarrow \|\text{Success at Home}–\text{Success Away}\| \leq 0.02$$

Tweets give in fact the latest information concerning one team like when a player has been being injured, or excluded and would not play next game. In addition, it is significant to know if a team is well supported or not, or it is in a good financial situation.

3 PLS-PM and Bootstrapping

Our last experiment consisted in comparing 3 PLS-PM models (as described in figure 8), where we applied the bootstrapping method to obtain information about the variability of our different estimated variables.

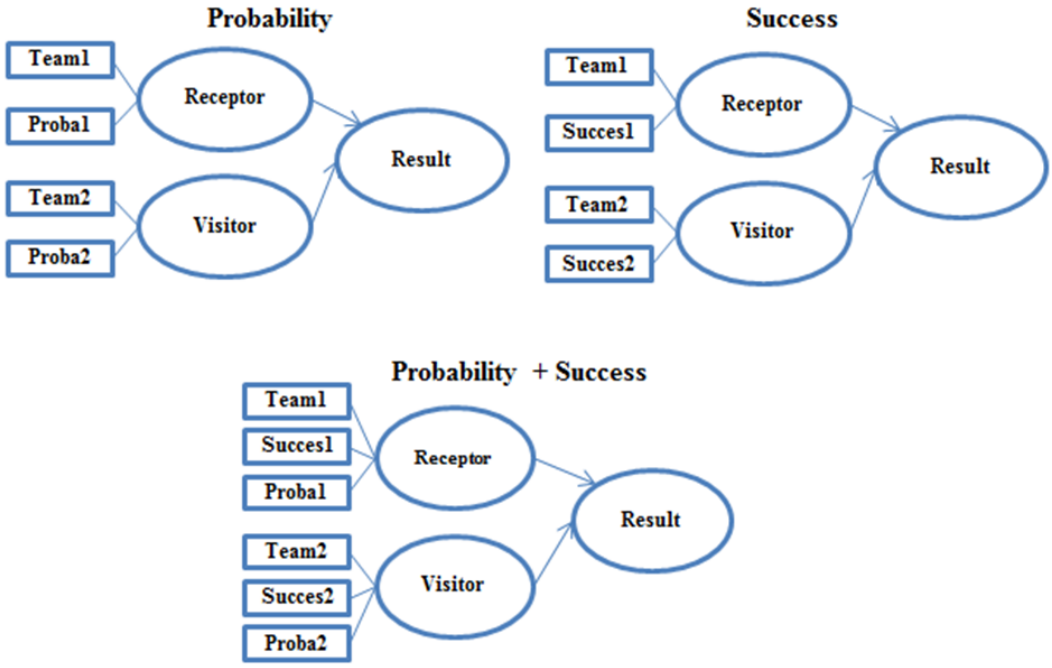


Fig. 8. Different models used

For that, we used in our model, in a first time, only variables containing the probability of winning for each team in such game. Then we repeated the same experiment by introducing variables containing this time the success of every team. Finally we mixed those models considering both variables. We applied those model on a collection composed of a variety of 240 games played this season in English premier league from the eleventh days, until the thirtieth one (table 5). We ignored the ten first days because the amount of data is not sufficient to compute success by using PLS-PM, so the results during those first journeys were not reliable to estimate the effectiveness of the model.

Table 5. Extract showing the kind of data we used in this experimentation

Team1	Suc1	Prob1	Team2	Suc2	Prob2	Day	Result
WestHam	-1.2342	1.94	Cardiff	-0.3484	4.02	11	1
WBA	-0.4313	2.29	Southampton	0.7968	3.12	11	-1
Swansea	-0.8592	4.37	ManUtd	0.7436	1.86	11	-1
Sunderland	-1.0079	2.21	Fulham	-0.4017	3.30	11	-1
Norwich	-0.3735	3.17	Everton	1.03976	2.31	11	0
Liverpool	1.4007	1.39	Stoke	-0.82890	8.67	11	1

Table 5 represents success at home (Suc1) and away (Suc2), we computed before, by applying the PLS-PM model and the probability of winning that match. The variable Result equals 1 when the receptor won, 0 when the match finished by a draw, and -1 when the visitor won. By validating our model using bootstrapping, we got the following results of model respectively based on probability, on success, and on both of them:

Table 6. Results of Bootstrapping

Model	Original	Mean.Boot	Std.Error	perc.025	perc.975
Probability	0.2155	0.2271	0.0456	0.1363	0.3158
PLS-PM	0.3384	0.3467	0.0465	0.2514	0.4171
Probability + PLS-PM	0.3010	0.3133	0.0463	0.2186	0.4011

As we can see in table 6, the lowest original value of R square obtained, depending on result, is when we used probability (21%), it means that the performance of the model based on probability is low. Poisson distribution depends on short term time periods and in this experiment, it loses its advantage, because we considered a long duration. Therefore, when considering the overall performance of a team over long periods, highest values of original R square rely on the success scores based on PLS-PM model.

4 Conclusion

Thanks to our different experimentations, we have shown that PLS-PM is a efficient method for prediction and prognostication. Starting by choosing the good Latent and manifest variables, then creating adequate relations between those variables, allowed us to get a model very close to the probabilistic one, which is considered to be the highest performance model in the state of the art. The main interest of PLS-PM is that we are able to introduce heterogeneous data in our model, and that is what permitted us to predict some draws by adding text data extracted from twitter. Draw case is very difficult, even the probabilistic model is not able to properly predict this kind of result. Our investigations showed that we cannot reach a good result without a lot of information. Nevertheless, such a result is known to be the weak point of PSL-PM, and we clearly observed it when we were not able to predict any match before the 10th journey. It means that, due to the lack of data, we could not predict 100 games. But even the probabilistic model has the same problem, despite that it does not need such an important mass of data as PLS-PM to predict games. There is a room of improvement in our results, and we intend to exceed the performance of probabilistic model as future work by trying to include more information that influences a football game to see how much game we can predict properly.

References

1. Henseler, J. *On the convergence of the partial least squares path modeling algorithm* Published online: 1 August 2009, DOI 10.1007/s00180-009-0164-x

2. Hu M. and Liu B. *Mining and Summarizing Customer Reviews.* Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), Aug 22-25, 2004, Seattle, Washington, USA,
3. Khazaal Y., Chatton A., Billieux J., Bizzini L., Monney G., Fresard E., Thorens G., Bondolfi G., El-Guebaly N., Zullino D., Khan R. *Effects of expertise on football betting* Substance Abuse Treatment, Prevention, and Policy 2012 7:18.
4. Liu B., Hu M. and Cheng J. *Opinion Observer: Analyzing and Comparing Opinions on the Web.* Proceedings of the 14th International World Wide Web conference (WWW-2005), May 10-14, 2005, Chiba, Japan.
5. Maher M. J. *Statistica Neerlandica Modelling association football scores* Published online: 29 APR 2008, DOI: 10.1111/j.1467-9574.1982.tb00782.x
6. Sanchez G. *PLS path modeling with R*
7. Wold. S., Eriksson L., Trygg J., Kettaneh N. *The PLS method partial least squares projections to latent structures and its applications in industrial RDP (research, development, and production)* Submitted version, June 2004

Parallel Learning Algorithm for Large-Scale Regression with Additive Models

Valeriy Khakhutskyy[1**] and Markus Hegland[2**]

[1] Institute for Advanced Study, Technische Universität Munchen, Lichtenbergstrasse 2a, D-85748 Garching, Germany,
khakhutv@in.tum.de

[2] Centre for Mathematics and its Applications, Mathematical Sciences Institute Australian National University, Canberra, ACT 0200, Australia.

Abstract. We present a novel parallel algorithm for training additive regression models. The approach relates to a number of other methods including the backfitting algorithm and alternating direction method of multipliers. However, we extend the scope of possible applications to include any ANOVA-type decomposition or an ensemble of random subspace projection models, and we show how the algorithm can be parallelised for distributed systems.
The experimental results illustrate the convergence and scaling properties of the algorithm on real and synthetic data.

Keywords: backfitting, ADMM, additive models, parallelisation, regression, BiCGStab, ensemble learning

1 Introduction

Machine learning applications continuously grow in size and dimensionality as with the growing interest in data science and increasing computational power more effort is spent on data gathering and analysis. At the same time, the asymptotics of the underlying function approximation algorithms remain unchanged.

With respect to the number of data entries, the linear basis expansion models, i.e. splines or sparse grids [1], show linear complexity of computation and storage. This is already optimal for a regression problem without any additional assumptions. Hence, further research attempts to minimise the complexity constants with better implementations, parallelisation, or sub-sampling heuristics [2–4].

With respect to dimensionality, the complexity of most approximation algorithms suffers from the "curse of dimensionality": a term coined by Richard Bellman in 1961, which denotes that the computation and storage complexity for a fixed approximation accuracy depends exponentially on the dimensionality

** With the support of the Technische Universität München Institute for Advanced Study, funded by the German Excellence Initiative (and the European Union Seventh Framework Programme under grant agreement n 291763).

[5]. The results from information-based complexity suggest that the curse of dimensionality can be avoided only if a problem possesses a special structure, i.e. smoothness or separability, that can be exploited by an algorithm [6, 7].

The methods that exploit this special structure include ANOVA decomposition, additive models [8], sparse grids, and random forests. Additive models are well established in statistics and thoroughly studied in the literature [9, 10]. Similar concepts are popular in the machine learning community. For example, recent developments apply new optimisation methods [11] and parallelisation paradigms [12]. Ensembles of random subspace projection models were successfully used for classification and regression [13]. Moreover, estimation of additive models is an integral part of generalised additive models – a more powerful but also a more computationally expensive representation concept [9].

In this paper, we discuss an approach for large-scale regression based on additive models and a parallel BiCGStab algorithm for optimisation. The method combines flexibility of choosing from a large number of suitable parametric and non-parametric models and an optimisation algorithm with fast convergence and a potential for efficient parallelisation both in data size and dimensionality. It is used for training additive models, but this restriction can be relaxed to include ensembles of subspace projection models. We extend the fitting method with data partitioning using kd-trees and orthogonalisation, which improves data locality for additive models and has a potential to adapt to manifolds.

The remainder of this paper is organised as follows: Section 2 introduces the necessary theoretical background. In Section 3 we suggest a Krylov-space method for solution of normal equations and show how the problem structure can be exploited for efficiency and parallelisation. We illustrate convergence and scalability of the new method using benchmark problems in Section 4. Finally, we conclude with a discussion of the results and provide an overview of future work in Section 5.

2 Theoretical Background

We consider a dataset of the form $(\mathbf{t}^{(1)}, y^{(1)}), \ldots, (\mathbf{t}^{(N)}, y^{(N)})$ with input variables $\mathbf{t}^{(i)}$ and target variables $y^{(i)}$. Ridge regression is often used to find an approximation of the input-target mapping f in a function space V:

$$\min_{f \in V} \tfrac{1}{2} \sum_{i=1}^{N} (f(\mathbf{t}^{(i)}) - y^{(i)})^2 + \tfrac{1}{2} \lambda \|Df\|_2^2, \tag{1}$$

with a positive regularisation parameter λ and some smoothness operator D.

As mentioned above, (1) suffers from the curse of dimensionality so that only the problems with a moderate number of input dimensions can be handled. Approaches that do not face the curse are based on decomposition of the space V into a sum of simpler function spaces

$$V = V_1 + \ldots + V_n. \tag{2}$$

For example, in the context of ANOVA, the decomposition of a function f with a d-dimensional input has the form

$$f(t) = f_0 + \sum_{j=1}^{d} f_j(t_j) + \sum_{1 \le i < j \le d} f_{i,j}(t_i, t_j) + \sum_{1 \le i < j < k \le d} f_{i,j,k}(t_i, t_j, t_k) + \dots ,$$

where t_j stands for the j-th component of the data point $\mathbf{t}$.

More general, the decomposition (2) may be not exact. For example, if we limit the number of terms V_j and restrict them to specific low dimensional function spaces, we obtain an ensemble of subspace projection models. Hereafter we consider only the 1-dimensional terms f_j, but extension to other function spaces is straightforward.

While it is not assumed that the V_j are linearly independent, we assume that the smoothness operator is consistent with the decomposition of V, such that the optimisation problem assumes the form

$$\min_{f_0 \in \mathbb{R}, f_1 \in V_1, \dots, f_d \in V_d} \frac{1}{2} \sum_{i=1}^{N} (f_0 + \sum_{j=1}^{d} f_j(t_j^{(i)}) - y^{(i)})^2 + \sum_{j=1}^{d} \frac{\lambda_j}{2} \|D_j f_j\|_2^2. \tag{3}$$

Many models for representation of f_j, for example, linear basis expansion models, can be cast in terms of linear algebra. Let vector $\mathbf{x}$ denote parameters of $f(\mathbf{t})$ with respect to some generating system, e.g. coefficients of a polynomial model. Furthermore, let $\mathbf{Ax}$ be a vector of function values $f(\mathbf{t}^{(i)})$ and $\mathbf{y}$ – the vector of target values $y^{(i)}$. Taking into account the residual $\mathbf{r}$, we obtain

$$\underbrace{[\mathbf{A}_1 \dots \mathbf{A}_d]}_{\mathbf{A}} \underbrace{\left[\mathbf{x}_1^T \dots \mathbf{x}_d^T\right]^T}_{\mathbf{x}} = \mathbf{y} - \mathbf{r} \tag{4}$$

with $\mathbf{x}_j \in \mathbb{R}^{m_j}$, $\mathbf{A}_j \in \mathbb{R}^{N \times m_j}$, $\mathbf{y}, \mathbf{r} \in \mathbb{R}^N$, $\mathbf{x} \in \mathbb{R}^m, \mathbf{A} \in \mathbb{R}^{N \times m}$, and $m := m_1 + \dots + m_d$. The problem (3) can now be written as

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{Ax} - \mathbf{y}\|_2^2 + \frac{\lambda}{2} \mathbf{x}^T \mathbf{D} \mathbf{x} \tag{5}$$

with $\mathbf{D}$ a block diagonal matrix, which can be partitioned in a structure compatible with that of $\mathbf{x}$. Often, $\mathbf{D}$ is just an identity matrix. In order to minimise (5), one needs to solve the normal equations

$$(\mathbf{A}^T \mathbf{A} + \lambda \mathbf{D}) \mathbf{x} = \mathbf{A}^T \mathbf{y}. \tag{6}$$

If we substitute (4) into (6) we obtain the system

$$\begin{bmatrix} \mathbf{A}_1^T \mathbf{A}_1 + \lambda \mathbf{D}_1 & \mathbf{A}_1^T \mathbf{A}_2 & \cdots & \mathbf{A}_1^T \mathbf{A}_d \\ \mathbf{A}_2^T \mathbf{A}_1 & \mathbf{A}_2^T \mathbf{A}_2 + \lambda \mathbf{D}_2 & \cdots & \mathbf{A}_2^T \mathbf{A}_d \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_d^T \mathbf{A}_1 & \mathbf{A}_d^T \mathbf{A}_2 & \cdots & \mathbf{A}_d^T \mathbf{A}_d + \lambda \mathbf{D}_d \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_d \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1^T \mathbf{y} \\ \mathbf{A}_2^T \mathbf{y} \\ \vdots \\ \mathbf{A}_d^T \mathbf{y} \end{bmatrix}. \tag{7}$$

We are particularly interested in problems where the solution of (7) is used to determine the predicted values $\hat{\mathbf{f}} = \mathbf{Ax} = \mathbf{y} - \mathbf{r}$, which in view of (4) can be written as $\hat{\mathbf{f}} := \mathbf{f}_1 + \dots + \mathbf{f}_d,$ with $\mathbf{f}_j = \mathbf{A}_j \mathbf{x}_j$, $j = 1, \dots, d$. If we multiply every

row block j of (7) by $\mathbf{A}_j(\mathbf{A}_j^T\mathbf{A}_j + \lambda\mathbf{D}_j)^{-1}$ from the left and introduce

$$\mathbf{S}_j := \mathbf{A}_j(\mathbf{A}_j^T\mathbf{A}_j + \lambda\mathbf{D}_j)^{-1}\mathbf{A}_j^T \quad j = 1, \ldots, d, \tag{8}$$

we obtain equations of the form

$$\underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{S}_1 & \mathbf{S}_1 & \ldots & \mathbf{S}_1 \\ \mathbf{S}_2 & \mathbf{I} & \mathbf{S}_2 & \ldots & \mathbf{S}_2 \\ \mathbf{S}_3 & \mathbf{S}_3 & \mathbf{I} & \ldots & \mathbf{S}_3 \\ \vdots & \vdots & & \ddots & \vdots \\ \mathbf{S}_d & \mathbf{S}_d & \mathbf{S}_d & \ldots & \mathbf{I} \end{bmatrix}}_{\mathbf{S}} \underbrace{\begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_3 \\ \vdots \\ \mathbf{f}_d \end{bmatrix}}_{\mathbf{f}} = \begin{bmatrix} \mathbf{S}_1\mathbf{y} \\ \mathbf{S}_2\mathbf{y} \\ \mathbf{S}_3\mathbf{y} \\ \vdots \\ \mathbf{S}_d\mathbf{y} \end{bmatrix}. \tag{9}$$

Following the tradition from statistics, we call $\mathbf{S}_j$ a *smoothing matrix*. The normal equation (9) is the one we are interested in solving instead of (7). In Section 4 we motivate this preference.

A popular choice for solving (5) on distributed systems is alternating direction method of multipliers (ADMM) [9]. In fact, one can show that if the penalisation parameter is equal to $1/\lambda$, the ADMM update steps would correspond to a sequence of a Jacobi-iteration for solution of (7) followed by an update of the auxiliary variable $\bar{\mathbf{z}}$. This motivates the comparison of the two methods in Section 4. We have to note, however, that this relationship does not transfer if the penalty or regularisation term is not quadratic.

3 Fitting Methods

Traditionally, problem (9) is solved using the *backfitting algorithm* [10] in sequence of a blocked Gauss-Seidel iterations:

$$\textbf{for } j = 1 \text{ to } d \textbf{ do: } \mathbf{f}_j = \mathbf{S}_j \left(\mathbf{y} - \sum_{k \neq j} \mathbf{f}_k \right).$$

Buja et al. have shown that the convergence speed of the backfitting algorithm heavily depends on the magnitude and distribution of the smoothers' eigenvalues which are significantly smaller than 1. The error terms corresponding to the eigenvectors with eigenvalues near 1 (e.g. constant, linear and low-frequency) would not be eliminated at all by the algorithm [10].

For regression models, however, it is not unusual that a number of large eigenvalues are clustered around 1.0. Moreover, a typical distribution of the eigenvalues of the matrix $\mathbf{S}$ contains a single large eigenvalue close to d, followed by a cluster of small eigenvalues.

At this point we suggest to use a BiCGStab-based Krylov method [14] for the solution of (9). Not only is it better suited for system matrices with clustered eigenvalues, it would eliminate the error components that are problematic for backfitting. We refer our reader to the original paper by van der Vorst [14] for a review of the original algorithm, and adopt its notation in this section.

A number of improvements for BiCGStab were suggested in the literature that would reduce the communication on distributed systems [15].

We derived a BiCGStab algorithm for fitting additive models that can be parallelised in the number of models and data entries. We can improve the scalability of the original algorithm by exploiting the special structure of the system matrix $\mathbf{S}$: the j-th block of the result of matrix-vector multiplication can be calculated as

$$\mathbf{v}_j = \mathbf{p}_j + \mathbf{S}_j(\sum_{i \neq j} \mathbf{p}_j) = \mathbf{p}_j + \mathbf{S}_j(\sum_{i=1}^{d} \mathbf{p}_i - \mathbf{p}_j).$$

Similar to the ideas in [15], we further reduce the communication overhead by postponing the calculation of the scalar products until the aggregation of the results of matrix-vector multiplications is necessary.

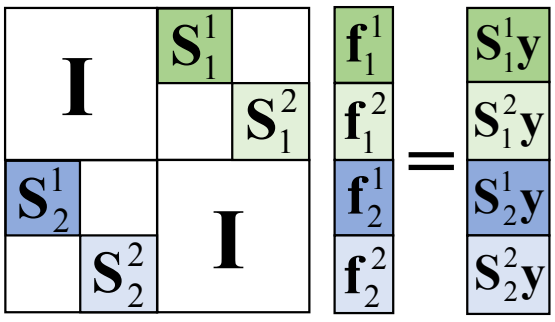
Algorithm 1 presents the new fitting procedure. The vectors with the subscript "Σ", e.g. $\mathbf{t}_\Sigma$, represent the sum of individual vectors, e.g. $\sum_i \mathbf{t}_i$. The function `Allreduce` is known from MPI programming and is similar to the reduce operation in the Map-Reduce framework. It denotes an application of an operation (in our case SUM) component-wise to the first argument across all processes and stores the results in the second argument. The scalar products in the second argument of the `Allreduce` functions in Lines 10 and 20 stand for variables containing the corresponding scalar products. In these calls a vector and scalar products are joined into a single memory segment to reduce communication.

Depending on the application requirements, two alternatives of data parallelism can be suggested: On the one hand, a simple SIMD parallelisation of the linear algebra operations can be performed using an appropriate BLAS implementation (MKL, OpenBLAS, etc) on shared memory systems. On the other hand, we suggest to partition data using kd-trees and to decouple individual problems as illustrated on Fig. 1. While we may loose smoothness of the solution across the partition borders, the decomposition completed with orthogonalisation is known to adapt to the internal data manifolds [16] and improves the data alignment important for additive models.

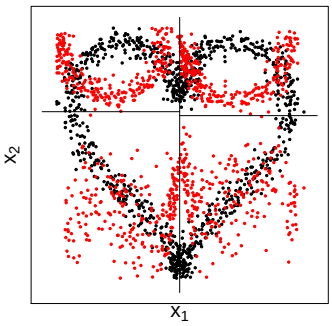
In this case of domain decomposition, communication in Lines 10 and 20 can be performed in three steps as illustrated on Fig. 2 to minimise the amount of sent and received data. In the first step, we calculate the sum of large vectors $[\mathbf{s}_j^T\mathbf{r}_j^0, \mathbf{t}_j^T\mathbf{s}_j, \mathbf{t}_j^T\mathbf{t}_j, \mathbf{t}_j^T\mathbf{r}_j^0, \mathbf{t}_j]$ and $[\mathbf{v}_j^T\mathbf{r}_j^0, \mathbf{v}_j]$ among the processors that share the same data partition (Fig. 2a). At this stage the size of the communicated vectors is basically the number of points in the partition. In the second step, only the scalar product results $[\mathbf{s}_j^T\mathbf{r}_j^0, \mathbf{t}_j^T\mathbf{s}_j, \mathbf{t}_j^T\mathbf{t}_j, \mathbf{t}_j^T\mathbf{r}_j^0]$ and $[\mathbf{v}_j^T\mathbf{r}_j^0]$ are reduced between the root processes of individual partitions (Fig. 2b). While the communication between partitions is more expensive in distributed clusters, we need to communicate only a small constant number of values, which is done very efficiently in the common MPI implementations. In the last step, we broadcast the updated values from the roots to all processes in the partitions (Fig. 2c).

Algorithm 1 Parallel BiCGStab Algorithm: code for processor $j, 0 \leq j \leq n-1$

Input: $\mathbf{S}_j$ smoothing matrix, $\mathbf{y}$ target vector
Output: $\mathbf{f}_j$ predictions of the function $f_j(x)$ at the data points
$\mathbf{r}_j^0 = S_j\mathbf{y}; \mathbf{r}_j = \mathbf{r}_j^0$
`Allreduce`$([\mathbf{r}_j, \mathbf{r}_j^T\mathbf{r}_j], [\mathbf{r}_\Sigma, \rho^{\text{new}}], \text{SUM})$
$\alpha = \omega = 1$; $\rho^{\text{old}} = \beta = \rho^{\text{new}}$
$\mathbf{f}_j = \mathbf{t}_j = \mathbf{v}_j = \mathbf{v}_\Sigma = \mathbf{p}_j = \mathbf{p}_\Sigma = \mathbf{s}_j = \mathbf{s}_\Sigma = \mathbf{0}$
while not converged **do**
  if iteration > 0 **then**
    $\rho^{\text{old}} = \rho^{\text{new}}$
    `Allreduce`$([\mathbf{s}_j^T\mathbf{r}_j^0, \mathbf{t}_j^T\mathbf{s}_j, \mathbf{t}_j^T\mathbf{t}_j, \mathbf{t}_j^T\mathbf{r}_j^0, \mathbf{t}_j], [\mathbf{s}^T\mathbf{r}^0, \mathbf{t}^T\mathbf{s}, \mathbf{t}^T\mathbf{t}, \mathbf{t}^T\mathbf{r}^0, \mathbf{t}_\Sigma], \text{SUM})$
    $\omega = \frac{\mathbf{t}^T\mathbf{s}}{\mathbf{t}^T\mathbf{t}}; \rho^{\text{new}} = \mathbf{s}^T\mathbf{r}^0 - \omega\mathbf{t}^T\mathbf{r}^0; \beta = \frac{\rho^{\text{new}}}{\rho^{\text{old}}} \cdot \frac{\alpha}{\omega}$
    $\rho^{\text{old}} = \rho^{\text{new}}$
    $\mathbf{r}_j = \mathbf{s}_j - \omega\mathbf{t}_j; \mathbf{r}_\Sigma = \mathbf{s}_\Sigma - \omega\mathbf{t}_\Sigma$
    $\mathbf{f}_j = \mathbf{f}_j + \omega\mathbf{s}_j$
    check convergence on $\mathbf{r}$
  end if
  $\mathbf{p}_j = \beta(\mathbf{p}_j - \omega\mathbf{v}_j) + \mathbf{r}_j$
  $\mathbf{p}_\Sigma = \beta(\mathbf{p}_\Sigma - \omega\mathbf{v}_\Sigma) + \mathbf{r}_\Sigma$
  $\mathbf{v}_j = \mathbf{p}_j + S_j(\mathbf{p}_\Sigma - \mathbf{p}_j)$
  `Allreduce`$\big([\mathbf{v}_j^T\mathbf{r}_j^0, \mathbf{v}_j], [\mathbf{v}^T\mathbf{r}^0, \mathbf{v}_\Sigma], \text{SUM}\big)$
  $\alpha = \rho^{\text{old}}/\mathbf{v}^T\mathbf{r}^0$
  $\mathbf{s}_j = \mathbf{r}_j - \alpha\mathbf{v}_j; \mathbf{s}_\Sigma = \mathbf{r}_\Sigma - \alpha\mathbf{v}_\Sigma$
  $\mathbf{f}_j = \mathbf{f}_j + \alpha\mathbf{p}_j$
  check convergence on $\mathbf{s}$
  $\mathbf{t}_j = \mathbf{s}_j + S_j(\mathbf{s}_\Sigma - \mathbf{s}_j)$
end while

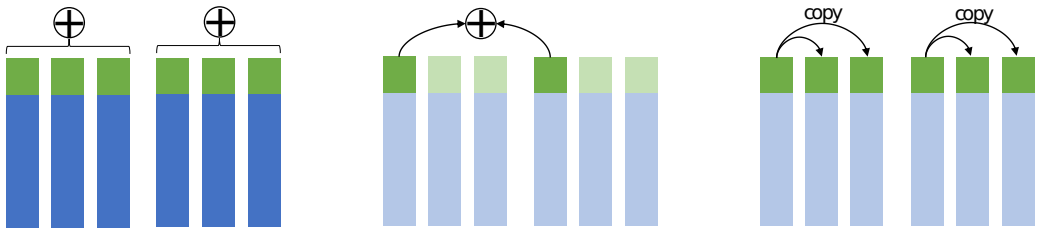


(a) Task and data decomposition of Equation (9).



(b) Data partitioning and transformation with kd-trees.

Fig. 1: Parallelisation of the fitting algorithm using task and data decomposition with kd-trees. On the left: Transformation of Equation (9). The coloured blocks are computed simultaneously. On the right: Kd-tree based partitioning (solid lines) of data with a low-dimensional manifold (black dots) and its orthogonalisation in individual partitions (red dots).



(a) Step 1. (b) Step 2. (c) Step 3.

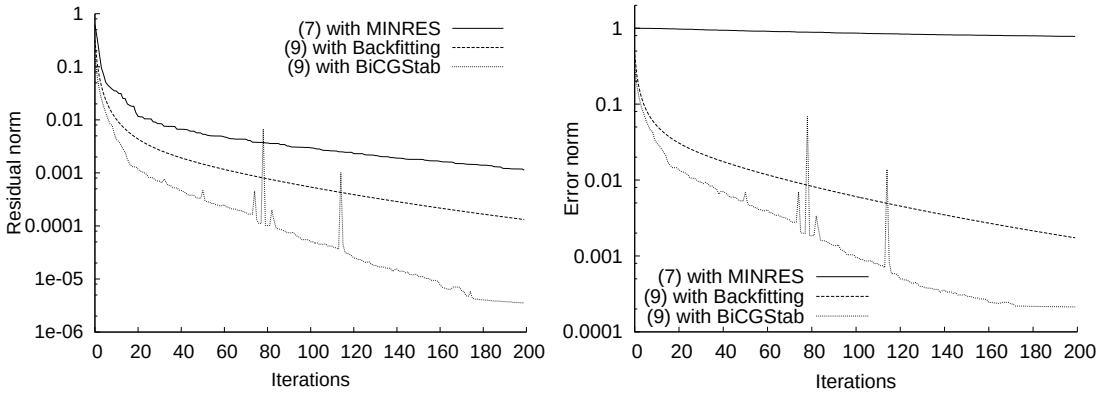
Fig. 2: Three-steps communication model for Lines 10 and 20 of Algorithm 1 with data partitioning.

4 Results

In this section we show results that highlight the properties of the presented fitting algorithm. We begin by comparing the convergence of different problem formulations and fitting methods. Then we discuss the impact of data partitioning and normalisation as well as show strong scaling results of the distributed parallel implementation of Algorithm 1.

To motivate the use of the smoothing matrix formulation and normal equations of the form (9) instead of (7), we begin by illustrating the convergence speed of the same problem in these two formulations.

The synthetic dataset used in this experiment was generated from a linear model with random coefficients and a small additive noise term. The dataset has 100 dimensions, whereas only 10 of them are informative.



(a) Residual (b) Error

Fig. 3: Comparison of problem formulations (7) and (9) for minimisation of residual and error of a synthetic 100-dimensional dataset from model. Fitting of 1,000 data points was performed using regression cubic splines with $\lambda = 10^{-7}$ and dof=10.

Figure 3 illustrates the relative residual and prediction error norms. One can see the superiority of the problem formulation (9) both by using the classical backfitting algorithm with Gauss-Seidel iterations and by using the BiCGStab

method. While the residual in the formulation (7) decreases, this does not considerably affect the error of the resulting additive model, which is our main goal.

We observed that the convergence of the backfitting algorithm is usually more stable but much slower. As BiCGStab does not directly minimise the error norm, the spikes as seen on Fig. 3 are not uncommon, although the algorithm would usually continue to converge after a spike.

The comparison between ADMM and BiCGStab is presented on Fig. 4. We use the Sloan Digital Sky Survey dataset in Data Release 5 (SDSS) [17] to predict the photometric redshift of galaxies based on 6 cosmological parameters [2]. Both fitting methods are comparable, although the BiCGStab-based method exhibits a faster convergence at the beginning. A parallel implementation of the backfitting algorithm with red-black Gauss-Seidel iterations would not converge.

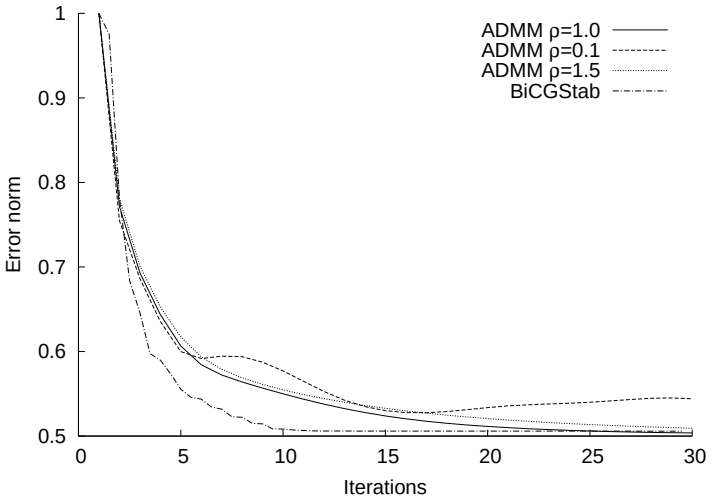


Fig. 4: Training error using ADMM and BiCGStab for regression on SDSS with 60,000 entries and random subspace projection ensemble with five 3-dimensional sparse grids with level 3 and $\lambda = 10^{-6}$.

As an example of a high-dimensional prediction problem we consider the year prediction task on the million song dataset [18] with 90 features and 463,715 examples in the training dataset. We use the first 88 features which is more convenient to split across a different number of processors. We use linear models as smoothing operators $\mathbf{S}_j$, however the extension to other linear basis expansion models is straightforwards.

As we described in the previous section, partitioning the dataset into subdomains and normalisation of individual partitions can improve the performance and accelerate the convergence of additive models. Figure 5 compares the relative residual norm and the mean prediction error for the same model with and without the orthogonalisation step. With orthogonalisation the algorithm converges after the first step.

Figure 6a illustrates the strong scaling of the algorithm for execution of 45 iterations using between 2 and 256 processors. There is a trade-off between the reduction of computation time and the increase of communication overhead (Fig. 6b). For up to 8 processors every process receives all examples, starting from 16 processors we introduce data partitioning, which is responsible for a jump in communication overhead. An application of more computationally intensive basis

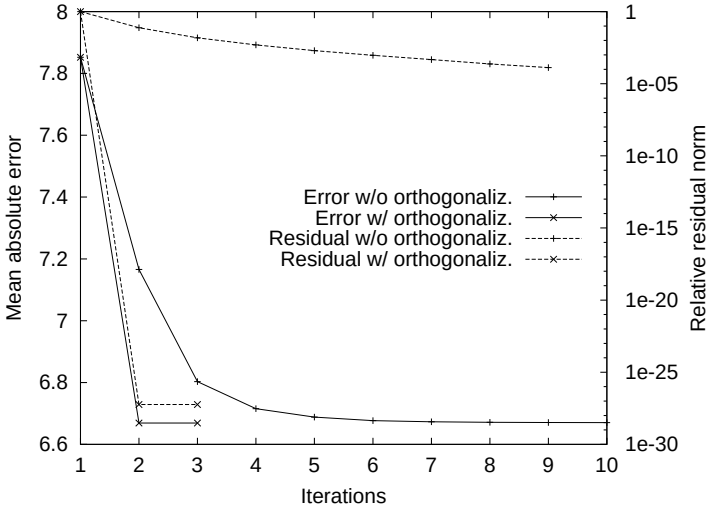


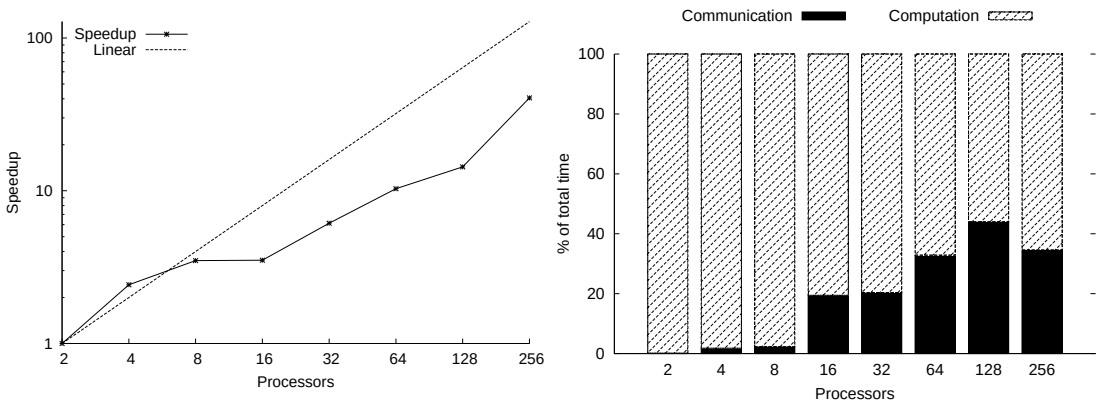
Fig. 5: Comparison of the relative residual norm and the mean prediction error for the same model with and without the orthogonalisation step on the million song dataset.

functions or smoothing kernels would improve the computation/communication ratio.

5 Conclusion

We presented a new approach for fitting additive models using BiCGStab algorithm that can be used for large-scale regression problems. While the convergence of the BiCGStab method cannot be proved theoretically, it usually works well in practice. It converges fast and can be efficiently parallelised for distributed computer architecture.

The method relates to ADMM for distributed optimisation and shows comparable convergence speed. Our future work will include the development of preconditioning methods to stabilise and accelerate the convergence.



(a) Strong scaling

(b) Computation/communication

Fig. 6: Scaling properties of Algorithm 1. On the left: strong scaling results for the million song dataset with 88 features and 463,715 examples. On the right: the computation-to-communication relationship for different number of processors.

References

1. J. Garcke, M. Griebel, and M. Thess, "Data mining with sparse grids," *Computing*, vol. 67, no. 3, pp. 225–253, 2001.
2. D. Pflüger, *Spatially Adaptive Sparse Grids for High-Dimensional Problems*. München: Verlag Dr. Hut, Aug. 2010.
3. A. Heinecke and D. Pflüger, "Emerging architectures enable to boost massively parallel data mining using adaptive sparse grids," *International Journal of Parallel Programming*, pp. 1–43, July 2012.
4. W. Xu and W. Xu, "Towards optimal one pass large scale learning with averaged stochastic gradient descent," *CoRR*, vol. abs/1107.2490, 2011.
5. R. Bellman, *Adaptive Control Processes: A Guided Tour*. 'Rand Corporation. Research studies, Princeton University Press, 1961.
6. E. Novak and H. Woźniakowski, "Approximation of infinitely differentiable multivariate functions is intractable," *Journal of Complexity*, vol. 25, pp. 398–404, Aug. 2009.
7. M. Hegland and G. W. Wasilkowski, "On tractability of approximation in special function spaces," *J. Complex.*, vol. 29, pp. 76–91, Feb. 2013.
8. C. J. Stone, "The dimensionality reduction principle for generalized additive models," *The Annals of Statistics*, vol. 14, pp. 590–606, June 1986.
9. T. Hastie and R. Tibshirani, *Generalized Additive Models*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability, Taylor & Francis, 1990.
10. A. Buja, T. Hastie, and R. Tibshirani, "Linear smoothers and additive models," *The Annals of Statistics*, vol. 17, pp. 453–510, June 1989.
11. E. Chu, A. Keshavarz, and S. Boyd, "A distributed algorithm for fitting generalized additive models," *Optimization and Engineering*, vol. 14, pp. 213–224, Mar. 2013.
12. D. Hsu, N. Karampatziakis, J. Langford, and A. Smola, *Parallel online learning*, ch. 14. Cambridge University Press, 2011.
13. T. K. Ho, "The random subspace method for constructing decision forests," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, pp. 832–844, Aug 1998.
14. H. van der Vorst, "Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, no. 2, pp. 631–644, 1992.
15. L. Yang and R. P. Brent, "The improved bicgstab method for large and sparse unsymmetric linear systems on parallel distributed memory architectures," in *Algorithms and Architectures for Parallel Processing, 2002. Proceedings. Fifth International Conference on*, pp. 324–328, Oct 2002.
16. C. Guangliang and M. Maggioni, "Multiscale geometric wavelets for the analysis of point clouds," in *Information Sciences and Systems (CISS), 2010 44th Annual Conference on*, pp. 1–6, March 2010.
17. J. K. Adelman-McCarthy et al., "The fifth data release of the sloan digital sky survey," *The Astrophysical Journal Supplement Series*, vol. 172, no. 2, p. 634, 2007.
18. T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

Generalizing, Optimizing, and Decoding Support Vector Machine Classification*

Mario Michael Krell[1], Sirko Straube[1], Hendrik Wöhrle[2], and Frank Kirchner[1,2]

[1] University of Bremen, Faculty 3 – Mathematics and Computer Science, Robotics Lab, Robert-Hooke-Str.1, 28359 Bremen, Germany,
krell@uni-bremen.de,
[2] German Research Center for Artificial Intelligence, DFKI Bremen, Robotics Innovation Center, Robert-Hooke-Str. 1, 28359 Bremen, Germany

Abstract. A major challenge in the classification of complex data, that requires the combination of several processing steps, is the selection of the *optimal* algorithms for preprocessing and classification. Here, we present three steps to face this problem. First, we introduce a *generalized* model for Support Vector Machine (SVM) variants which generates both unary and online classifiers. This model improves the understanding of relationships between the variants which facilitates the choice and implementation of the classifier. Second, we propose the signal processing and classification environment pySPACE which enables the systematic evaluation and comparison of algorithms. Third, we introduce an approach called backtransformation which enables a visualization of the complete processing chain in the the input data space and thereby allows for a joint interpretation of preprocessing and classification to *decode* the decision process. Finally, the benefit of combining all three approaches is shown in an application on handwritten digit classification.

Keywords: pySPACE, support vector machine, relative margin, zero separation approach, online learning, backtransformation

1 Introduction

Dealing with classification tasks of complex spatiotemporal data like the electroencephalogram (EEG) one major issue lies in the generation of meaningful features. This is due to the fact that the data often consists of a superposition of a multitude of signals, together with dynamic, and observational noise. Hence, the data processing usually requires the combination of different preprocessing steps additionally to a classifier. In fact, the generation of good features is often more important than the actual classification algorithm [1]. Consequently, in many cases expert knowledge is required in order to specify the data processing. Furthermore, there is a very large number of processing algorithms and the interplay between them is often hard to grasp. Altogether, this makes it difficult

* This work was supported by the *German Federal Ministry of Economics and Technology* (BMWi, grants FKZ 50 RA 1012 and FKZ 50 RA 1011).

to automize the process of optimizing the data processing chain to get the best preprocessing and classification. In this paper, we present three related tools to make this process easier.

Due to the ever-growing number of classification algorithms, it is difficult to decide which ones to consider. Knowledge about the relations between the classifiers facilitates the choice and implementation of classifiers. As such, instead of further specializing existing classifiers we take a unifying view. Considering only the variants of the SVM [2–5] we developed the following general concepts building connections between these classifiers. The first concept, called relative margin [6, 7], enables a connection of SVM and regularized Fisher's linear discriminant (RFLD) [8]. The second concept, the zero separation approach, allows to define unary classifiers with the help of binary classifiers by taking the origin as a second class. Third, the single iteration approach transfers batch learning classifiers to online classifiers. If the batch algorithm is repeatedly iterating over the training samples to update a linear classification function, an online learning algorithm can be generated by performing this update only once with each incoming sample. Knowing these connections simplifies the implementation of the algorithms and makes it possible to transfer extensions or modifications from one algorithm to the other connected ones. Thus, it enables to build a classifier that fits into the individual research aims.

Nevertheless, it still required to optimize the hyperparameters and the preprocessing. Hence, it is necessary to have "an infrastructure that makes experimenting with many different learners, data sources, and learning problems easy and efficient" [1]. To solve this problem, we developed the signal processing and classification environment pySPACE [9]. It provides functionality for a systematic and automated comparison of numerous algorithms and parameterizations in a signal processing chain. Additionally, pySPACE enables the visualization of data, algorithms, and evaluation results in a common framework.

Optimizing the processing and knowing the relations between classifiers is not sufficient. It is also important to *understand* the final processing model to find out what lies behind the data. A first step is to visualize the data and the single processing steps, but this might not be sufficient for a complete picture, especially when dimensionality reduction algorithms are used in the preprocessing. This is quite often the case for high-dimensional and noisy data. Hence, a representation of the entire processing chain including *both* classification and preprocessing is required. Our approach to calculate this representation is called backtransformation. It iteratively transforms the classification function back through the signal processing chain to generate a representation in the same format as the input data. This representation provides weights for each part of the data to tell which components are relevant for the complete processing and which parts are ignored. It can be directly visualized using classical data visualization approaches as they are used for visualizations of images, EEG and functional magnetic resonance imaging (fMRI) data. This visualization can then be used to support the "close collaboration between machine learning experts and application domain ones" [1]. This can help to improve the processing

and to generate new knowledge about the data. In some cases even new expert knowledge might be generated.

In the following sections, we present our steps to improve and automatize the process of designing a good processing chain for a classification problem (classifier connections, pySPACE, backtransformation) including the related work in the respective area. We conclude by giving an application example which combines the three approaches in a unified approach.

2 Generalization: Classifier Connections

The classical SVM (C-SVM) is motivated by the concept of maximizing the distance between two hyperplanes, which separate positive from negative samples. This type of regularization is extended with a loss term, which allows for samples on the opposite side of these hyperplanes. Furthermore, lifting the data into a higher-dimensional space to make it linearly separable can be replaced with kernels. Only the scalar product of two samples is substituted by a kernel function. These powerful ideas and good performance results make the SVM attractive for numerous *variants*. Some examples are: support vector regression (SVR) [10], relative margin machines (RMMs) [6, 7], least squares SVM (LS-SVM) [11], ν-SVM [12], one-class SVM (νoc-SVM) [13], support vector data description (SVDD) [14], and passive-aggressive perceptrons (PAPs) [15]. Furthermore, RFLD can be seen as an SVM variant, too [7, 8]. Some connections between these classifiers are known. In the following sections, general concepts for a unifying view are proposed to connect these classifiers and ease the process of choosing a fitting classifier: relative margin, zero separation approach, and single iteration approach. They can generate a large number of additional variants (see Fig. 1).

2.1 Relative Margin

The relative margin concept [6] adds two additional (outer) parallel hyperplanes to the C-SVM model with a relative distance (range R) to the decision hyperplane. Relative distance means that the real distance is R times $\frac{1}{\|w\|}$, when w is the classification vector. Note, $\frac{2}{\|w\|}$ is the distance between the aforementioned maximum margin hyperplanes. If outliers at the new outer margin are treated in the same way as in the inner margin, the model is called balanced relative margin machine (BRMM) [7]. This model is equivalent to SVR (with the dependent variables $Y = \{-1, 1\}$) and connects SVM ($R = \infty$) and RFLD classification ($R = 1$) [7]. A squared loss function, kernels, and implementation techniques can be directly transferred from C-SVM to BRMM. BRMM has two hyperparameters: the range R and the C-SVM complexity parameter C. For optimization it is efficient to start with high values and iteratively decrease the values with a pattern search algorithm [16]. To save resources, the *warm start* principle can be used, to adapt the batch learning algorithms to the changed parameters [17]. With this parameter optimization, it is no longer necessary to choose between SVM and RFLD.

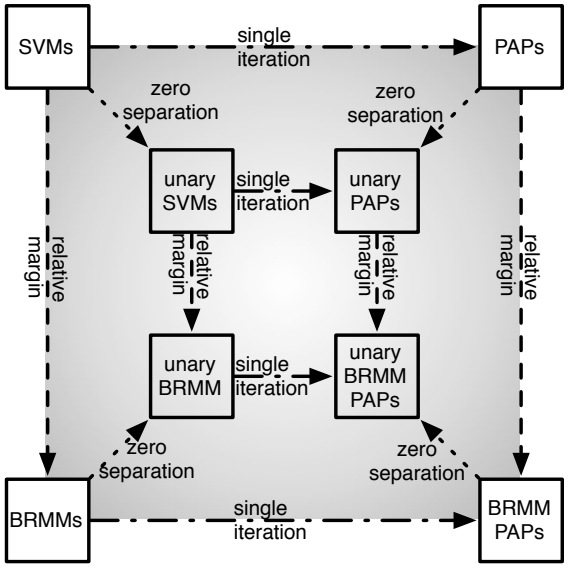


Fig. 1: Combining the approaches, introduced in Sec. 2: relative margin (vertical arrows) to generate the balanced relative margin machine (BRMM) which is the connection to the regularized Fisher's linear discriminant (RFLD), single iteration (horizontal arrows) to generate online classifiers like the passive-aggressive perceptron (PAP), and the zero separation (perpendicular arrows) to generate unary classifiers from binary ones.

2.2 Zero Separation Approach

In some applications, a second class is not of interest [18] or not enough examples of this class can be given as in outlier and novelty detection [19]. Hence, unary classifiers are required. In the zero separation approach a binary classifier is transferred to an unary classifier. The origin (zero) is added as a sample of the opposite (negative) class to the training data and the respective binary classifier is applied [13, 20]. The application of this concept to ν-SVM results in νoc-SVM, but it can be also applied to other classifiers like BRMM. Implementation techniques of the original model can be directly used.

2.3 Single Iteration Approach

The C-SVM is traditionally solved with sequential minimal optimization [21] as implemented in the LibSVM [22]. In the linear case, there are simplifications where the offset b in the decision function is omitted [17] or integrated in the data space using homogenous coordinates [23, 24] as implemented in the LIBLINEAR library [25]. Here, the solution algorithms iterate over single samples and update the classification function parameters w and b of the decision function $f(x) = \operatorname{sgn}(\langle w, x\rangle + b)$ to the optimal values in relation to this sample. The *single iteration approach* creates a variant of a classification algorithm by performing this update only once. This directly results in *online learning*

algorithms. PAPs can be derived from C-SVM with this approach [7]. Online learning can be used to speed up the training procedure with low processing resources or to improve algorithms in terms of run time. In some cases it can deliver comparable performance to the original algorithm [18, 26, 27]. It is even possible to combine batch and online learning. First, the classifier is trained on a larger dataset with a batch learning. Then by using the *single iteration approach*, the connected online learning algorithm can be adapted and used in the application.

3 Optimization: pySPACE

There are several open source signal processing and machine learning libraries. Some important libraries are NumPy [28], SciPy [29], Modular Toolkit for Data Processing [30], Weka [31], LibSVM [22], and Scikit-learn [32]. pySPACE also provides a plethora of algorithms as depicted in Fig. 2 and wraps several libraries. For finding the best processing chain, access to numerous algorithms is helpful but only to a certain extend. In contrast to other libraries, pySPACE can be seen as a *high-level framework* which provides numerous methods for *both* classification *and* preprocessing. It automates the data processing, including loading and storing of the data, parallel processing of numerous different processing flows, and evaluation of the results. The interfacing to the data and algorithms is based on configuration files and *not* on scripts. The folder which contains all datasets which shall be processed, the parameters and algorithms which should be varied, and the list of algorithms which should be applied sequentially on the data, are the only user-defined specifications. Hence, our configuration files allow scientists with little programming experience to use the software. The streamlined format of a data processing configuration can be easily shared and compared even in publications. The present approach simplifies communication between scientists and would not be possible in the same way with scripts or graphical user interfaces (GUIs). The evaluation can be performed on a cluster for fast processing and provides a result tabular with numerous metrics [33] to analyze the differences between the compared algorithms and parameterizations. pySPACE was originally developed to allow for automatic benchmarking and tuning of EEG data processing chains [34–36]. A summary on respective evaluations is given in [9] as well as more details on pySPACE.

4 Decoding: Backtransformation

To understand classifiers, it is not only important to know the relations between them, but also to interpret them when they are applied on data. This understanding might lead to an improved processing chain or even to additional information about the data or the process which generated the data. Hence, new expert knowledge could be generated. A straightforward approach is to visualize the weights of the linear classification function [37, 38]. An extension to nonlinear classifiers has been suggested in [39] based on the derivative of the classification

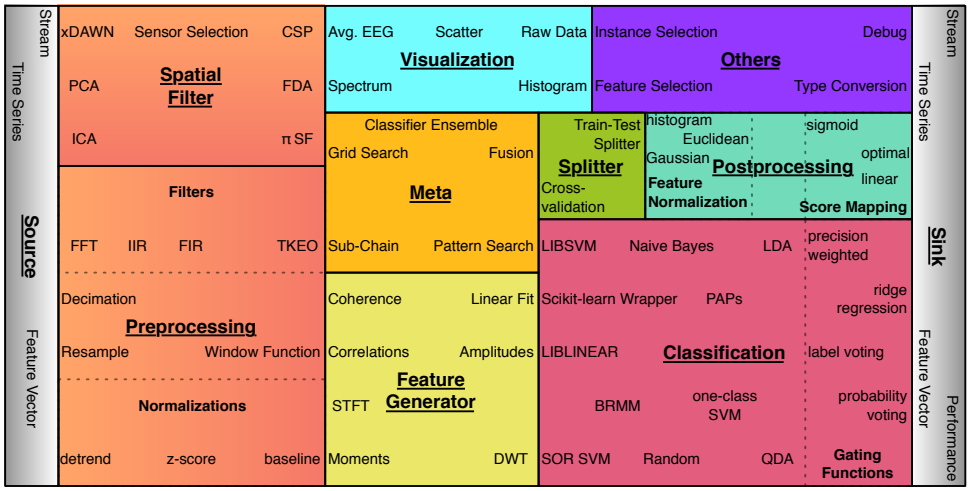
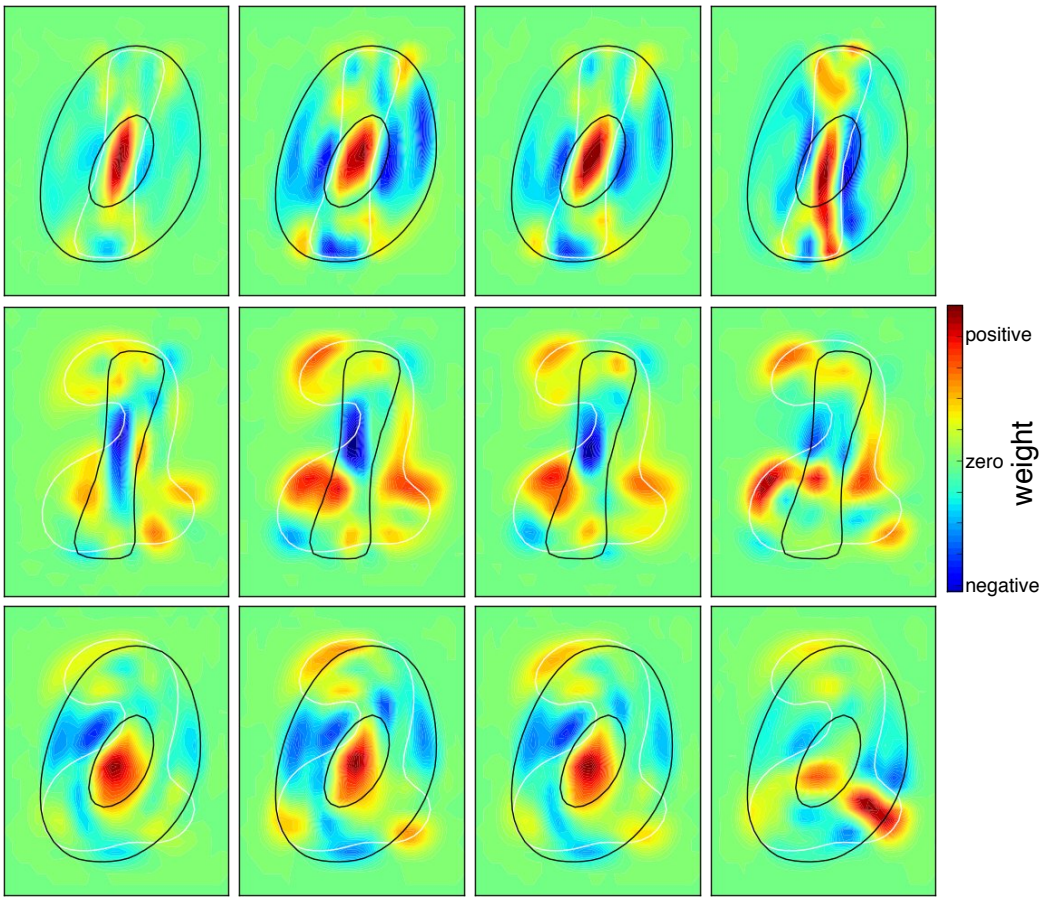


Fig. 2: Overview of the more than 100 processing nodes in pySPACE [9]. They are arranged according to processing categories (package names) and subcategories. The size of the boxes indicates the respective number of currently available algorithms.

function. Unfortunately, a derivative has to be calculated for every input sample which complicates the application and interpretation. The here proposed back-transformation concept is the extension of these methods to a complete signal processing chain, which ends with a (linear) classification function [40]. Therefore, the respective weights are calculated iteratively beginning with the classifier and going back through the processing chain. The final weights have the same format as the input data and could be visualized in the same way. Backtransformation is especially attractive when a dimensionality reduction algorithm is applied in the signal processing chain. In this case, an interpretation of the pure classifier weights is not informative without the weights of the dimensionality reduction algorithm. For nonlinear algorithms, a general transformation cannot be given anymore but it is possible to apply the chain rule and calculate the derivative of the signal processing function in the sample of interest. So for each input sample, a weight vector is obtained describing the *local* importance of each data component. Additionally to the visualization of the processing chain, back-transformation can be used to select features, to adapt a classifier to changing preprocessing (co-adaptation), and to enable sparse classification related to the input data (e.g., relevant time of the observed signal [35] or number of used sensors [34]).

5 Application Example

As a proof of concept, a classification on handwritten digit data was conducted (MNIST [41]). The classification of the digits 0, 1, and 2 is compared. First, the data was reduced in dimensionality with a principal component analysis



(a) RFLD (b) SVM (c) PAP (d) unary SVM

Fig. 3: *Contour plots of backtransformation weights for handwritten digit classification with different classifiers:* The white and black silhouettes display an average contour of the original data (digits 0, 1, and 2). The colored contour plots show the respective weights in the classification process. Negative weights (blue) are important for the classification of the first class (black silhouette) and positive weights (red) for the second class (white silhouette). Green weights are close to zero and do not contribute to the classification process. For the unary classification, the second class (white) was used.

(PCA) [42] from 784 to 40 and then normalized with a standardization (zero mean and variance of one on the given training data). For classification, a squared loss penalization of misclassifications was used to obtain a Gaussian loss in RFLD. RFLD, SVM, the respective SVM perceptron, and νoc-SVM were compared. Backtransformation can summarize all three processing steps and provides the respective weights belonging to the input data. This is visualized in Fig. 3. The classifiers itself do only determine the 40 weights of the normalized principal components. These weights would be difficult to interpret, but with the given backtransformation the weighting and its correspondence to the average

shapes can be observed. As expected due to the model similarities (single iteration approach) similar weight distributions were obtained for SVM and its online learning variant (PAP). The visualizations of SVM and RFLD look similar due to the connection with BRMM. However, for the distinction between the two digits 0 and 2 some larger differences can be observed. The one class classifier is different to the other classifiers as expected because it has been trained on a single digit only. Hence, characteristics of the other class can be only marginally observed due to the use of PCA which has been trained on both classes. This can be seen in the second and third row: although trained on the digit 2 in both cases, the classification results look different.

6 Conclusion

Optimizing the classification of spatiotemporal data is a difficult task which often requires expert knowledge. To ease this process especially for non-experts, three approaches are shown in this paper to improve the design and understanding of signal processing and classification with SVM variants. The pySPACE framework was presented, to process the data, tune algorithms and their parameters, and to enable the communication between scientists. Several connections between existing SVM variants have been shown and resulted in additional new SVM variants for unary classification and online learning. Due to the connections, it is easier to understand differences and similarities between the classifier variants and save time when implementing the classifiers. To finally interpret the complete signal processing chain which ends with a classifier, the backtransformation approach was presented. In case of solely affine transformations, it results in a representation of the processing chain, giving weights for each component in the input domain, which can be directly visualized. All three approaches were combined in an application on handwritten digit classification.

In future, the backtransformation concept should be implemented and tested on nonlinear signal processing chains. All three introduced concepts should be analyzed in further applications to prove their usefulness. The longterm goal is to make pySPACE a tool for autonomous learning.

References

1. Domingos, P.: A few useful things to know about machine learning. Communications of the ACM **55**(10) (2012) 78–87
2. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press (2000)
3. Müller, K.R., Mika, S., Rätsch, G., Tsuda, K., Schölkopf, B.: An introduction to kernel-based learning algorithms. IEEE Transactions on Neural Networks **12**(2) (2001) 181–201
4. Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge (2002)
5. Vapnik, V., Others: The nature of statistical learning theory. 1995. Springer **120**(6) (2000)

6. Shivaswamy, P.K., Jebara, T.: Maximum relative margin and data-dependent regularization. Journal of Machine Learning Research **11** (2010) 747–788
7. Krell, M.M., Feess, D., Straube, S.: Balanced Relative Margin Machine The missing piece between FDA and SVM classification. Pattern Recognition Letters **41** (2014) 43–52
8. Mika, S.: Kernel Fisher Discriminants. PhD thesis, Technische Universität Berlin (2003)
9. Krell, M.M., Straube, S., Seeland, A., Wöhrle, H., Teiwes, J., Metzen, J.H., Kirchner, E.A., Kirchner, F.: pySPACE a signal processing and classification environment in Python. Frontiers in Neuroinformatics **7**(40) (2013) https://github.com/pyspace.
10. Smola, A.J., Schölkopf, B.: A tutorial on support vector regression. Statistics and Computing **14**(3) (2004) 199–222
11. Van Gestel, T., Suykens, J.A.K., Lanckriet, G., Lambrechts, A., De Moor, B., Vandewalle, J.: Bayesian framework for least-squares support vector machine classifiers, gaussian processes, and kernel Fisher discriminant analysis. Neural Computation **14**(5) (2002) 1115–1147
12. Schölkopf, B., Smola, A.J., Williamson, R.C., Bartlett, P.L.: New Support Vector Algorithms. Neural Computation **12**(5) (2000) 1207–1245
13. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Computation **13**(7) (2001) 1443–1471
14. Tax, D.M.J., Duin, R.P.: Support Vector Data Description. Machine Learning **54**(1) (2004) 45–66
15. Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y.: Online Passive-Aggressive Algorithms. Journal of Machine Learning Research **7** (2006) 551 – 585
16. Eitrich, T., Lang, B.: Efficient optimization of support vector machine learning parameters for unbalanced datasets. Journal of Computational and Applied Mathematics **196**(2) (2006) 425–436
17. Steinwart, I., Hush, D., Scovel, C.: Training SVMs without offset. Journal of Machine Learning Research **12** (2009) 141–202
18. Krell, M.M., Kirchner, E.A., Wöhrle, H.: P300 Detection as an Unary Classification Problem. (submitted 2014)
19. Aggarwal, C.C.: Outlier Analysis. Springer New York, New York (2013)
20. Krell, M.M., Wöhrle, H.: New One-Class Classifiers based on the Zero Separation Approach. (submitted 2014)
21. Platt, J.C.: Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In Schölkopf, B., Burges, C.J.C., Smola, A.J., eds.: Advances in Kernel Methods. MIT Press (1999) 61–74
22. Chang, C.C., Lin, C.J.: LIBSVM. ACM Transactions on Intelligent Systems and Technology **2**(3) (2011) 1–27
23. Mangasarian, O.L., Musicant, D.R.: Successive Overrelaxation for Support Vector Machines. IEEE Transactions on Neural Networks **10** (1998) 1032 – 1037
24. Hsieh, C.J., Chang, K.W., Lin, C.J., Keerthi, S.S., Sundararajan, S.: A dual coordinate descent method for large-scale linear SVM. In: Proceedings of the 25th International Conference on Machine learning (ICML 2008), ACM Press (2008) 408–415
25. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A Library for Large Linear Classification. The Journal of Machine Learning Research **9** (2008) 1871–1874

26. Wöhrle, H., Krell, M.M., Straube, S., Kirchner, E.A., Kirchner, F.: An Online Adaptable Spatial Filter for User-Independent Single Trial Detection of Event-Related Potentials. (submitted 2014)
27. Wöhrle, H., Teiwes, J., Krell, M.M., Kirchner, E.A., Kirchner, F.: A dataflow-based mobile brain reading system on chip with supervised online calibration. In: Proc. International Congress on Neurotechnology, Electronics and Informatics, Vilamoura, ScitePress (2013) 46–53
28. Dubois, P.F.: Extending Python with Fortran. Computing Science and Engineering **1**(5) (1999) 66–73
29. Jones, E., Oliphant, T., Peterson, P., et al.: SciPy: Open source scientific tools for Python (2001) http://www.scipy.org/.
30. Zito, T., Wilbert, N., Wiskott, L., Berkes, P.: Modular toolkit for Data Processing (MDP): a Python data processing framework. Frontiers in Neuroinformatics **2**(8) (2008)
31. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software. ACM SIGKDD Explorations Newsletter **11**(1) (2009) 10–18
32. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research **12** (2011) 2825–2830
33. Straube, S., Krell, M.M.: How to evaluate an agent's behaviour to infrequent events? – Reliable performance estimation insensitive to class distribution. Frontiers in Computational Neuroscience **8**(43) (2014)
34. Feess, D., Krell, M.M., Metzen, J.H.: Comparison of sensor selection mechanisms for an ERP-based brain-computer interface. PLoS ONE **8**(7) (2013) e67543
35. Kirchner, E.A., Kim, S.K., Straube, S., Seeland, A., Wöhrle, H., Krell, M.M., Tabie, M., Fahle, M.: On the applicability of brain reading for predictive human-machine interfaces in robotics. PLoS ONE **8**(12) (2013) e81732
36. Krell, M.M., Tabie, M., Wöhrle, H., Kirchner, E.A.: Memory and Processing Efficient Formula for Moving Variance Calculation in EEG and EMG Signal Processing. In: Proc. International Congress on Neurotechnology, Electronics and Informatics, Vilamoura, ScitePress (2013) 41–45
37. LaConte, S., Strother, S., Cherkassky, V., Anderson, J., Hu, X.: Support vector machines for temporal classification of block design fMRI data. NeuroImage **26**(2) (2005) 317–329
38. Blankertz, B., Lemm, S., Treder, M., Haufe, S., Müller, K.R.: Single-Trial Analysis and Classification of ERP Components–a Tutorial. NeuroImage **56**(2) (2011) 814–825
39. Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., Müller, K.R.: How to Explain Individual Classification Decisions. Journal of Machine Learning Research **11** (2010) 1803–1831
40. Krell, M.M., Straube, S.: Backtransformation: A new representation of data processing chains with a scalar decision function. (under revision 2014)
41. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11) (1998) 2278–2324
42. Abdi, H., Williams, L.J.: Principal component analysis. Wiley Interdisciplinary Reviews: Computational Statistics **2**(4) (2010) 433–459

Robust Optimization using Machine Learning for Uncertainty Sets

Theja Tulabandhula and Cynthia Rudin

CSAIL, MIT, Cambridge, MA 02139, USA

Abstract. Our goal is to build robust optimization problems for making decisions based on complex data from the past. In robust optimization (RO) generally, the goal is to create a policy for decision-making that is robust to our uncertainty about the future. In particular, we want our policy to best handle the the worst possible situation that could arise, out of an *uncertainty set* of possible situations. Classically, the uncertainty set is simply chosen by the user, or it might be estimated in overly simplistic ways with strong assumptions; whereas in this work, we learn the uncertainty set from data collected in the past. The past data are drawn randomly from an (unknown) possibly complicated high-dimensional distribution. We propose a new uncertainty set design and show how tools from statistical learning theory can be employed to provide probabilistic guarantees on the robustness of the policy.
Keywords: machine learning, uncertainty sets, robust optimization, data-driven decision making, decision making under uncertainty.

1 Introduction

In this work, we consider a situation often faced by decision makers: a policy needs to be created for the future that would be a best possible reaction to the worst possible uncertain situation; this is a question of *robust optimization*. In our case, the decision maker does not know what the worst situation might be, and uses complex data to estimate the *uncertainty set*, which is the set of uncertain future situations. Here we are interested in answering questions such as: How might we construct a principled uncertainty set from these complex data? Can we ensure that with high probability our policy will be robust to whatever the future brings?

The uncertainty set $\mathcal{U}$ can be defined in many ways, and the central goal of this work is how to model $\mathcal{U}$ from complex data from the past. The data $\{(\mathbf{x}^i, y^i)\}_{i=1}^n$ take the form of features and labels, with $\mathbf{x}^i \in \mathcal{X} \subseteq \mathbb{R}^d$ and $y^i \in \mathcal{Y}$. Some of the different ways uncertainty sets can be constructed are:

• Using a priori assumptions: We may have *a priori* knowledge about the range of possible future situations. This knowledge can guide us in constructing the uncertainty set $\mathcal{U}$ using, for instance, interval constraints.

• Using empirical statistics: We could create an uncertainty set using empirical statistics of the labels ignoring the feature vectors altogether.

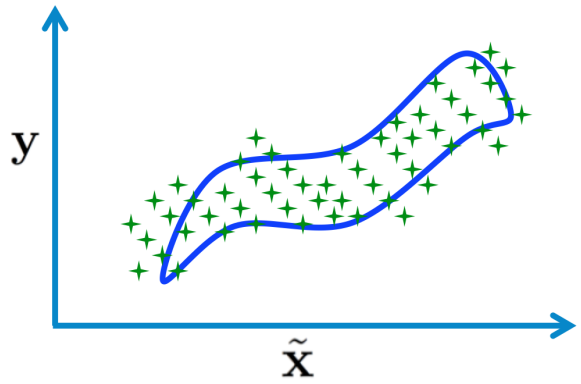
• Using linear regression to model complex data: Here, we use the complex past data $\{(\mathbf{x}^i, \mathbf{y}^i)\}_{i=1}^n$, but we make strong (potentially incorrect) assumptions on the probability distribution these data are drawn from.
• Using machine learning to model complex data, which is the topic of this work: This setting is more general than linear regression and with much weaker assumptions. We provide two principled ways to construct set $\mathcal{U}$ using historical data. In both, we optimize prediction models over the data $\{(\mathbf{x}^i, \mathbf{y}^i)\}_{i=1}^n$, and use them to construct uncertainty set $\mathcal{U}$. $\mathcal{U}$ is used within the robust optimization problem to construct $\boldsymbol{\pi}^*$, and Theorem 1 provides a guarantee on its robustness; this guarantee is derived using statistical learning theory. Theorem 1 describes the guarantee for a generic class of prediction models and Theorem 2 specializes the guarantee for a specific set of prediction models, namely, the conditional quantile models. The only assumption made in this approach is that the data are drawn i.i.d from an unknown source distribution. In particular, there is no normality assumption. Let us give examples of how the two methods we propose for this approach would work when $\mathcal{U}$ is constructed from a regression problem:

– For the first method, for every $\tilde{\mathbf{x}}$ the uncertainty set $\mathcal{U}$ corresponds to the domain of a indicator function on part of the set $\mathcal{Y}$. It is 1 on most of the training examples and is 0 farther away from them. Figure 1(a) shows an illustration of this.
– For the second method, we estimate the 95^{th} and 5^{th} percentiles of $\mathbf{y}$ given $\tilde{\mathbf{x}}$ and set $\mathcal{U}$ to be all values of $\mathbf{y} \in \mathcal{Y}$ between the two estimates. Figure 1(b) illustrates this.

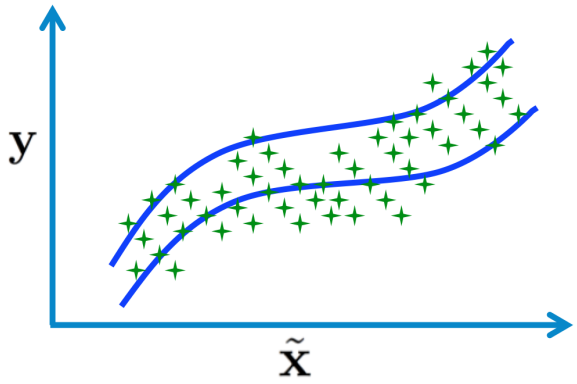
Being able to define uncertainty sets from predictive models is important: the uncertainty sets can now be specialized to a given new situation $\tilde{\mathbf{x}} \in \mathcal{X}$, and this is true even if we have never seen $\tilde{\mathbf{x}}$ before. For instance, when ordering daily supplies $\mathbf{y}^i$ for an ice cream parlor in Boston, an uncertainty set that depends on the weather might be much smaller than one that does not; planning for too much uncertainty in the weather can be too conservative and very costly: it would not be wise to budget for the largest possible summer sales in the middle of the winter.

Our approaches for constructing uncertainty sets are flexible, intuitive, easy to understand from a practitioner's point of view, and at the same time can bring all the rich theoretical results of learning theory to justify the data-driven methodology. Our uncertainty set designs can handle prediction models for classification, regression, ranking and other supervised learning problems. A main theme of this work is that RO is a new context in which many learning theory results naturally apply and can be directly used.

The closest work to ours is possibly that of [1], where the authors provide a linear-regression-based robust decision making paradigm for portfolio allocation problems, where they assume a multivariate linear regression model for the learning step. A big departure from this approach is that in our work, we are able to design uncertainty sets for a general class of decision making problems while making weak assumptions about the distributional aspects of the historical data.



(a) Using optimized set function



(b) Using optimized conditional quantile functions

Fig. 1. The empirical data $\{\mathbf{x}^i, \mathbf{y}^i\}_{i=1}^n$ is shown along with the boundaries created by the proposed methods in each of the above figures. Evaluation of these boundaries at a given $\tilde{\mathbf{x}}$ produces an uncertainty set. In (a), a set function is optimized over the sample and its evaluation at every $\tilde{\mathbf{x}}$ is plotted. In (b), we use optimized conditional quantile models to get the boundaries.

We base our uncertainty set design on regularized empirical risk minimization, which is quite a bit more general than regression.

2 Formulation

Let all the uncertain parameters of the decision problem be denoted by a vector $\mathbf{u} \in \mathbb{R}^m$. Given a realization of $\mathbf{u}$, let the (basic non-robust) decision making problem be written as:

$$\min_{\boldsymbol{\pi}} \rho(\boldsymbol{\pi}, \mathbf{u}) \quad \text{s.t.} \;\; F(\boldsymbol{\pi}, \mathbf{u}) \in \mathcal{K}. \tag{1}$$

Here $\boldsymbol{\pi} \in \mathit{\Pi} \subseteq \mathbb{R}^{d_1}$ is the decision vector and $f : \mathit{\Pi} \times \mathbb{R}^m \to \mathbb{R}$ is the objective function. Function $F : \mathit{\Pi} \times \mathcal{U} \to \mathcal{K}$ and convex cone $\mathcal{K} \subseteq \mathbb{R}^{d_2}$ describe the constraints of the problem.

The robust version of the decision problem in Equation (1) is thus:

$$\min_{\boldsymbol{\pi}} \max_{\mathbf{u} \in \mathcal{U}} f(\boldsymbol{\pi}, \mathbf{u}) \quad \text{s.t.} \;\; F(\boldsymbol{\pi}, \mathbf{u}) \in \mathcal{K} \text{ for all } \mathbf{u} \in \mathcal{U}, \tag{2}$$

where $\mathcal{U} \subset \mathbb{R}^m$ represents the uncertainty set.

To solve Equation (2), we prescribe the following steps:

Step 1: Construct $\mathcal{U}$ using any of the four methods listed in this section.

Step 2: Obtain a robust solution, using either of the two options below:

Option 1: If $\mathcal{U}$ is a "nice" set, then there are natural ways [2] to transform it into a relaxed set $\mathcal{U}'$ so that the robust optimization problem can be solved to obtain a robust solution π^*. For instance, if U can be bounded using a box or an ellipsoid, that box or ellipsoid can be $\mathcal{U}'$.

Option 2: If $\mathcal{U}$ is not a "nice" set, then sample L elements from $\mathcal{U}$ uniformly. Then solve the sampled version of Equation (2) to obtain a robust solution $\boldsymbol{\pi}^*$.

We focus on **Step 1**. The goal is to ensure that the true realization of parameter $\mathbf{u} \in \mathbb{R}^m$ belongs to set $\mathcal{U}$ with a high likelihood. Let $\mathbf{u}$ be equal to an m-dimensional vector of unknown labels $[\tilde{y}^1 \;\; \ldots \;\; \tilde{y}^m]^T$, where each label $\tilde{y}^j \in \mathcal{Y}$ can be predicted given a corresponding feature vector $\tilde{\mathbf{x}}^j \in \mathcal{X}$. Thus m labels $\{\tilde{y}^j\}_{j=1}^m$, which can be forecasted from $\{\tilde{\mathbf{x}}^j\}_{j=1}^m$, feed into the decision problem of Equation (2).

General prediction models:

Let $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$ represent a feature vector and $y \in \mathcal{Y} \subseteq \mathbb{R}$ represent a label. Consider a class of set functions $I \in \mathcal{I}$, where $I : \mathcal{X} \to \mathfrak{M}_{\mathbb{R}}$, where $\mathfrak{M}_{\mathbb{R}}$ is the set of all measurable sets of $\mathbb{R}$. Let us say that we have a procedure that picks a function I^{Alg} so that most of the labels of the training examples obey $y^i \in I^{\text{Alg}}(\mathbf{x}^i),\ i = 1, ..., n$. As long as I^{Alg} belongs to a set of "simple" functions, we have a guarantee on how well I^{Alg} will generalize to new observations. Specifically, consider the following empirical risk minimization procedure:

$$\min_{I \in \mathcal{I}} \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}[y^i \notin I(\mathbf{x}^i)], \tag{3}$$

where $\mathbf{1}[\cdot]$ is the indicator function. Let an optimal solution to the above problem be I^{Alg}. Then, define the uncertainty set $\mathcal{U}$ as:

$$\mathcal{U} = \Pi_{j=1}^{m} I^{\text{Alg}}(\tilde{\mathbf{x}}^j), \tag{4}$$

where $\mathcal{U}$ is a product of m measurable sets.

The above setting is quite general. In particular, since the range of function I^{Alg} is $\mathfrak{M}_{\mathbb{R}}$, we can capture sets that are arbitrarily more complicated than simple intervals. For instance, if $\mathbb{P}_{y^j|\tilde{\mathbf{x}}^j}$ is bimodal, then for certain values of $\tilde{\mathbf{x}}^j$, $I^{\text{Alg}}(\tilde{\mathbf{x}}^j)$ can be the union of two disjoint intervals.

Conditional quantile models:

In this method, we specialize the generic function class $\mathcal{I}$ to the class of set functions defined using conditional quantile models. We will estimate an upper quantile of $\tilde{y}$ for each $\tilde{\mathbf{x}}$, and a lower quantile of $\tilde{y}$ for each $\tilde{\mathbf{x}}$. The uncertainty set will be the interval between the two quantile estimates. This method is applicable when our prediction task is a regression problem.

When $y \sim \mathbb{P}_y$, the τ^{th} quantile of y, denoted by μ^τ, is defined as $\mu^\tau := \inf\{\mu : \mathbb{P}_y(y \leq \mu) = \tau\}$. Here τ can vary between 0 and 1. In the special case when τ is set to 0.5, this defines the median. Similarly, when $(\mathbf{x}, y) \sim \mathbb{P}_{\mathbf{x},y}$, the conditional quantile μ^τ can be defined as a function from $\mathcal{X}$ to $\mathcal{Y}$, $\mu^\tau(\mathbf{x}) := \inf\{\mu : \mathbb{P}_{y|\mathbf{x}}(y \leq \mu) = \tau\}$.

In our setting, $\tilde{y}^j$ conditioned on $\tilde{\mathbf{x}}^j$ is distributed according to $\mathbb{P}_{\tilde{y}^j|\tilde{\mathbf{x}}^j}$. Thus, given a value of $\tau \in [0, 1]$, $\mathbb{P}_{\tilde{y}^j|\mathbf{x}=\tilde{\mathbf{x}}^j}(\tilde{y}^j \leq \mu^\tau(\tilde{\mathbf{x}}^j)) = \tau$ where $\mu^\tau(\mathbf{x})$ is the conditional quantile defined earlier. Our method picks two values of τ, $\delta_p \leq \delta_q$ such that:

$$\mathbb{P}_{\tilde{y}^j|\tilde{\mathbf{x}}^j}(\tilde{y}^j \leq \mu^{\delta_p}(\tilde{\mathbf{x}}^j)) = \delta_p, \text{ and } \mathbb{P}_{\tilde{y}^j|\tilde{\mathbf{x}}^j}(\tilde{y}^j \leq \mu^{\delta_q}(\tilde{\mathbf{x}}^j)) = \delta_q.$$

For example, a typical value for the pair (δ_p, δ_q) can be $(0.05, 0.95)$ which makes $\mu^{\delta_p}(\tilde{\mathbf{x}}^j)$ correspond to the 5% conditional quantile and $\mu^{\delta_q}(\tilde{\mathbf{x}}^j)$ correspond to the 95% conditional quantile. Given these two conditional quantiles, we have:

$$\mathbb{P}_{\tilde{y}^j|\tilde{\mathbf{x}}^j}(\mu^{\delta_p}(\tilde{\mathbf{x}}^j) < \tilde{y}^j \leq \mu^{\delta_q}(\tilde{\mathbf{x}}^j)) = \delta_q - \delta_p.$$

Thus, the unknown future realization of $\tilde{y}^j$ belongs to the interval $[\mu^{\delta_p}(\tilde{\mathbf{x}}^j), \mu^{\delta_q}(\tilde{\mathbf{x}}^j)]$ with high probability if δ_p and δ_q are chosen appropriately.

Quantile regression can be seen as an empirical risk minimization algorithm where the loss function is defined appropriately to obtain a conditional quantile function. That is, we aim to obtain an estimator function $\beta(\mathbf{x})$ of the true conditional quantile function $\mu^\tau(\mathbf{x})$ given a predefined quantile parameter τ. In particular, the *pinball* loss (or newsvendor loss) function defined below is used.

$$l^\tau(\beta(\mathbf{x}), y) = \begin{cases} \tau \cdot (y - \beta(\mathbf{x})) & \text{if } y - \beta(x) \geq 0, \\ (\tau - 1) \cdot (y - \beta(\mathbf{x})) & \text{otherwise.} \end{cases}$$

Let $l^\tau_{\mathbb{P}}(\beta) = \mathbb{E}_{\mathbf{x},y}[l^\tau(\beta(\mathbf{x}), y)]$. In our setting, we will let $\mathcal{B}_0$ be our hypothesis class that we want to pick conditional quantile functions from.

Let the empirical risk minimization procedure using the pinball loss output a conditional quantile model $\beta^{Alg,\tau}$ when given the historical sample $S = \{(\mathbf{x}^i, y^i)\}_{i=1}^n$ of size n and a parameter τ. That is, let $l_S^\tau(\beta) = \frac{1}{n}\sum_{i=1}^n l^\tau(\beta(\mathbf{x}^i), y^i)$ and $\beta^{Alg,\tau} \in \arg\min_{\beta\in\mathcal{B}_0} l_S^\tau(\beta)$. The following definition of $\mathcal{U}$ uses two empirical conditional quantile functions with $\tau = \delta_p$ and $\tau = \delta_q$ respectively:

$$\mathcal{U} = \Pi_{j=1}^m \left[\min\left(\beta^{\mathrm{Alg},\delta_p}(\tilde{\mathbf{x}}^j), \beta^{\mathrm{Alg},\delta_q}(\tilde{\mathbf{x}}^j)\right), \max\left(\beta^{\mathrm{Alg},\delta_p}(\tilde{\mathbf{x}}^j), \beta^{\mathrm{Alg},\delta_q}(\tilde{\mathbf{x}}^j)\right)\right]. \quad (5)$$

Here $\mathcal{U}$ is again a product of m intervals, each one constructed so that it contains the unknown $\tilde{y}^j$ with high probability.

3 Robustness guarantee using general prediction functions

Consider the setting described in Section 2, where we have a class of general set functions $\mathcal{I}$. Let $S := \{(\mathbf{x}^i, y^i)\}_{i=1}^n$ be the training data which are independent and identically distributed. Let algorithm A represent a generic learning procedure. That is, it takes S as an input and outputs I^{Alg}. Since I^{Alg} is a function of sample S, we will show that the unknown $\tilde{y}^j$ belong to the interval $I^{\mathrm{Alg}}(\tilde{\mathbf{x}}^j)$ with high probability over S as long as the set of functions $\mathcal{I}$ from which I^{Alg} is picked is "simple". Note that we do not assume anything about the source distribution.

In order to state our result, we will define the following quantity known as the empirical Rademacher average [3]. For a set $\mathcal{F}$ of functions, the *empirical Rademacher average* is defined with respect to a given random sample $S' = \{z^i\}_{i=1}^n$ as

$$\mathcal{R}_{S'}(\mathcal{F}) = \mathbb{E}_{\sigma^1,\ldots,\sigma^n}\left[\frac{1}{n}\sup_{f\in\mathcal{F}}\left|\sum_{i=1}^n \sigma^i f(z^i)\right|\right],$$

where for each $i = 1,..,n$, $\sigma^i = \pm 1$ with equal probability. The *Rademacher average* is defined to be the expectation of the empirical Rademacher average over the random sample S: $\mathcal{R}(\mathcal{H}) = \mathbb{E}_{z^1,\ldots,z^n}[\mathcal{R}_S(\mathcal{H})]$. The interpretation of the Rademacher average is that it measures the ability of function class $\mathcal{F}$ to fit noise, coming from the random $\sigma_i's$. If the function class can fit noise well, it is a highly complex class. The Rademacher average is one of many ways to measure the richness of a function class, including covering numbers, fat-shattering dimensions [4] and the Vapnik-Chervonenkis dimension [5].

Theorem 1. *If $\mathcal{U}$ is defined as in Equation (4), then with probability at least $1-\delta$ over training sample S, we have robustness guarantee*

$$\mathbb{P}_{\{\tilde{\mathbf{x}}^j,\tilde{y}^j\}_{j=1}^m}\Big(F(\pi^*, [\tilde{y}^1 ... \tilde{y}^m]^T) \in \mathcal{K}\Big) \geq \left(\left[1 - \frac{1}{n}\sum_{i=1}^n \mathbf{1}[y^i \notin I^{Alg}(\mathbf{x}^i)] - 2\mathcal{R}(l\circ\mathcal{I}) - \sqrt{\frac{\log\frac{1}{\delta}}{2n}}\right]_+\right)^m,$$

where $\epsilon > 0$ *is a pre-determined constant, and* $\left[\cdot \right]_+$ *is shorthand for* $\max(0, \cdot)$.

The result is a lower bound on the probability of infeasibility. This bound depends on the performance of the data dependent set function I^{Alg}. If I^{Alg} is such that its performance, measured in terms of $\frac{1}{n}\sum_{i=1}^{n} \mathbf{1}[y^i \notin I^{\text{Alg}}(\mathbf{x}^i)]$ is good (i.e., lower in value), then the right hand side of the inequality increases, resulting in a higher chance of feasibility. This probability of feasibility also depends on the number of estimates m that enter the decision problem of Equation (2). When $n \to \infty$, the Rademacher term and the square root terms become zero and the probability of feasibility depends on the asymptotic performance of I^{Alg} (which converges to I^*, the 'best-in-class' set function), as desired.

4 Robustness guarantee using conditional quantile functions

Theorem 2. *If* $\mathcal{U}$ *is defined as in Equation (5), then with probability at least* $1 - \delta$ *over training sample* S*, we have*

$$\mathbb{P}_{\{\tilde{\mathbf{x}}^j, \tilde{y}^j\}_{j=1}^m}\Big(F(\pi^*, [\tilde{y}^1 ... \tilde{y}^m]^T) \in \mathcal{K}\Big) \geq$$

$$\left(\left[\frac{1}{n}\sum_{i=1}^{n}\Big(r_\epsilon^-(y^i - \beta^{Alg,\delta_q}(\mathbf{x}^i)) - r_\epsilon^+(y^i - \beta^{Alg,\delta_p}(\mathbf{x}^i))\Big) - \frac{8}{\epsilon}\mathcal{R}(\mathcal{B}_0) - 2\sqrt{\frac{\log\frac{2}{\delta}}{2n}}\right]_+\right)^m, \tag{6}$$

where $\epsilon > 0$ *is a pre-determined constant,* $\left[\cdot \right]_+$ *is shorthand for* $\max(0, \cdot)$*,* $r_\epsilon^-(z) := \min\Big(1, \max\big(0, -\frac{z}{\epsilon}\big)\Big)$ *and* $r_\epsilon^+(z) := \min\Big(1, \max\big(0, 1 - \frac{z}{\epsilon}\big)\Big)$.

The lower bound is a function of the empirical performance of the two conditional quantile estimators and the Rademacher average of the hypothesis set. As $n \to \infty$, the Rademacher average and the square-root term tend to zero at a rate $O(\frac{1}{\sqrt{n}})$. The term $\frac{1}{n}\sum_{i=1}^{n}\Big(r_\epsilon^-(y^i - \beta^{\text{Alg},\delta_q}(\mathbf{x}^i)) - r_\epsilon^+(y^i - \beta^{\text{Alg},\delta_p}(\mathbf{x}^i))\Big)$ converges to $\mathbb{P}_{\mathbf{x},y}(\beta^{\text{Alg},\delta_p}(\mathbf{x}) \leq y \leq \beta^{\text{Alg},\delta_q}(\mathbf{x}))$.

5 Conclusion

In this paper, we considered a class of single-stage decision making problems where the uncertainty is derived from statistical modeling. We present two principled approaches to design uncertainty sets in the robust optimization framework for these problems using statistical learning theory. In the first approach, we use a general class of set functions and define the uncertainty set using them. The second approach develops this idea further using the notion of quantiles to define the uncertainty set. For both approaches, we give probabilistic guarantees on the feasibility of the robust solutions thus obtained.

Bibliography

[1] Donald Goldfarb and Garud Iyengar. Robust portfolio selection problems. *Mathematics of Operations Research*, 28(1):1–38, 2003.

[2] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton University Press, 2009.

[3] Peter L Bartlett, Olivier Bousquet, and Shahar Mendelson. Localized rademacher complexities. In *Computational Learning Theory*, pages 44–58. Springer, 2002.

[4] Peter L Bartlett, Philip M Long, and Robert C Williamson. Fat-shattering and the learnability of real-valued functions. *Journal of Computer and System Sciences*, 52(3):434–452, 1996.

[5] Vladimir N Vapnik. *Statistical learning theory*. Wiley, 1998.

Heterogeneous Dataflow Hardware Accelerators for Machine Learning on Reconfigurable Hardware *

Hendrik Woehrle[1], Johannes Teiwes[2], Mario Michael Krell[2], Anett Seeland[1], Elsa Andrea Kirchner[1,2], and Frank Kirchner[1,2]

[1] German Research Center for Artificial Intelligence, DFKI Bremen, Robotics Innovation Center, Robert-Hooke-Str. 1, 28359 Bremen, Germany
hendrik.woehrle@dfki.de,
[2] University of Bremen, Faculty 3 – Mathematics and Computer Science, Robotics Lab, Robert-Hooke-Str.1, 28359 Bremen, Germany,

Abstract. The trend to develop increasingly more intelligent systems leads directly to a considerable demand for more and more computational power. However, in certain application domains, such as robotics, there are several technical limitations, like restrictions regarding power consumption and physical size, that make the use of powerful generic processors unfeasible. One possibility to overcome this problem is the usage of specialized hardware accelerators, which are designed for typical tasks in machine learning. In this paper, we propose an approach for the rapid development of hardware accelerators that are based on the heterogeneous dataflow computing paradigm. The developed techniques are collected in a framework to provide a simple access to them. We discuss different application areas and show first results in the field of biosignal analysis that can be used for rehabilitation robotics.

Keywords: Robotics, Embedded Systems, FPGA, Hardware Acceleration, Dataflow

1 Introduction

Machine learning techniques are widely used nowadays for a broad range of applications. Depending on the specific application, they are employed either on commodity hardware like desktop PCs or powerful high-end systems, like clusters. The primary objective in these settings is to achieve a good accuracy of the methods. Other objectives, like computational efficiency and memory consumption are often regarded as less important or even entirely ignored.

However, with the increasing importance of portable and embedded systems and in the eras of Big Data and wearable computing, these formerly secondary objectives become more and more important. Especially in the case of

* This work was supported by the *German Federal Ministry of Economics and Technology* (BMWi, grants FKZ 50 RA 1012 and FKZ 50 RA 1011).

autonomous systems and robotics, stringent requirements have to be satisfied: 1) the available physical space is limited, which often prohibits the usage of off-the-shelf components like desktop PCs, 2) the power consumption should be small to reduce heat generation, allow the usage of small accumulators, and increase the running time of the system, 3) often real time processing is needed, since the systems have to work in a real world environment and have to keep up with the environment.

1.1 The Problem of Generic Processors

These requirements make the usage of standard *generic* processors suboptimal. The main purpose of generic processors is the execution of arbitrary software, but not the high performance execution of *specific* algorithms. Consequently, they are 1) either bulky and powerful, or small but weak, 2) waste to much energy for the computational power they provide, 3) can not guarantee real time processing themselves, but require real time operating systems. Fortunately, genericness is not always required. Especially many algorithms in machine learning and signal processing depend on a small set of operations like matrix-vector computations (e.g., in neural networks, and support vector machines (SVMs)) or convolution (e.g., in finite impulse response (FIR) filtering or edge detection in image processing).

The above stated facts make it possible to use a dichotomy here: combine a *generic*, but *weak* CPU for software tasks, with *application specific hardware accelerators* for high performance computations. This pattern is widely used. Examples are the usage of specific accelerators, e.g., to reduce the energy consumption in smart phones [1] or to perform machine learning tasks in the Microsoft Kinect [2]. In these devices application specific integrated circuits (ASICs) are used to fulfill a single, specific task. ASICs can not be transferred to other applications - every time the requirements change, a new ASIC has to be constructed. This time consuming and expensive and therefore only reasonable if large quantities are produced. Consequently, ASICs are inflexible, since it is impossible to consider improvements of the underlying algorithm after the ASIC is manufactured. This is not feasible for robotics or machine learning in mobile systems.

Another example is generic computing on graphics processing units (GPUs), which is also often used in the machine learning community [3]. However, GPUs have a high power consumption and are rarely available as individual chips to, e.g., place them on printed circuit boards (PCBs) that have to be built to satisfy the space constraints in robotic systems.

1.2 Field Programmable Gate Arrays

One solution approach to overcome these problems is based on Field Programmable Gate Arrays (FPGAs). FPGAs consist of generic logic elements that can be configured to form specific circuits to implement an algorithm *in hardware*. This has several advantages for machine learning and robotics. First, the hardware implementation of an algorithm can provide significant performance improvements

while keeping the power consumption low. Second, since the configuration process is very flexible, the disadvantages of ASICs regarding development costs do not apply here. The algorithm can be modified if needed, since the circuit can be changed by a reconfiguration of the FPGA.

Traditionally, FPGAs were used as simple, but flexible logic elements or to provide other simple functionalities in electronic devices. However, in the last couple of years the application areas were extended to other fields, such as digital signal processing. Vendors integrate components into FPGAs to further improve the usability in these application areas. Examples are DSP slices such as the DSP48 slice in Xilinx FPGAs [4] to efficiently perform multiply-accumulate operations, or memory elements such as block RAMs [5] to buffer data. A further advantage is the inherent real time capability of the FPGA. It is possible to design the circuit in such a way that it executes an algorithm in an exact number of clock cycles to meet time constraints.

However, these advantages are not for free: a major problem of FPGAs is the design complexity that requires careful attention of the FPGA designer. For example, the designer has to decide for every arithmetic operation, if it should be performed as a single or double precision floating or even fixed point operation. To save resources, the latter is preferred, but this can result in numerical problems. Furthermore, the exact timing of all operations has to be specified and the circuit must be made accessible for the software side. Up to now, this design complexity has prevented FPGAs from being widely used in the machine learning community.

1.3 FPGAs for Machine Learning

Often, FPGAs are still used as *classical* electronic components: to provide glue logic or otherwise simple functionality like low-level communication. However, there are various approaches that use FPGAs for increasingly more complex tasks that range from simple signal processing to complex control architectures.

Furthermore, FPGA implementations for different popular machine learning applications are presented in the literature, e.g. neural networks [6] or support vector machines [7]. However, most approaches are *singular*, i.e., they conduct just a *single* functionality, without any possibility of generalization or transferability to other applications. A generic framework for machine learning and robotics has been proposed in [8]. However, due to its design, it is only suitable for stationary high performance systems, but not for robotics. Furthermore, the framework is only usable in a concrete hardware setup. In order to facilitate the usage of FPGAs for the development of innovative machine learning-based applications in future robotic and mobile systems, the typical *engineering* problems have to be *hidden* or *simplified*. In order to achieve this, the following points are of importance:

1. The possibility to rapidly implement typical algorithms in the field of machine learning as hardware accelerators.
2. It should be possible to easily integrate third party implementations, so called intellectual property (IP) cores, into the framework.

3. One should be able to easily verify the functionality of the hardware accelerator.
4. Mechanisms to simplify the accessibility to the hardware accelerator from the software side have to be provided.
5. Mechanisms to automatically optimize the design regarding required FPGA-resources, given a set of defined constraints, are required.

In this paper, we discuss an approach to meet these requirements by proposing the *re*configurable *S*ignal *P*rocessing *A*nd *C*lassification *E*nvironment (*reSPACE*). It is designed to reduce the complexity of development while retaining the most important advantages especially in the field of machine learning and signal processing, the later is especially relevant since often an appropriate feature extraction is of high importance [9]). We illustrate the properties of reSPACE on an example of biomedical signal processing, namely the realtime prediction of movements based on single trial analysis of the human electroencephalogram. In future, this can for example be used in the field of rehabilitation robotics embedded in a wearable assistive robotic device.

2 Accelerator Hardware Architecture

The proposed framework is based on the *static heterogeneous synchronous dataflow* computing paradigm. A dataflow-like concept is used in frameworks that are popular for machine learning such as MDP [10], scikits-learn [11] or pySPACE [12]. Using reSPACE, it is easy to implement FPGA-based application specific dataflow accelerators (DFAs) that speed up machine learning operations.

2.1 Heterogeneous Synchronous Dataflow Computing Paradigm

In the dataflow computing paradigm, data is streamed through a sequence of algorithms and transformed on its way through them [13]. In the following, we call this sequence a *flow*, and the implementations of the specific algorithms *nodes*. Since it is possible to combine different nodes, we use a *heterogeneous* dataflow paradigm. Since the structure is pre-defined for a specific application, it is a *static* dataflow. However, in contrast to software dataflow concepts, the data is shifted by one step through the flow on each clock tick of the system, resulting in a *synchronous* design.

In reSPACE, we provide two different operating modes of the system. The *stream mode* allows to process an ongoing stream of data. This mode is required when the signal for processing originates directly from analog to digital converters, e.g., force-torque sensors or electrodes for reading biosignals. The other operating mode is designed to process separate *windows* of data, where the samples within a window are adjacent to each other. Examples are feature vectors consisting of single feature elements or images that consist of pixels. We use a model-based design approach here that is currently based on System Generator for DSP [14] or VHDL. To *implement a complete system*, we provide two different alternatives.

Either a *library of predefined, parametrized nodes* can be used, that contains basic, widely used algorithms such as FIR and IIR filters, direct current offset removal, standardization, etc., are provided as directly usable nodes that can be directly arranged to build up the overall system. These nodes are generic and can be instantiated using different sets of parameters. This allows the designer to configure the overall system for, e.g., different number of channels or different precision of the calculations.

The other opportunity is *customizable circuit generation for matrix-multiply based algorithms*. We provide mechanisms to generate specific circuits for resource efficient parallel matrix operations. The multiply accumulate operations are mapped to the DSP slices of the FPGA. This is done using a domain specific language that allows to specify a sequence of matrix vector operations and choose a parallel or serial implementation. The parallel implementation uses a minimal number of clock cycles but high amount of logic resources, whereas the serial implementation is resource efficient but takes more cycles to run.

2.2 System Architecture

There are at least two possibilities for the integration of DFAs into a system:

Inside a *System on Chip (SoC)* In this setup, the DFA is connected to a host CPU by some type of bus system, e.g., an AXI bus [15]. This setup is applicable if the main system is controlled by software that is running on the host CPU.

Direct access This setup is useful if certain processing should be performed in a decentralized way. Applications are sensor data processing in proximity to a sensor to, e.g., perform *complex* preprocessing or dimensionality reduction of the data or to implement *intelligent* control algorithms.

In the *SoC setup*, the DFA has to be accessed from software. The software is responsible to either transfer the data or results to and from the DFA or to initialize the transfer if direct memory access is used. For this, *device drivers* are required. The implementation of device drivers is a tedious and error prone task. Therefore, reSPACE supports this task by using automatic driver and middleware generation (see Sec. 3). In contrast, the *direct access setup* requires that the DFA has to be accessed directly from other hardware components or from a remote hardware system via low-level communication interfaces. Here, reSPACE hides the technical details and allows to focus on the algorithm development.

2.3 Limitations

A general problem for FPGA-accelerated machine learning operations is the amount of available memory. The amount of data, that can be stored in the available block RAMs of even the current high-end FPGA devices, is in the range of some dozens of MB [16]. Hence, the storage of large amounts of data *inside the FPGA itself* is usually not feasible in practice, and external memory is needed. Another algorithmic solution approach is the usage of incremental or online learning methods.

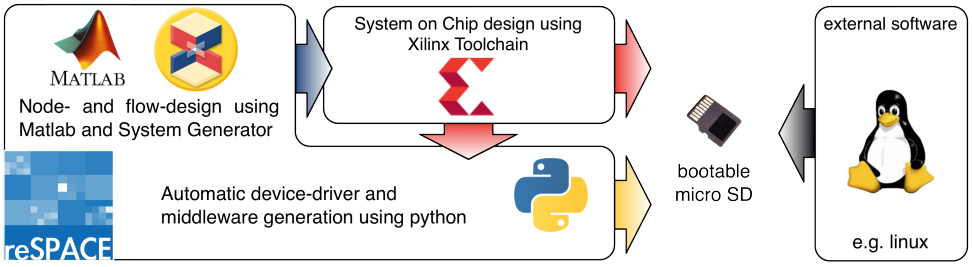


Fig. 1. Workflow for hardware accelerator integration. The processing-flow created by Matlab and System Generator gets integrated into the SoC using the Xilinx toolchain. Output products of the Xilinx toolchain are used for automatic driver and middleware generation. Third party software (e.g., bootloader and kernel) can be added to, e.g., get a bootable medium.

3 Software Infrastructure and Integration

In the in SoC setup, the DFA is used to accelerate a specific software task. Usually, an operating system, like Linux, is running on the host CPU. Hence, to access the DFA from a software application, device drivers are needed that interface with the DFAs and run as kernel modules.

3.1 Automatic Software Generation

In the given context, however, these drivers have a limited duty: the transfer of the data from main memory to the DFA and collection of the results, while the internal *business logic* of the driver is neglectable and can be implemented in the user space. If the systems physical memory-map is known, it is possible, to generate the required drivers *fully automatically*.

The same approach can be used to automatically generate additional interface libraries to other higher-level languages, such as Python. Consequently, the DFA can be directly used from popular machine learning frameworks with a minimum of effort. Currently, we provide automatic generation of code to integrate reSPACE nodes into pySPACE [12] to allow the use of software-centric mechanism like cross validation and performance evaluation. We refer to this process as *automatic driver and middleware generation*. For details about this software generation process, see Fig. 1.

3.2 Functional Verification

Still, the design and implementation of DFAs can be a complex and error-prone task, due to the FPGA properties, like cycle-accurate timing and numerical issues due to fixed-point computations that need to be solved. Accordingly, a high effort is required in order to verify and ensure the functionality of the DFAs. In reSPACE, we use a verification approach that is based on pySPACE. In pySPACE, it is straightforward to generate test data and information regarding

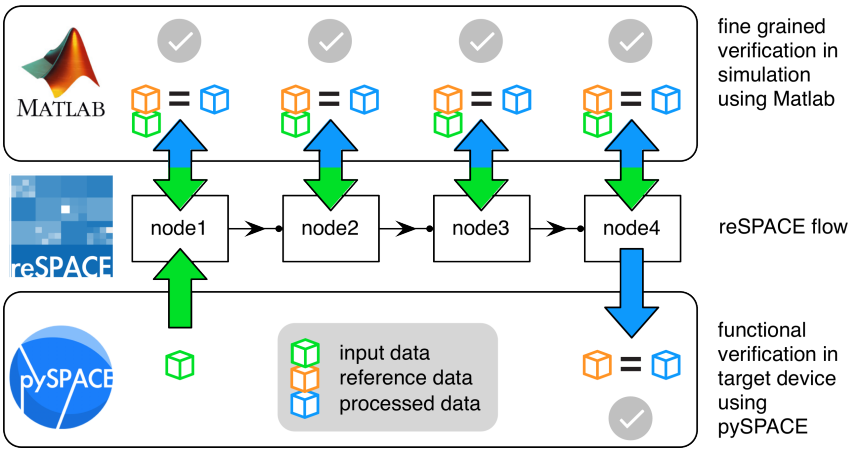


Fig. 2. Dataflow paradigm and verification for reSPACE generated flows. In simulation, the flow can be verified on node level whereas the final implementation inside the FPGA fabric is verified by comparing the final processed data with precomputed reference data inside pySPACE.

the configuration of the specific nodes. The pySPACE flows can be augmented with helper nodes that monitor and store all data persistently that is passed through them.

Simulation Verification Our tools for hardware verification build on this functionality of pySPACE. The intermediate data can be used directly to generate testbenches. These allow the verification of the hardware accelerator in simulation and an in-depth investigation and analysis of any differences between the pySPACE results and intermediate results of the dataflow hardware accelerator. These might occur due to, e.g., fixed-point computations that are usually employed in FPGAs.

Hardware Verification Besides the simulation based verification, reSPACE also supports verification directly on the target system. There, we use a hierarchical approach to verify the availability and functionality of the dataflow hardware accelerator. First, it is tested if the Linux device file is accessible and reachable from the software side. Second, a dedicated test code can be read from the device to check if the dataflow hardware accelerator is available. Finally, the persistent test data is used to test the dataflow hardware accelerator for full functionality. Therefore, the test data is used for stimulation and comparison with the results of the hardware accelerator.

4 Applications

In this section, we outline how the proposed framework can be used for different application areas. We briefly show results for online electroencephalogram (EEG) analysis as a more extensive example.

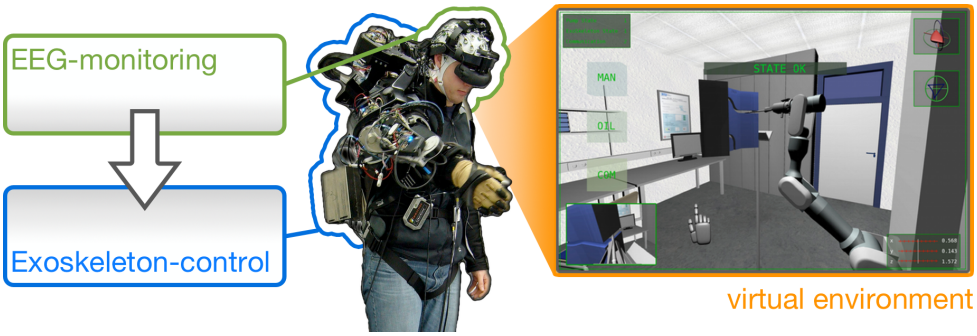


Fig. 3. Biosignal-augmented exoskeleton. The operator controls a robotic arm in a virtual environment (right). The task is to move the end-effector of a robotic arm like in an *hot wire game* through a labyrinth without touching it. For control, the operator wears an exoskeleton that maps the position of the operator's hand to the end-effector. The operator's EEG is continuously analyzed to detect upcoming movements that can be used to adapt the control algorithms of the exoskeleton.

4.1 Biomedical Signal Processing for Teleoperation and Rehabilitation Robotics

Performing teleoperation of robotic systems using current input devices like joysticks is usually a demanding task. To simplify this, the exoskeletons can be used as more intuitive command interfaces. The details of the investigatio have been described in [17, 18] and are depicted in Fig. 3. To enhance the movability of the exoskeleton, the joint control algorithms can be enhanced by integrating predictions of upcoming movements based on the detection of movement-related cortical potentials [19]. For the predictions, the EEG of the operator is continuously analyzed by a movement prediction system, which performs a prediction each 50 ms. EEG data is high dimensional (64 channels sampled at 5kHz), require a range of different signal processing and machine learning operations to detect the to upcoming movements, and fast computations, since the processing has to be finished *before* a movement is executed. Since the operator should be able to move freely, the exoskeleton has to be be independent from stationary hardware. In future, the exoskeleton will also be used as a rehabilitation systems [20]. Therefore, all computations should be performed in devices that are *embedded* in the exoskeleton, were reSPACE can be used to map the operations to FPGAs to provide the necessary computational power. For the detections, different signal processing operations have to be applied to the data *online*. We use the following procedure: DC offset removal, anti-alias filtering and downsampling to 20 Hz, spatial filtering, feature vector extraction, standardization, and classification using a passive-aggressive perceptron, whose parameter c was optimized using a grid search. We realized this twice using pySPACE and reSPACE and compare the classification and computation performance for both approaches.

The obtained results are shown in Table 1. It can be observed that there exist no major differences regarding classification performance. However, by using the DFA an $\approx 7\times$ speedup is achieved.

Table 1. Application oriented metrics for online and pseudo-online sessions in terms of Balanced Accuracy (BA) and computation time for 10 recording sessions on 3 subjects. The BA is defined as the mean of true positive and negative rates. The two different devices were a mobile processor (MP, ARM Cortex A9, 666 Mhz) and a combined mobile processor with DFA (running at 100 Mhz). All computations on the MP were performed as double precision floating point operations, all computations in the DFA used fixed point arithmetic. The reported computation times are the amount of time that is required to process 1s of EEG data for the different processing platforms.

	Performance Mobile CPU	Performance Mobile CPU + DFA
BA (%)	76.012 ± 4.106	75.717 ± 4.018
Computation time (ms)	1199.943 ± 7.170	174.403 ± 5.872

4.2 Further Application Areas

There are various other application areas in machine learning, robotics and autonomous systems that would gain a substantial benefit from FPGA-based accelerators. Obvious examples are various image processing techniques, such as SURF generation [21] and traffic sign detection [22] or enhancing the control of robots [23].

5 Conclusion and Future Work

In this paper, we discussed the necessity, requirements and state of the art of FPGA-based machine learning accelerators that should be employed in mobile and robotic systems. As a solution to the current problems, we proposed our framework reSPACE that supports the implementation of such accelerators. We described the techniques that we use in reSPACE to simplify the design process in order to allow algorithm developers to use FPGAs and to verify the resulting hardware implementations in simulation or in the final system. In future, we will enhance the framework to improve the usability further. First, we want to transfer all components to pure VHDL implementations to achieve a higher degree of vendor and third party independence, and to provide the framework as open source to allow it to be used easily by machine learning and robotics researchers.

References

1. Carroll, A., Heiser, G.: An analysis of power consumption in a smartphone. In: Proceedings of the 2010 USENIX conference. (2010) 21–21
2. Zhang, Z.: Microsoft kinect sensor and its effect. MultiMedia, IEEE **19**(2) (2012) 4–10
3. Steinkrau, D., Simard, P.Y., Buck, I.: Using gpus for machine learning algorithms. In: Proceedings of the Eighth International Conference on Document Analysis and Recognition, IEEE Computer Society (2005) 1115–1119

4. Xilinx Corporation: UG479: 7 Series DSP48E1 Slice User Guide (2014)
5. Xilinx Corporation: DS444: IP Processor Block RAM, Product Specification (2011)
6. Omondi, A.R., Rajapakse, J.C.: FPGA implementations of neural networks. Springer (2006)
7. Anguita, D., Boni, A., Ridella, S.: A digital architecture for support vector machines: theory, algorithm, and fpga implementation. Neural Networks, IEEE Transactions on **14**(5) (2003) 993–1009
8. Graf, H.P., Cadambi, S., Durdanovic, I., Jakkula, V., Sankaradass, M., Cosatto, E., Chakradhar, S.T.: A massively parallel digital learning processor. In: NIPS. (2008) 529–536
9. Domingos, P.: A few useful things to know about machine learning. Communications of the ACM **55**(10) (2012) 78–87
10. Zito, T., Wilbert, N., Wiskott, L., Berkes, P.: Modular toolkit for Data Processing (MDP): a Python data processing framework. Frontiers in Neuroinformatics **2**(8) (2008)
11. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12** (2011) 2825–2830
12. Krell, M.M., Straube, S., Seeland, A., Wöhrle, H., Teiwes, J., Metzen, J.H., Kirchner, E.A., Kirchner, F.: pySPACE - a signal processing and classification environment in Python. Frontiers in Neuroinformatics **7**(40) (2013)
13. Flagg, M.: Dataflow principles applied to real-time multiprocessing. In: COMPCON Spring'89. Thirty-Fourth IEEE Computer Society International Conference: Intellectual Leverage, Digest of Papers., IEEE (1989) 84–89
14. Xilinx Corporation: UG640: System Generator for DSP User Guide (2012)
15. ARM: AMBA AXI and ACE Protocol Specification (2013)
16. Altera Corporation: Stratix V Device Overview (2014)
17. Folgheraiter, M., Kirchner, E.A., Seeland, A., Kim, S.K., Jordan, M., Woehrle, H., Bongardt, B., Schmidt, S., Albiez, J., Kirchner, F.: A multimodal brain-arm interface for operation of complex robotic systems and upper limb motor recovery. In: Proc. of the 4th International Conference on Biomedical Electronics and Devices (BIODEVICES-11), Rome (2011) 150–162
18. Folgheraiter, M., Jordan, M., Straube, S., Seeland, A., Kim, S.K., Kirchner, E.A.: Measuring the improvement of the interaction comfort of a wearable exoskeleton. International Journal of Social Robotics **4**(3) (2012) 285–302
19. Seeland, A., Woehrle, H., Straube, S., Kirchner, E.A.: Online movement prediction in a robotic application scenario. In: 6th International IEEE EMBS Conference on Neural Engineering (NER), San Diego, California (2013) 41–44
20. Kirchner, E.A., Albiez, J., Seeland, A., Jordan, M., Kirchner, F.: Towards assistive robotics for home rehabilitation. In Chimeno, M.F., Solé-Casals, J., Fred, A., Gamboa, H., eds.: Proceedings of the 6th International Conference on Biomedical Electronics and Devices (BIODEVICES-13), Barcelona, ScitePress (2013) 168–177
21. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In: Computer Vision–ECCV 2006. Springer (2006) 404–417
22. Zaklouta, F., Stanciulescu, B.: Real-time traffic sign recognition in three stages. Robotics and Autonomous Systems **62**(1) (2014) 16–24
23. Langosz, M., von Szadkowski, K., Kirchner, F.: Introducing particle swarm optimization into a genetic algorithm to evolve robot controllers. In: Proceedings of the 16th Annual Conference on Genetic and Evolutionary Computation. GECCO '14, ACM (2014)

Multivariate Normal Distribution Based Multi-Armed Bandits Pareto Algorithm

Saba Q. Yahyaa, Madalina M. Drugan and Bernard Manderick

Vrije Universiteit Brussel, Department of Computer Science,
Pleinlaan 2, 1050 Brussels, Belgium
{syahyaa,mdurgan,bmanderi}@vub.ac.be

Abstract. In the stochastic multivariate multi-armed bandit, arms generate a vector of stochastic normal rewards, one per objective, instead of a single scalar reward. As a result, there is not only one optimal arm, but there is a set of optimal arms (Pareto front) using Pareto dominance relation. The goal of an agent is to trade-off between exploration and exploitation. Exploration means finding the Pareto front and exploitation means selecting fairly or evenly the optimal arms. We propose annealing-Pareto algorithm that trades-off between exploration and exploitation by using a decaying parameter ϵ_t in combination with Pareto dominance relation. We compare experimentally Pareto-KG, Pareto-UCB1 and annealing-Pareto on multi-objective normal distributions and we conclude that the annealing-Pareto is the best performing algorithm.

Keywords: Multi armed bandit problem, multi objective optimization, annealing algorithm, exploration/exploitation.

1 Introduction

The Multi-Objective Multi-Armed Bandit (MOMAB) problem is a sequential stochastic learning problem. At each time step t, an agent pulls one arm i from an available arm set A and receives a reward vector $\boldsymbol{r}_i$ of the arm i with D variates (or objectives) as feedback signal. The reward vector is drawn from a normal probability distribution vector $N(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2)$, where $\boldsymbol{\mu}_i$ is the true mean vector and $\boldsymbol{\sigma}_i^2$ is the covariance matrix parameters of the arm i. The reward vector $\boldsymbol{r}_i$ that the agent receives from the arm i is independent from all other arms and independent from the past reward vectors of the selected arm i. Moreover, the mean vector of the arm i has *independent* D distributions, i.e. $\boldsymbol{\sigma}^2$ is a *diagonal covariance matrix*. We assume that the true mean vector and covariance matrix of each arm i are unknown parameters to the agent. Thus, by drawing each arm i, the agent maintains estimations of the true mean vector and the diagonal covariance matrix (or the variance vector) which are known as $\hat{\boldsymbol{\mu}}_i$ and $\hat{\boldsymbol{\sigma}}_i^2$, respectively.

The MOMAB problem has a set of Pareto optimal arms (Pareto front) A^*, that are incomparable, i.e. can not be classified using a designed partial order relations. The agent has not to only find the optimal arms (exploring), to minimize the total Pareto loss of not pulling the optimal arms, but also has to play

them fairly (exploiting), to minimize the total unfairness loss. This problem is known as the *trade-off between exploration and exploitation in the multi-objective optimization* [1]. At each time step t, the Pareto loss (or Pareto regret) is the distance between the set mean of Pareto optimal arms and the mean of the selected arm. While, the unfairness loss (or unfairness regret) is the variance in selecting the optimal arms [2]. Thus, the total Pareto regret and the total unfairness regrets are the cumulative summation of the Pareto and unfairness regret over t time steps, respectively. Since, the total unfairness regret grows exponentially on the number of time steps and does not take into account the total number of selecting optimal arms, we propose to compute the unfairness regret using the *entropy measure* [3]. The entropy unfairness regret is a measure of disarray (or disorder) on selecting the optimal arms in the Pareto front A^*.

The Pareto front A^* can be found for example, by using Pareto dominance relation (or Pareto partial order relation) which finds the Pareto front A^* by optimizing directly the Multi-Objective (MO) space [4]. To solve the trade-off between exploration and exploitation problem directly in the MO space, [2] used Upper Confidence Bound (UCB1) [5] policy and [6] used Knowledge Gradient (KG) [7] policy in the MOMAB problem. Both UCB1 and KG policies trade-off between exploration and exploitation by adding an exploration term (or bound) to the estimated mean vector $\hat{\boldsymbol{\mu}}_i$ for each arm i in each objective (or dimension) $d, d \in D$ and select the optimal arms by using Pareto dominance relation. However, the exploration bound of UCB1 for arm i requires only knowledge about that arm, while in case of KG it also requires knowledge about the other arms.

In this paper, we propose annealing-Pareto algorithm that detects the optimal arms in the multi-objective space. The annealing-Pareto controls the trade-off between exploration and exploitation by using a decaying parameter ϵ_t, $\epsilon_t \in (0, 1)$ in combination with the Pareto dominance relation. The decaying parameter ϵ_t has a high value at the beginning of time step t to explore all the available arms and increase the confidence in the estimated means, however, as the time step t increases, the ϵ_t parameter decreases to exploit the arms that have maximum estimated mean. To keep track on all the optimal arms in the Pareto front A^*, at each time step t, the annealing-Pareto uses Pareto dominance relation.

The rest of the paper is organized as follows: In Section 2 we introduce the multivariate normal multi-armed bandit problem. In Section 3 we present the MOMAB algorithms for normal multivariate distributions. In Section 4 we present the performance measure in the MOMAB problem. In Section 5 we introduce the annealing-Pareto algorithm in normal distribution. In Section 6, we describe the experiments set up followed by experimental results. Finally, we conclude and discuss future work.

2 Multi Objective Normal Distributions Multi Armed Bandits Problem

Let us consider the MOMABs problems with $|A| \geq 2$ arms and with *independent* D objectives per arm. At each time step t, the agent selects one arm i

and receives a reward vector $\boldsymbol{r}_i$. The reward vector $\boldsymbol{r}_i$ is drawn from a corresponding normal probability distribution $N(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2)$ with unknown mean parameter vector $\boldsymbol{\mu}_i$, $\boldsymbol{\mu}_i = [\mu_i^1, \cdots, \mu_i^D]^T$ and unknown variance parameter vector $\boldsymbol{\sigma}_i$, $\boldsymbol{\sigma}_i = [\sigma_i^1, \cdots, \sigma_i^D]^T$, where T is the transpose. Thus, by drawing each arm i, the agent maintains estimate of the mean parameter vector $\hat{\boldsymbol{\mu}}_i$ and the variance $\hat{\boldsymbol{\sigma}}_i^2$ parameter vector, and computes the number of times N_i arm i is drawn. The agent updates the estimated mean $\hat{\mu}_i^d$, the estimated variance $\hat{\sigma}_i^{2,d}$ of the selected arm i in each dimension $d, d \in D$ and the number of times N_{i+1} arm i has been selected as follows [8]:

$$N_{i+1} = N_i + 1, \quad \hat{\mu}_{i+1}^d = (1 - \frac{1}{N_{i+1}})\, \hat{\mu}_i^d + \frac{1}{N_{i+1}} r_{t+1}^d \tag{1}$$

$$\hat{\sigma}_{i+1}^{2,d} = \frac{N_{i+1} - 2}{N_{i+1} - 1}\, \hat{\sigma}_i^{2,d} + \frac{1}{N_{i+1}} (r_{t+1}^d - \hat{\mu}_i^d)^2 \tag{2}$$

where $\hat{\mu}_{i+1}^d$ is the updated estimated mean, and $\hat{\sigma}_{i+1}^{2,d}$ is the updated estimated variance of the arm i in the dimension d and r_{t+1}^d is the observed reward of the arm i in the dimension d.

When the objectives are conflicting with one another then the mean component μ_i^d of arm i corresponding with objective d, $d \in D$, can be better than the component μ_j^d of another arm j but worse if we compare the components for another objective d': $\mu_i^d > \mu_j^d$ but $\mu_i^{d'} < \mu_j^{d'}$ for objectives d and d', respectively. The agent has a set of optimal arms (Pareto front) A^* which can be found by the Pareto dominance relation (or Pareto partial order relation).

The Pareto dominance relation finds the Pareto front A^* directly in the multi-objective MO space [4]. It uses the following relations between the mean vectors of two arms. We use i and j to refer to the mean vector (estimated mean vector or true mean vector) of arms i and j, respectively:

Arm i dominates or is better than j, $i \succ j$, if there exists at least one objective d for which $i^d \succ j^d$ and for all other objectives d' we have $i^{d'} \succeq j^{d'}$. Arm i is incomparable with j, $i \parallel j$, if and only if there exists at least one objective d for which $i^d \succ j^d$ and there exists another objective d' for which $i^{d'} \prec j^{d'}$. Arm i is not dominated by j, $j \not\succ i$, if and only if there exists at least one objective d for which $j^d \prec i^d$. This means that either $i \succ j$ or $i \parallel j$.

Using the above relations, Pareto front $A^*, A^* \subset A$ be the set of arms that are not dominated by all other arms. Moreover, the optimal arms in A^* are incomparable with each other.

3 Multi Objective Multi Armed Bandits Algorithms in Normal Distribution

Pareto-UCB1 [2] and Pareto-KG [6] trade-off between exploration and exploitation by combination one-objective, Multi-Armed Bandits (MAB) algorithms (or policies) with Pareto dominance relation.

3.1 Pareto-UCB1 in Normal distribution

Pareto-UCB1 is the extension of the UCB1 policy [5] to the MOMABs. Pareto-UCB1 plays initially each arm i once. At each time step t, it estimates the mean vector of each of the multivariate arms i, i.e. $\hat{\boldsymbol{\mu}}_i = [\hat{\mu}_i^1, \cdots, \hat{\mu}_i^D]^T$ and adds to each dimension d an upper confidence bound which represents the *exploration bound* ExpB_i^d, $\text{ExpB}_i^d = \sqrt{(2\ln(t\sqrt[4]{D|A^*|}))/N_i}$ in the dimension d to trade-off between exploration and exploitation, where D is the number of objectives, $|A^*|$ is the number of optimal arms, and N_i is the number of times arm i has been selected. Pareto-UCB1 uses a Pareto dominance relation, Section 2 to find the Pareto-UCB1 optimal arm set A^*_{UCB1}. Thus, for all the non-optimal arms $k \notin A^*_{UCB1}$ there exists a Pareto optimal arm $j \in A^*_{UCB1}$ that is not dominated by the arms k, i.e. $\hat{\boldsymbol{\mu}}_k + \mathbf{ExpB}_k \nsucc \hat{\boldsymbol{\mu}}_j + \mathbf{ExpB}_j$, where $\mathbf{ExpB}_j$, $\mathbf{ExpB}_j = [\text{ExpB}_j^1, \cdots, \text{ExpB}_j^D]$ is the exploration bound vector of the arm j. Pareto-UCB1 selects uniformly randomly one of the arms in the set A^*_{UCB1}. The idea is to select most of the times one of the optimal arm in the Pareto front, $i \in A^*$. An arm $j \notin A^*$ that is closer to the Pareto front according to metric measure is more selected than the arm $k \notin A^*$ that is far from A^*. After pulling the chosen arm i, Pareto-UCB1, updates the estimated mean $\hat{\boldsymbol{\mu}}_i$ vector, the number of times arm i is chosen N_i and computes the Pareto and the unfairness regrets.

3.2 Pareto-KG in Normal distribution

Pareto-KG is the extension of the KG policy [7] to the MOMABs. Pareto-KG plays each arm initial *Steps*. At each time step t, Pareto-KG calculates an exploration bound $\mathbf{ExpB}_i$, $\mathbf{ExpB}_i = [\text{ExpB}_i^1, \cdots, \text{ExpB}_i^D]^T$ for each arm i. The exploration bound of arm i depends on the estimated mean of all arms and on the estimated standard deviation of the arm i. The exploration bound of arm i for dimension d (ExpB_i^d) is calculated as follows:

$$\text{ExpB}_i^d = (L-t) * |A| D * v_i^d, \quad v_i^d = \hat{\bar{\sigma}}_i^d \; x \left(-|\frac{\hat{\mu}_i^d - \max\limits_{j \neq i,\, j \in A} \hat{\mu}_j^d}{\hat{\bar{\sigma}}_i^d}| \right), \quad \forall_{d \in D} \quad (3)$$

where v_i^d is the index of an arm i for dimension d, L is the horizon of experiment which is the total number of time steps, $|A|$ is the total number of arms, and $\hat{\bar{\sigma}}_i^d$, $\hat{\bar{\sigma}}_i^d = \hat{\sigma}_i^d / \sqrt{N_i}$ is the root mean square error of an arm i for dimension d. After computing the exploration bound for each arm, Pareto-KG sums the exploration bound of arm a with the corresponding estimated mean. Thus, Pareto-KG selects the optimal arms j that are not dominated by all other arms $k, k \in |A|$ using Pareto dominance relations, $\hat{\boldsymbol{\mu}_k} + \mathbf{ExpB}_k \nsucc \hat{\boldsymbol{\mu}_j} + \mathbf{ExpB}_j$, Section 2 Pareto-KG chooses uniformly randomly one of the optimal arms in A^*_{KG}, where A^*_{KG} is the Pareto-KG optimal arm set. After pulling the chosen arm i, Pareto-KG, updates the estimated mean $\hat{\boldsymbol{\mu}}_i$, and the estimated variance $\hat{\boldsymbol{\sigma}}_i^2$ vectors, the number of times arm i is chosen N_i and computes the Pareto and the unfairness regrets.

Pareto-UCB1 and Pareto-KG control the trade-off between exploration and exploitation by adding an exploration bound ExpB_i^d to the estimated mean μ_i^d of

each arm i in each objective d. The added exploration bound ExpB_i^d for the arm i in the objective d by Pareto-KG depends on the estimated mean of all available arms in the objective d and on the root mean square error $\hat{\hat{\sigma}}_i^d$ of the arm i, i.e. each objective has different exploration bound. While, the added exploration bound ExpB_i^d for the arm i in the dimension d by Pareto-UCB1 depends only on the arm i, i.e. each objective has the same exploration bound.

4 Performance Measure

In the MOMAB, the agent has not only to find the Pareto front A^* (or exploring the optimal arms), but also has to play them fairly (or exploiting) the optimal arms). As a result, there are two regret measures.

Pareto regret measure (R_{Pareto}) [2] measures the distance between a mean vector of an arm i that is pulled at time step t and the Pareto front A^*. Pareto regret R_{Pareto} is calculated by finding firstly the virtual distance dis^*. The virtual distance dis^* is defined as the minimum distance that is added to the mean vector of the pulled arm $\boldsymbol{\mu}_t$ at time step t in each dimension to create a virtual mean vector $\boldsymbol{\mu}_t^*$, $\boldsymbol{\mu}_t^* = \boldsymbol{\mu}_t + \boldsymbol{\varepsilon}^*$ that is incomparable with all the arms in Pareto set A^*, i.e. $\boldsymbol{\mu}_t^* || \boldsymbol{\mu}_i \; \forall_{i \in A^*}$. Where $\boldsymbol{\varepsilon}^*$ is a vector, $\boldsymbol{\varepsilon}^* = [dis^{*,1}, \cdots, dis^{*,D}]^T$. Then, the Pareto regret R_{Pareto}, $R_{Pareto} = dis(\boldsymbol{\mu}_t, \boldsymbol{\mu}_t^*) = dis(\boldsymbol{\varepsilon}^*, \mathbf{0})$ is the distance between the mean vector of the virtual arm $\boldsymbol{\mu}_t^*$ and the mean vector of the pulled arm $\boldsymbol{\mu}_t$ at time step t, where dis, $dis(\boldsymbol{\mu}_t, \boldsymbol{\mu}_t^*) = (\sum_{d=1}^{D} (\mu_t^{*,d} - \mu_t^d)^2)^{(1/2)}$ is the Euclidean distance. Thus, the regret of the Pareto front is 0 for optimal arms, i.e. the mean of the optimal arm coincides itself.

The unfairness regret metric is the *Shannon's entropy measure* [3] which is a measure of disorder (or disarray) on the Pareto front A^*. The higher the entropy, the higher the disorder. At time step t, the Shannon regret is $R_{SE}(t)$, $R_{SE}(t) = -\frac{1}{N_{|A^*|}(t)} \sum_{i^* \in A^*} p_{i^*}(t) \ln(p_{i^*}(t))$, where $p_{i^*}(t)$, $p_{i^*}(t) = {}^{N_{i^*}(t)}/_{N(t)}$ is the probability of selecting an optimal arm i^* at time step t, where $N_{i^*}(t)$ is the number of times the optimal arm i^* has been selected and $N(t)$ is the number of times all arms $i = 1, \cdots, A$ have been selected at time step t, and $N_{|A^*|}(t)$ is the number of times the optimal arms, $i^* = 1, \cdots, |A^*|$ have been selected at time step t.

5 The Annealing-Pareto Algorithm

Annealing-Pareto algorithm has a specific mechanism to control the trade-off between exploration and exploitation. It uses an exponential decay ϵ_t, $\epsilon_t = {}^{\epsilon_{decay}^t}/_{(|A|D)}$, where ϵ_{decay} is the decay parameter and Pareto dominance relation. At the beginning of time step t, ϵ_t has a high value to explore all the available arms. As the time step t is increased, ϵ_t has a low value to exploit only the optimal arms. To keep track on all the optimal arms in the Pareto front A^*, the annealing-Pareto uses Pareto dominance relation. The decay parameter ϵ_{decay}, $\epsilon_{decay} \in (0, 1)$, when $\epsilon_{decay} = 0$ means the annealing-Pareto is a fully Pareto dominance relation and when $\epsilon_{decay} = 1$ means the annealing-Pareto uses a fixed exponential decay. The pseudocode of the annealing-Pareto is given in Algorithm 1.

As initialization step, Algorithm 1 plays each arm i once to estimate the corresponding mean vector $\hat{\boldsymbol{\mu}}_i$ and the ϵ-Pareto optimal arm set A^*_ϵ contains all the arms in the arm set A. At each time step t, Algorithm 1 trades-off between exploration and exploitation by using the decay parameter ϵ_{decay} in the exponential decay ϵ_t (step: 4). In each objective d, $d \in D$, the Algorithm 1 detects the optimal arm in that objective $i^{*,d}$, $i^{*,d} = \text{argmax}_{i=1,\cdots,A}\, \hat{\mu}^d_i$, where $\hat{\mu}^d_i$ is the estimated mean for arm i in the dimension d (step: 7). Algorithm 1 selects all the arms in the objective d that have estimated mean between $[\hat{\mu}^{*,d} - \epsilon_t, \hat{\mu}^{*,d}]$ and include them in the corresponding selected arm set S^d (steps: 8-12), where $\hat{\mu}^{*,d}$, $\hat{\mu}^{*,d} = \max_{i \in A} \hat{\mu}^d_i$ is the estimated mean of the optimal arm $i^{*,d}$ in the objective d. Algorithm 1 constructs the total selected arm set $S(t)$ at time step t by reunion of the selected arm set (step: 14). To keep track on the Pareto front A^*, the Algorithm 1 uses Pareto dominance relation (step: 17) on the arms j that are elements in the previous ϵ-Pareto optimal arm set $A^*_\epsilon(t-1)$ and are not element in the total selected arm set $S(t)$. If the arm j is not dominated by all other arms, then this arm will be added to the total selected arm set $S(t)$ (step: 18). Algorithm 1 updates its ϵ-Pareto optimal arm set $A^*_\epsilon(t)$ to be the total selected arm set $S(t)$ (step: 21). It pulls uniformly at random one of the arms i^* that is an element in the ϵ-Pareto optimal arm set $A^*_\epsilon(t)$ (step: 22), observes the corresponding reward vector $\boldsymbol{r}_{i^*}$ and updates its estimated mean vector $\hat{\boldsymbol{\mu}}_{i^*}$ and the number of times N_{i^*} arm i^* is selected (step: 23). Then, it calculates the Pareto and unfairness regrets. This procedure is repeated until the end of playing L time steps which is the horizon of an experiment.

In Fig. (1), the dynamic of the algorithm is illustrated on 2-objective 5-armed bandit. The optimal arms a^*_1, a^*_2, and a^*_3 have the means μ^*_1, μ^*_2 and μ^*_3, respectively. The non-optimal arms a_4, and a_5 have the means μ_4 and μ_5, respectively. At the beginning of time step, $t = 1$ the total selected arm set $S(t)$ almost contains all the arms (optimal and non-optimal arms), and the ϵ-Pareto optimal arm set A^*_ϵ contains all the arms as shown in subfigure a. As the time step increases, $S(t)$ contains some of the optimal arms, i.e. a^*_2 as shown in subfigure b and c, therefore, to maintain all the Pareto front, the algorithm constructs its updated ϵ-Pareto optimal arm set $A^*_\epsilon(t)$ to be the set that contains the non dominated arms (a^*_1 and a^*_3) in the previous $A^*_\epsilon(t-1)$ and the arms in the set $S(t)$.

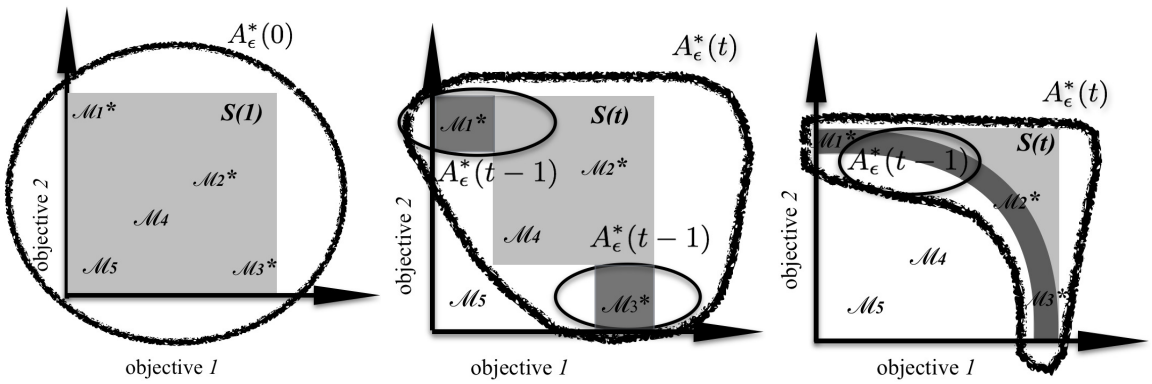
6 Experiments

In this section, we experimentally compare Pareto-UCB1, Pareto-KG and annealing-Pareto. The performance measures are: 1) the cumulative average regret at each time step which are the average of M experiments. 2) the cumulative average unfairness at each time step which are the average of M experiments.

The number of experiments M and the horizon of each experiment L are 1000. The rewards of each arm i in each objective d, $d \in D$ are drawn from normal distribution $N(\boldsymbol{\mu}_i, \boldsymbol{\sigma}^2_{i,r})$ where $\boldsymbol{\mu}_i = [\mu^1_i, \cdots, \mu^D_i]^T$ is the unknown true mean and $\boldsymbol{\sigma}^2_{i,r} = [\sigma^{2,1}_{i,r}, \cdots, \sigma^{2,D}_{i,r}]^T$ is the true unknown variance of the reward. The standard deviation σ^d_r for arms in each objective is set to $0.01, 0.1$ or 1. For

Algorithm 1 (Annealing-Pareto in Normal Distribution)

1. **Input:** Horizon of an experiment L; time step t; number of arms $|A|$; number of objectives $|D|$; reward distribution $r \sim N(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$; selected arm set $S^d(t) = \{\ \}\ \forall_d$; decay parameter $\epsilon_{decay} \in (0, 1)$.
2. **Intialize:** play each arm i initial steps to estimate its mean vector $\hat{\boldsymbol{\mu}}_i = [\hat{\mu}_i^1, \cdots, \hat{\mu}_i^D]^T$; initial ϵ-Pareto front set $A_\epsilon^*(0) = A$.
3. **For time step** $t = 1, \cdots, L$
4. Set the decay parameter $\epsilon_t = \epsilon_{decay}^t / (|A||D|)$
5. **For objective** $d = 1, \cdots, D$
6. $S^d(t) = \{\phi\}$
7. $\hat{\mu}^{*,d} = \max_{1 \le i \le A} \hat{\mu}_i^d$
8. **For arm** $i = 1, \cdots, A$
9. If $\hat{\mu}_i^d \in [\hat{\mu}^{*,d} - \epsilon_t, \hat{\mu}^{*,d}]$
10. $S^d(t) \leftarrow \{S^d(t), i\}$
11. End If
12. **End For**
13. **End For**
14. $S(t) \leftarrow S^1(t) \cup S^2(t) \cup \cdots \cup S^D(t)$
15. $S_{difference} \leftarrow A_\epsilon^*(t-1) - S(t)$
16. **For arm** $j \in S_{difference}$ do
17. If $\hat{\boldsymbol{\mu}}_k \not\succ \hat{\boldsymbol{\mu}}_j, \forall_k \in A$
18. $S(t) \leftarrow S(t) \cup j$
19. End If
20. **End For**
21. $A_\epsilon^*(t) \leftarrow S(t)$
22. Select an optimal arm i^* uniformly, at random from $A_\epsilon^*(t)$
23. Observe: reward vector $r_{i^*}, r_{i^*} = [r_{i^*}^1, \cdots, r_{i^*}^D]^T$;Update: $\hat{\boldsymbol{\mu}}_{i^*}$; $N_{i^*} \leftarrow N_{i^*} + 1$
24. **End For**
25. **Output:** Unfairness regret; Pareto regret



a. At $t = 1$ *b.* At $t > 1$ *c.* At $t >> 1$

Fig. 1. The dynamic of the annealing-Pareto algorithm.

Pareto-UCB1 and the annealing-Pareto, each arm is played initially one time, i.e. $Initial = 1$. Pareto-KG needs the estimated variance for each arm, $\hat{\boldsymbol{\sigma}}_i^2$, therefore, each arm is played initially 2 times which is the minimum number to estimate

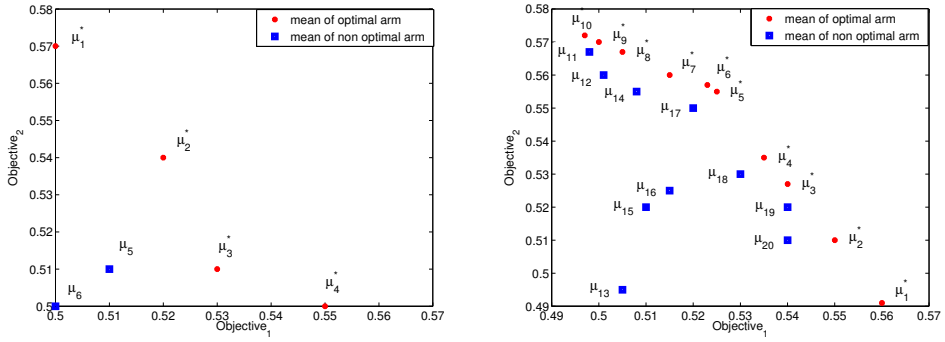


Fig. 2. Non-convex and convex mean vector set. Left figure shows a non-convex set with 2-objective, 6-armed. Right figure shows a convex set with 2-objective, 20-armed.

the variance. To get rid of tuning the parameter ϵ_{decay}, we generate uniformly at random the parameter $\epsilon_{decay} \in (0, 1)$. Shannon entropy measures the unfairness regret, Section 4. For example, for 2-objective, 6-armed with Pareto front $A^* = \{a_1^*, a_2^*, a_3^*, a_4^*\}$, where a_i^* is an optimal arm, Experiment 1. If the number of selecting each arm vector $\boldsymbol{N}$ by an algorithm is $\boldsymbol{N} = [30, 20, 20, 15, 10, 5]^T$ and the optimal number $\boldsymbol{N}^*$ of selecting each arm is $\boldsymbol{N}^* = [25, 25, 25, 25, 0, 0]^T$ at time step $t = 100$ without *initial* steps, then Shannon entropy is 0.0143.

Non-Convex Mean Vector Set;

Experiment 1. We use the same example in [2], since it is simple to understand and the Pareto mean set contains values close to each others. The number of arms $|A|$ is 6, and the number of objectives $|D|$ is 2. The true mean vector set is $(\boldsymbol{\mu}_1 = [0.55, 0.5]^T, \boldsymbol{\mu}_2 = [0.53, 0.51]^T, \boldsymbol{\mu}_3 = [0.52, 0.54]^T, \boldsymbol{\mu}_4 = [0.5, 0.57]^T, \boldsymbol{\mu}_5 = [0.51, 0.51]^T, \boldsymbol{\mu}_6 = [0.5, 0.5]^T)$, the standard deviation for arms in each objective is set to 0.1. Note that the Pareto front is $A^* = (a_1^*, a_2^*, a_3^*, a_4^*)$ where a_i^* refers to the optimal arm i^*. The suboptimal a_5 is not dominated by the two optimal arms a_1^* and a_4^*, but a_2^* and a_3^* dominates a_5 while a_6 is dominated by all the other mean vectors. Fig. 2 shows a set of 2-objective true mean with a non-convex set.

Experiment 2. We add extra 3 objectives and 14 arms in Experiment 1, resulting in 5-objective, 20-armed, we add 3 optimal arms and 11 dominated arms by all the arms in Pareto front A^*. Pareto front contains 7 optimal arms. Fig. 3 gives the average cumulative Pareto and unfairness regret performances. The y-axis is either the average of the cumulative Pareto or unfairness regret performance. The x-axis is the time steps. Fig. 3 shows the performance of algorithms. The annealing-Pareto is the best algorithm and Pareto-UCB1 is the worst one. Pareto-KG has an intermediate performance.

Convex Mean Vector Set

Experiment 3. With number of objectives D equals 2, number of arms $|A|$ equals 20 and convex Pareto mean set, $(\boldsymbol{\mu}_1 = [.56, .491]^T, \boldsymbol{\mu}_2 = [.55, .51]^T, \boldsymbol{\mu}_3 = [.54, .527]^T, \boldsymbol{\mu}_4 = [.535, .535]^T, \boldsymbol{\mu}_5 = [.525, .555]^T, \boldsymbol{\mu}_6 = [.523, .557]^T, \boldsymbol{\mu}_7 = [.515, .56]^T, \boldsymbol{\mu}_8 = [.505, .567]^T, \boldsymbol{\mu}_9 = [.5, .57]^T, \boldsymbol{\mu}_{10} = [.497, .572]^T, \boldsymbol{\mu}_{11} = [.498, .567]^T, \boldsymbol{\mu}_{12} = [.501, .56]^T, \boldsymbol{\mu}_{13} = [.505, .495]^T, \boldsymbol{\mu}_{14} = [.508, .555]^T, \boldsymbol{\mu}_{15} = [.51, .52]^T, \boldsymbol{\mu}_{16} = [.515, .525]^T, \boldsymbol{\mu}_{17} = [.52, .55]^T, \boldsymbol{\mu}_{18} = [.53, .53]^T, \boldsymbol{\mu}_{19} = [.54, .52]^T, \boldsymbol{\mu}_{20} = [.54, .51]^T)$,

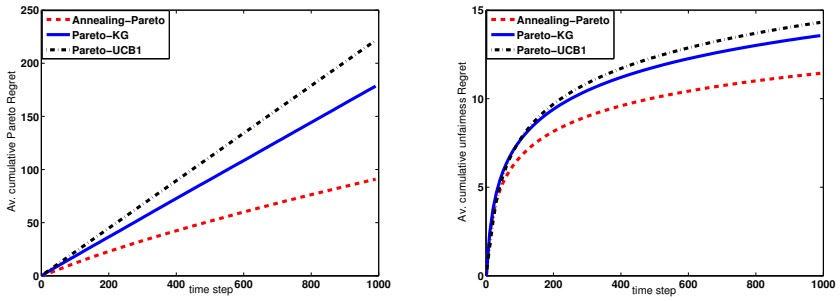


Fig. 3. Performance comparison on 5-objective, 20-armed with non-convex mean vector set. Left sub-figure shows the average cumulative Pareto regret performance. Right sub-figure shows the average cumulative unfairness regret performance.

the standard deviation for arms in each objective is set to 0.1. The Pareto front A^* contains 10 optimal arms, $A^* = (a_1^*, a_2^*, a_3^*, a_4^*, a_5^*, a_6^*, a_7^*, a_8^*, a_9^*, a_{10}^*)$. Fig. 2 shows a set of 2-objective convex true mean vector set.

Experiment 4. We add extra 3 objectives and 10 arms in Experiment 3, resulting in 5-objective, 20-armed, we add dominated arms by all the arms in A^*. Pareto front A^* still contains 10 optimal arms. Fig. 4 gives the average cumulative Pareto and unfairness regrets and shows the annealing-Pareto performance is the best algorithm, and the Pareto-UCB1 performance is the worst one according to the Pareto regret performance, while according to the unfairness regret performance Pareto-KG is the worst algorithm.

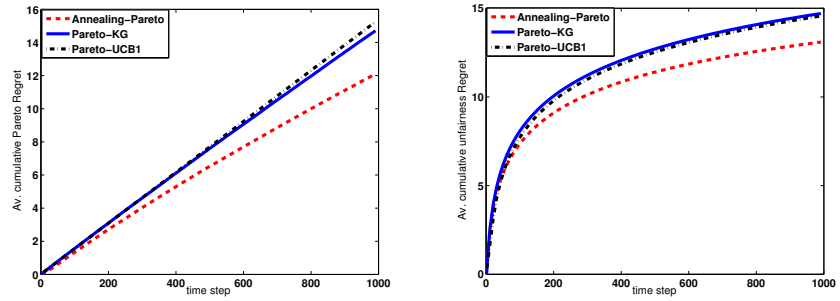


Fig. 4. Performance comparison on 5-objective, 20-armed with convex mean vector set. Left sub-figure shows the average cumulative Pareto regret performance. Right sub-figure shows the average cumulative unfairness regret performance.

From the above experiments, we see that the annealing-Pareto algorithm is the best one according to both the unfairness and Pareto regrets. The intuition is that the annealing-Pareto does not have an exploration term that decreases fast to 0 after time steps to control the trade-off between exploration and exploitation. Instead, the annealing-Pareto has a decay parameter that decreases slowly to 0, this means that the annealing-Pareto explores widely the available arms.

For convex mean vector set, Pareto-KG outperforms Pareto-UCB1 according to the Pareto and unfairness regrets.While, for non-convex mean vector set, Pareto-KG outperforms Pareto-UCB1 according to the Pareto regret and Pareto-UCB1 outperforms Pareto-KG according to the unfairness regret. The intuition is the exploration term. The exploration term for UCB1 depends on the time step t and the number of times N_i arm i is pulled and it will be high if the arm i is less selected. Thus, UCB1 plays fairly the optimal arms because it selects the optimal arms that have either larger estimated mean or larger exploration term. In contrast, the exploration term for KG policy depends on the estimated mean of all other arms and on the estimated variance of arm i. The exploration term is large if the variance of arm i is low, or if the estimated mean of arm i exceeds in the future. Thus, KG selects more efficiently the optimal arms.

7 Conclusion

We introduced the normal MOMAB, Pareto dominance relation, the performance measure in the MOMAB, Pareto-KG and Pareto-UCB1. We proposed the annealing-Pareto algorithm. We proposed using the entropy measure as a performance measure in the MOMAB. We studied empirically the trade-off between exploration and exploitation (or the trade-off for short) in the normal MOMAB. Pareto-KG and Pareto-UCB1 trade-off by using KG and UCB1 policy, respectively. While, the annealing-Pareto trades-off by using a decay parameter. Finally, we compared Pareto-KG, Pareto-UCB1, and the annealing-Pareto and concluded that: the annealing-Pareto is the best algorithm according to both the Pareto and the unfairness regret performance measures.

References

1. Yahyaa, S.Q., Drugan, M.M., Manderick, M.: The Scalarized Multi-Objective Multi-Armed Bandit Problem: An Empirical Study of its Exploration vs. Exploration Tradeoff. In: International Joint Conference on Neural Networks. (2014)
2. Drugan, M.M., Nowe, A.: Designing Multi-Objective Multi-Armed Bandits Algorithms: A study. In: International Joint Conference on Neural Networks. (2013)
3. Sethna, J.: Statistical Mechanics: Entropy, Order Parameters and Complexity. Oxford University Press, (2006)
4. Zitzler, E. and et al.: Performance Assessment of Multiobjective Optimizers: An Analysis and Review. J. IEEE Transactions on Evolutionary Computation 7, 117–132 (2002)
5. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-Time Analysis of the Multiarmed Bandit Problem. J. Machine Learning 47(2-3), 235–256 (2002)
6. Yahyaa, S.Q., Drugan, M.M., Manderick, B.: Knowledge Gradient for Multi-Objective Multi-Armed Bandit Algorithms. In: International Conference on Agents and Artificial Intelligence (ICAART). (2014)
7. Ryzhov, I.O., Powell, W.B., Frazier, P.I.: The Knowledge Gradient Policy for a General Class of Online Learning Problems. In: Operation Research, (2011)
8. Powell, W.B.: Approximate Dynamic Programming: Solving the Curses of Dimensionality. John Willey and Sons, New York, USA, (2007)

Managing Ventilation Systems for Improving User Comfort in Smart Buildings using Reinforcement Learning Agents

Jiawei Zhu[1], Fabrice Lauri[1], Abderrafiaa Koukam[1], and Vincent Hilaire[1]

IRTES-SET, UTBM, 90010 Belfort cedex, France
(jiawei.zhu, fabrice.lauri, abder.koukam, vincent.hilaire)@utbm.fr

Abstract. With the fast development of information technology and increasingly prominent environmental problems, building comfort and energy management become the major tasks for an intelligent residential building system. This paper identifies the system requirements of Smart Buildings, analyzes the problems that need to be solved and how Reinforcement Learning is suitable for dealing with them. It also proposes to represent parts of Smart Buildings as Cyber-Physical Systems. Although the global goal is to model and manage a complex and whole system of a Smart Building, since the work is in progress, in this paper we mainly focus on how Reinforcement Learning technique is good at controlling subsystems, specifically the Ventilation System. The experimental results show the advantages of our system compared with the widely used baselines: On/Off control and PI control approaches.

Keywords: energy, smart buildings, reinforcement learning, multi-agent system, cyber-physical system.

1 Introduction

According to United Nations Environment Programme [1], buildings use about 40% of global energy, 25% of global water, 40% of global resources, and they emit approximately 1/3 of Green House Gas (GHG) emissions. With the development of human society, environmental issues have drawn more and more attention. In this background, buildings can offer a great potential for achieving significant GHG emission reductions in different countries. Furthermore, energy consumption in buildings can be reduced by using advanced technologies and management. On the other hand, people spend greater part of their time in buildings. As the quality of life in home is increasingly considered as of paramount importance, many people constantly seek to improve comfort in their living spaces. Meanwhile, the popularization of the concept of home office makes the productivity in smart buildings economically significant. How to manage buildings in a proper way to improve energy efficiency and comfort level while reducing pollution at the same time is therefore a subject of uttermost importance.

Corresponding to the increasing demands for environment, comfort, energy, and productivity, advanced methods are applied for improving comfort conditions in smart buildings thanks to the dramatically rapid development of information technologies. Widespread utilization of computing devices, powerful but low cost sensors and actuators, and ubiquitous networks make the intelligent control more easily come true. Actually the implementation of smart buildings involves controls of different subsystems and devices. Hence itself is a system of systems.

Based on this context, Cyber-Physical System (CPS) can be used to model this complex system, which is integrations of computation, networking and physical processes, in which embedded computers and networks monitor and control the physical processes with feedback loops where physical processes affect computations and vice versa [2]. CPSs integrate the dynamics of the physical processes with those of the software and networking, providing abstractions and modelling, design, and analysis techniques for the integrated whole. Modelling and controlling smart buildings as CPSs can bring many advantages: different subsystems such as heating, ventilation and air-conditioning (HVAC) can communicate with other electrical devices to form an intelligent whole; more information can be integrated and shared, for example, real-time and forecasting local weather data from observatories can be used through networks to assist HVAC system to make better decisions or even to help power distributors to balance loads; the system can be more robust and the cost can be reduced by separating sensors and actuators from traditional electrical devices.

In this work, we try to reformulate smart buildings as CPSs and capitalise on Reinforcement Learning (RL) and Multi-Agent techniques to control the whole system. Due to the work being in progress, in this paper we mainly focus on modeling and controlling subsystems, specifically the ventilation system. Our contributions is threefold: firstly we identify the system requirements for smart buildings; then inspired from [3] we propose the method to model ventilation system as CPS; finally RL is proposed to control the ventilation system, its performance is analyzed and compared with PI control and On/Off control. The rest of this paper is organized as follows: Section 2 presents the related work. Section 3 analyzes system requirements for smart buildings. Section 4 focuses on RL and its feasibility. Section 5 investigates ventilation system and models it as CPS. Experimental results are presented and analyzed in Section 6. Finally we conclude in Section 7.

2 Related Work

Research is increasing in the emerging field of smart buildings. Kleissl *et al.* [4] regard smart buildings as CPS, and by examining different buildings and their energy use in detail they point out opportunities available to improve energy efficiency operation through various strategies from lighting to computing. However, the requirements for developing smart buildings and the architecture of the system have not been analysed and proposed. In order to improve com-

fort in buildings, authors of [5] propose an adaptive smart home system named CASAS, which utilizes machine learning techniques to discover patterns in occupant's daily activities and to generate automation policies that mimic these patterns. Although the user's explicit or implicit wishes can be adapted, the energy consumption of the building has not been taken into account. Actually energy efficiency is one of the key factors that need to be considered when designing smart buildings, since more comfort usually comes at the expense of higher energy consumption. Therefore these two conflicting points should be carefully balanced. In [3] and [6], thermal comfort and indoor air quality in buildings are improved separately by intelligent control methods while less energy is used compared with conventional controllers. However, a smart building is a system of systems, and only individual subsystems being well controlled is not enough. Hence, in our work we undertake the analysis and design from a global view, while implementing the system by a bottom-up approach.

Both to make subsystems have the ability to take intelligent decisions, and the global system learn good strategies to schedule and coordinate these subsystems, RL brings advantageous properties. Up to now, RL has been successfully used on a wide range of problems. Peters *et al.* [7] propose an intelligent decentralized control mechanism, which is able to operate in different Smart Electricity Markets, by using autonomous broker agents. These agents can accommodate arbitrary economic signals and learn efficiently over the large state spaces resulting from the signals with function approximation. After learning, they are capable of deriving long-term, profit-maximizing policies. Li *et al.* [8] present an improved MAXQ [9] method to minimize electricity costs on the premise of satisfying the power balance and generation limit of units in a microgrid and the proposed multi-agent architecture is beneficial to handle the problem of "curse of dimensionality" and speed up learning in the unknown large-scale world. Other works focusing on robotic and traffic light control can be found in [10,11] and [12,13] respectively.

3 System Requirements for Smart Buildings

Actually a Smart building is a system of systems. It requires to think about different subsystems and devices as an integrated whole that has a global objective. Different functional parts of this whole can be modelled as agents so that the individual objectives and the global objective can be reached by cooperating, coordinating, and negotiating among these agents. Therefore, the smart building can be regarded as a multi-agent problem. On the other hand, improving the comfort of a single building is important whereas reducing energy consumption so as to reduce electric bill, balancing power distribution, and shifting peak load are also vital. So the desired strategy is to reasonably balance comfort and energy consumption not only in one building but also among different buildings of a district. In order to model this complex system, we first need to analyse the requirements for smart buildings.

3.1 Multi-Authority and Multi-Level

A smart building comprises various agents, such as air-conditioning agent, ventilation agent, water-heating agent and so on. This forms multi-authority of lower level. If there does not exist communication between agents, each of them can be considered as selfish, that is each of them try to maximize its individual goal regardless of the states of other agents and the global objective. From a higher viewpoint, each smart building can be regarded as an independent agent and some of these agents constitute a smart district, in which smart buildings can improve their comfort, balance total grid load and reduce energy consumption through communication, cooperation and coordination. This is multi-level requirement for smart buildings.

3.2 Multi-Objective

Authorities of different levels want to reach their own objectives and global objectives of higher level. For instance, thermal comfort, indoor air quality, and visual comfort are three basic factors which determine the comfort conditions in buildings [14], and relative devices treat improving these comforts as their individual objective, while they also need to consider global objective: reducing energy consumption and peak load shifting. Individual objective and global objective are often conflicting: improving comfort often means consuming more energy.

3.3 Heterogeneity

Heterogeneity in smart buildings mainly comes from three aspects. First, the devices in buildings are diverse with different control strategies. How to integrate them together and have a proper management is important. Moreover, multi-level structure, that has been presented in Section 3.1, brings difficulties in designing the system. In addition, different occupants have different user preferences, including comfort definition, device type, and their physical activities in buildings. Electrical devices in buildings are heterogeneous. In general, they can be divided into two categories: power consuming devices and power producing devices. However, in [15] these two categories are unified by a new word called prosumer which means either producing or consuming. By convention, a positive prosumption represents a production and a negative one a consumption. But in order to analyse device properties we still use former notions.

Power Consuming Devices In this category, devices consume power when they are working and they are divided as negotiable and non-negotiable. Negotiable devices are these who can reduce working power, called power-negotiable (*e.g.* intelligent air-conditioning, intelligent ventilation system), so as to reduce comfort within a range that occupants could accept; who can postpone scheming start-time, named time-negotiable (*e.g.* washing machine), in order to shift

peak load; who can both reduce working power and postpone scheming start-time, called power-time-negotiable (*e.g.* water heating system, storage system), to provide a flexible service. Non-negotiable devices are inflexible, that means when people turn them on, they always consume power as required (*e.g.* daylight lamp, TV, computer).

Power Producing Devices In smart buildings, there often exist power producing devices to make full use of green energy and help decrease the load on main grids. Some of these devices can provide constant power like fuel cells, micro turbines, and storage systems, while the others can merely offer variable outputs, which strongly depend on weather conditions, such as photovoltaic panels and wind turbines. At this time, storage systems are required for these devices to play a role as buffers, which can achieve constant outputs to protect the micro-grid.

3.4 Scalability

The desired system should be scalable. In a single building, devices often plug in and out, and in a district, new buildings may participate. This requires the architecture we design have the ability of scalability and the decision-making algorithms need to be decentralized.

3.5 Incremental Change

Although smart buildings can bring numerous benefits, traditional buildings with traditional devices already exit. Hence any changes introduced in the future should be reasonably gradual so as not to disturb and damage the working system and its service. This requires the designing system can tolerate and integrate the exiting traditional devices.

4 Reinforcement Learning

Compared with traditional control, CPSs enable consider more inputs to better realize the dynamic of the physical world so as to support decision making. For example, nowadays most air-conditionings use On/Off and PI control. On/Off control regulates temperature by using a compressor that is periodically either working at maximum capacity or switched off entirely, whereas PI control has a variable-frequency drive that incorporates an adjustable electrical inverter to control the speed of the motor and thus the compressor and cooling output. The inputs for On/Off and PI control are only indoor temperature. However, for CPSs they can capitalize on more inputs like occupant number, since bodies are also heat source that can increase indoor temperature. In order to benefit from CPSs, it seems that traditional control methods, which are often straightforward, are not enough.

Hence, we advocate the use of RL to control smart buildings, which offers a suitable set of techniques to address these challenges. The classical reinforcement learning framework is based on Markov Decision Processes (MDPs). An MDP can be depicted by a tuple (X, U, f, ρ, γ). X is the set of states it can perceive, U is the set of possible actions it can perform in these states, f is the state transition function, ρ is the reward function that evaluates the immediate effect of an action, and γ is the discount factor. The goal of RL is to find an optimal policy, $h : X \rightarrow U$, that maximizes the return from any initial state x_0: $R^h(x_0) = \sum_{k=0}^{\infty} \gamma^k \rho(x_k, h(x_k))$, where $\gamma \in [0, 1)$ and k is discrete time step. The discount factor can be interpreted intuitively as a measure of how "far-sighted" the controller is on its rewards, or as a way of taking into account increasing uncertainty about future rewards [16]. In order to characterize policies, state-action value function (Q-function) is used, $Q^h : X \times U \rightarrow \mathbb{R}$. After finding an optimal Q-function $Q^*(x, u)$, optimal policy $h^*(x)$ can be obtained greedily by $h^*(x) \in \arg\max_u Q^*(x, u)$. In this work, the model-free online algorithm Q-learning [17] is used to update the Q-function. The choice of this algorithm is motivated by the fact that no explicit model of the dynamics of a smart building is available, due to the great amount of involved devices. Moreover, some inputs like the number of occupants presented in the room are random. Hence, considering f as a stochastic function is more realistic. Q-learning can work as a sample-based algorithm to deal with stochastic approximation procedure. The Q-function is updated online at every new sample of the form $(x_k, u_k, x_{k+1}, r_{k+1})$, using the following equation: $Q_{k+1}(x_k, u_k) \leftarrow Q_k(x_k, u_k) + \alpha_k[r_{k+1} + \gamma \max_{u'} Q_k(x_{k+1}, u') - Q_k(x_k, u_k)]$. In subsystems, RL can be utilized as intelligent controller to control devices such as heating and ventilating systems. In global system, RL can optimize the coordination of subsystems.

5 Ventilation Controlling Subsystem

In this section, we mainly focus on the ventilation subsystem. A thermal subsystem has been investigated in our previous work [18].

The ventilation controlling system is used to improve indoor air quality. In most cases, people can obtain a good indoor air quality by simply opening windows. However, in some situations, we need mechanical ventilating devices to exchange indoor air, for example, when there are many visitors in the room, when outdoor air speed is close to zero, when there is no window in the room, and for people who live in modern skyscrapers in which opening windows will raise the possibility of hidden danger and hence is often forbidden.

The indoor air quality is mainly decided by CO_2 concentration. People generate CO_2 and consume oxygen, at a rate that depends primarily on their body size and their level of physical activity[19]. The rate of oxygen consumption in L/s of a person is given as

$$V_{O_2} = \frac{0.00276 A_D M}{(0.23RQ + 0.77)} \tag{1}$$

where RQ is the respiratory quotient (the relative volumetric rates of carbon dioxide produced to oxygen consumed). M is the level of physical activity or the metabolic rate per unit of surface area in met (1 met = 58.2 W/m^2). A_D is the DuBois surface area in m^2, which can be estimated by the following equation

$$A_D = 0.203 H^{0.725} W^{0.425} \tag{2}$$

where H is the body height in meter and W is the body mass in kg. For an average size adult, A_D is about $1.8m^2$.

The value of RQ depends on diet, the level of physical activity and the physical condition of the person. It is equal to 0.83 for an average size adult engaged in light or sedentary activities (about 1 met), and increases to a value of about 1 for heavy physical activity (about 5 met). The carbon dioxide generation rate in L/s of an individual is

$$V_{CO_2} = V_{O_2} \times RQ \tag{3}$$

Steady state CO_2 concentration can be determined for a given ventilation rate based on a single zone mass balance analysis. Assuming that in a room there are N adults and the room is equipped with an electric fan. The mass balance of CO_2 in the room can be expressed as follows:

$$V \frac{dC}{dt} \times 10^{-6} = G \times 10^{-3} + Q \times (C_{out} - C) \times 10^{-6} \tag{4}$$

where V is building volume in m^3, C is indoor CO_2 concentration in $ppm(v)$, C_{out} is outdoor CO_2 concentration in $ppm(v)$, t is time in second, G is indoor CO_2 generation rate in L/s, and Q is ventilation rate in m^3/s. Generally an acceptable value of indoor CO_2 concentration varies from 600 $ppm(v)$ to 1000 $ppm(v)$, and 800 $ppm(v)$ is set as a reasonable setpoint for a good indoor air quality.

Figure 1 depicts the architecture of the CPS for ventilation control. In this figure, there are three parts: Physical World, Network, and Cyber World. Physical World contains Sensor Domain and Actuator Domain. Different indoor and outdoor parameters, such as air speed, CO_2 concentration, occupant number, metabolic rate, etc., can be observed by different sensors and these data are transmitted to Cyber World though the network. In Cyber World, we use RL because it can deal with the random appearance of occupants. With this algorithm, the captured data are used to decide the current state and the reward of last time. RL can make good decisions automatically by trial and error without the need of a specific model of the problem. Based on this technique multiple devices in Physical World are controlled by the actuating signals from Cyber World.

6 Experiments

The goal of this work is to control the mechanical ventilation system to keep the CO_2 concentration at the setpoint while reduce energy consumption. Specifically, according to the present information of occupants' number and indoor

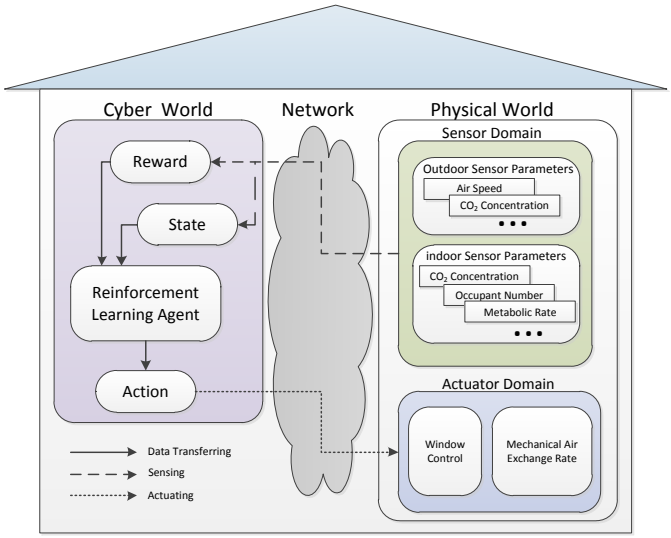
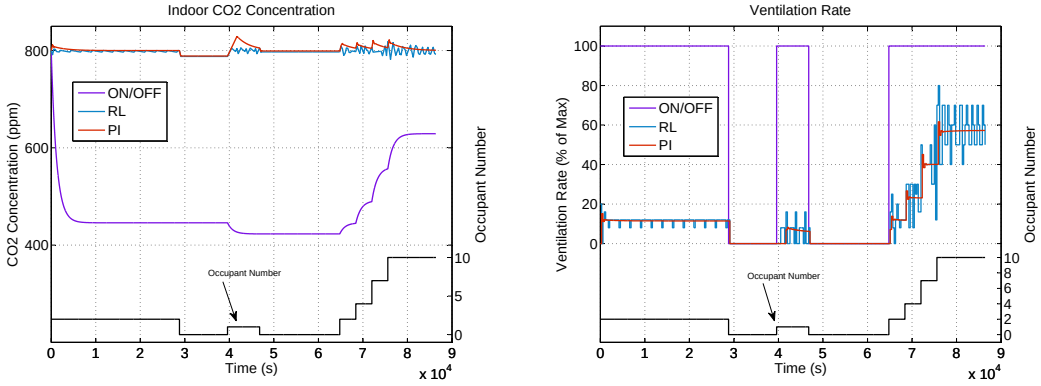


Fig. 1. Cyber-Physical System for ventilation control

CO_2 concentration, the system should adjust the electric fan's speed to control the ventilation rate. We assume the maximum number of occupants is 10 with average size and doing light or sedentary activities in a house of $100m^2 \times 3m$. The CO_2 concentration in $[750, 1200]$ is discretized into 450 states, plus 2 over boundary states. Therefore the total number of state is $452 \times 11 = 4972$. The mechanical ventilation system used has a maximum ventilation rate of $0.25m^3/s$ with power of 40 W and can take 13 actions: {0,4,8,12,16,20,25,30,40,50,60,70,80,100}% of the maximum ventilation rate. The reward function is $r_{k+1} = e^{-\frac{(C_k-800)^2}{20000}} \times 8 - 8$, where C_k is the indoor CO_2 concentration at time step k. Due to the slow variation property of CO_2 concentration, the time step is set to 300 seconds.

Figure 2(a) compares the CO_2 concentration variations within one day by three different control methods. The black line is the occupant number change during this period of time. The result indicates that the CO_2 concentration can be unnecessarily reduced far below 800 *ppm* by On/Off control, which simply turns on the fan with maximum power if any occupants are detected in the room while turns off if not. Although it can provide continuously fresh air flow, it consumes much more energy than the others. PI control (proportional gain: 0.003, integral gain: 0.000001) can keep the CO_2 concentration at the setpoint smoothly, except for every change of number of occupant's presence, which causes the overshoot of the concentration. RL method can offer the best comfort, even though the CO_2 concentration has small vibration, that is caused by the discrete definition of actions and occurs often in RL applications. For residential buildings, this slight fluctuation will not affect inhabitants' comfort. The comparison of ventilation rates is presented in Figure 2(b). It reflects that when there are more occupants in the room, the quicker dynamic of physical environment makes



(a) Indoor CO_2 Concentration (b) Ventilation Rate

Fig. 2. Experimental Results

it more challenging for RL to control. The total energy spent is 0.1379 kWh/day by PI control, 0.1403 kWh/day by RL, and 0.6401 kWh/day by On/Off control. Compared with On/Off control, RL can save 78.08% energy, and compared with PI control, although RL use 1.74% more energy, it is not only able to maintain good indoor air quality but also more suitable and feasible for implementing CPSs in smart buildings.

7 Conclusion

In this paper, we identified the system requirements for smart buildings, including multi-authority and multi-level, multi-objective, heterogeneity, scalability, and incremental change. Then we presented the framework of Reinforcement Learning and explained why RL is suitable to resolve most of the smart building challenges. After that, a subsystem, specifically a ventilation system, was investigated and modeled by a CPS approach. The experimental results revealed that Q-Learning, a model-free online RL technique, is more adaptable and feasible than conventional PI and On/Off approaches for managing a ventilation system in a smart building to improve comfort level while reduce energy consumption.

In the future, function approximation will be utilized, since there are more information (input variables) available for CPSs and often these input variables are continuous, so it will not be applicable to discretize them anymore. In addition, various RL techniques will be compared and analyzed to find their applicability for smart building management. After that, different subsystems in smart buildings will be integrated together based on Multi-agent System approach.

References

1. UNEP. http://www.unep.org/sbci/AboutSBCI/Background.asp

2. Lee, E.A.: Cyber physical systems: Design challenges. In: 11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing (ISORC), IEEE (2008) 363–369
3. Cheng, Z., Shein, W.W., Tan, Y., Lim, A.: Energy efficient thermal comfort control for cyber-physical home system. In: IEEE International Conference on Smart Grid Communications (SmartGridComm). (Oct 2013) 797–802
4. Kleissl, J., Agarwal, Y.: Cyber-physical energy systems: Focus on smart buildings. In: 47th ACM/IEEE Design Automation Conference (DAC). (June 2010) 749–754
5. Rashidi, P., Cook, D.: Keeping the resident in the loop: Adapting the smart home to the user. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans **39**(5) (Sept 2009) 949–959
6. Wang, Z., Wang, L.: Intelligent control of ventilation system for energy-efficient buildings with co2 predictive model. IEEE Transactions on Smart Grid **4**(2) (June 2013) 686–693
7. Peters, M., Ketter, W., Saar-Tsechansky, M., Collins, J.: Autonomous data-driven decision-making in smart electricity markets. In: Proceedings of the 2012 European Conference on Machine Learning and Knowledge Discovery in Databases. (2012) 132–147
8. Li, F.D., Wu, M., He, Y., Chen, X.: Optimal control in microgrid using multi-agent reinforcement learning. ISA Transactions **51**(6) (2012) 743 – 751
9. Dietterich, T.G.: Hierarchical reinforcement learning with the maxq value function decomposition. Journal of Artificial Intelligence Research **13**(1) (nov 2000) 227–303
10. Smart, W., Kaelbling, L.: Effective reinforcement learning for mobile robots. In: IEEE International Conference on Robotics and Automation. Volume 4. (2002) 3404–3410
11. Mataric, M.: Reinforcement learning in the multi-robot domain. Autonomous Robots **4**(1) (1997) 73–83
12. Abdulhai, B., Pringle, R., Karakoulas, G.J.: Reinforcement learning for true adaptive traffic signal control. Journal of Transportation Engineering **129**(3) (2003) 278–285
13. Wiering, M.: Multi-agent reinforcement learning for traffic light control. In: ICML. (2000) 1151–1158
14. Wang, Z., Yang, R., Wang, L.: Multi-agent intelligent controller design for smart and sustainable buildings. In: 4th Annual IEEE Systems Conference. (April 2010) 277–282
15. Frey, S., Diaconescu, A., Menga, D., Demeure, I.: A holonic control architecture for a heterogeneous multi-objective smart micro-grid. In: IEEE 7th International Conference on Self-Adaptive and Self-Organizing Systems (SASO). (Sept 2013) 21–30
16. Busoniu, L., Babuska, R., De Schutter, B., Ernst, D.: Reinforcement learning and dynamic programming using function approximators. CRC Press (2010)
17. Watkins, C.J., Dayan, P.: Q-learning. Machine learning **8**(3-4) (1992) 279–292
18. Zhu, J., Lauri, F., Koukam, A., Hilaire, V., Simoes, M.G.: Improving thermal comfort in residential buildings using artificial immune system. In: IEEE 10th International Conference on Ubiquitous Intelligence and Computing(UIC). (Dec 2013) 194–200
19. Emmerich, S.J., Persily, A.K.: State-of-the-art review of co2 demand controlled ventilation technology and application. US Department of Commerce, Technology Administration, National Institute of Standards and Technology (2001)

A Framework for Pattern Classifier Selection and Fusion

Fabio A. Faria*, Anderson Rocha, and Ricardo da S. Torres

Institute of Computing, University of Campinas (Unicamp)
Cidade Universitária "Zeferino Vaz",
Campinas, SP, Brazil – CEP 13083-852
{ffaria,anderson.rocha,rtorres}@ic.unicamp.br
http://www.ic.unicamp.br/~ffaria

Abstract. In this work, we propose a framework for classifier selection and fusion. Our method seeks to combine image characterization and learning methods by means of a meta-learning approach responsible for assessing which methods contribute more towards the solution of a given problem. The framework uses three different strategies of classifier selection that pinpoint the less correlated, yet effective, classifiers through a series of diversity measure analysis. The experiments show that the proposed approaches yield comparable results to well-known algorithms from the literature on many different applications but using less learning and description methods as well as not incurring in the curse of dimensionality and normalization problems common to some fusion techniques. Furthermore, our approach yields effective classification results using very reduced training sets.

Keywords: meta-learning; diversity measure; rank aggregation; kendall correlation

1 Motivation

The huge amount of visual data created through the popularization of mobile devices (e.g., cell phone, camera, and tablet), makes us face many new challenges unthinkable two decades ago. Image and video classification tasks have been inserted in different and complex applications and the use of machine learning-based solutions has become the most popular approach to several applications. However, there is no single solution (learning or extraction technique) that solves all the problems. Depending on the extraction and learning methods used might create different classifiers that provide complementary information. One common strategy that has been used to take advantage of these complementary information and improve classification results is the Multiple Classifier System (MCS). In MCS, the diversity of classifiers is an essential factor to reach better effectiveness

* The author thank São Paulo Research Foundation - FAPESP (grant 2010/14910-0) 2010-2014 and CAPES (grant 1260-12-0) for the financial support.

results [17]. Diversity measures assess the degree of agreement/disagreement between classifiers and might identify potential classifiers for fusion. In this sense, Kuncheva and Whitaker [18] studied different diversity measures as well as discussed their impacts on the final accuracy of ensemble systems. Different works have been using diversity measures to select appropriate high-performance classifiers, but the challenge of finding the optimal number of classifiers for a target task has not been properly addressed yet. In general, the proposed solutions rely on the a priori use of ad hoc strategies for selecting classifiers, followed by the evaluation of their effectiveness results during training. Searching by the optimal number of classifiers, however, makes the selection process more expensive.

Currently, some of the most important challenges in MCS involve: choosing the best diversity measure to be used; combining different available measures for classifier selection in an ensemble system; and finding out whether or not the existing measures describe the "real" diversity within the ensemble systems [4]. Typically, works in the literature have adopted a single diversity measure or combined different measures using simple strategies (e.g., based on average of the classification scores [5]). However, the aforementioned methods might not take full advantage of the different opinions provided by all of the available diversity measures. Moreover, another problem in MCS approaches is how to combine different and non-correlated extraction and learning methods automatically.

In the literature, many works have been proposed to try sorting out problems cited previously as for example, the well-known AdaBoost [13] and Bagging [2] approaches. AdaBoost and Bagging ensemble approaches have been used in several works in the literature due to their good results achieved in diverse applications. However, previous work has also shown their limitations in terms of efficiency, normalization, overfitting, and feature dimensionality problems. In [21], for example, training time has been a concern when more features were used to train an AdaBoost algorithm for face localization. The same problem has been reported in [19], which trained an AdaBoost algorithm for tracking indoor soccer players using videos. In [16], the authors discuss another problem: the sensitivity of the classical AdaBoost algorithm to noisy datasets. They have proposed different solutions to reduce the overfitting effect caused in those cases. In [20], the authors discuss the problems of feature normalization in the context of combining classifiers. More detail about tracking down fusion and classification problems can be found in [6].

The combination of multiple feature vectors defined by different image descriptors in AdaBoost and Bagging approaches is usually based on their concatenation (feature binding). Usually, when performing feature binding of different nature/domain, normalization techniques should be applied to standardize all feature values in the same range, which is a very challenging task [20]. Another common problem faced when features are concatenated refers to the "curse of dimensionality" [22]. The curse of dimensionality problem is related to the fact that the dimension of the feature space increases in such a way that the available training instances become indistinguishable and it is not enough for allowing the definition of a good decision hyperplane [1].

2 Objective and Contributions

In this work, we seek an alternative to AdaBoost and Bagging ensembles, which might suffer curse of dimensionality and normalized problems. Our objective is to propose a stacking framework, able to perform automatic fusion of different visual properties (color, texture, and shape) and learning methods in existence in the literature for different multimedia recognition tasks.

The framework assesses several descriptors and learning methods performing fusion in a final stage (late fusion) using a low-dimension feature vector and simple (fast) classifiers. Another difference of the proposed method, when compared to AdaBoost and Bagging techniques, is that the proposed framework seeks greater diversity between the simple classifiers being able to choose only the ones that effectively contribute to the solution of the classification problem of interest.

Diversity may be obtained in different ways such as using: (a) different learning methods and the same training set; (b) the same learning method and different training samples; (c) different methods using different types of classifier outcomes during the combination; and (d) predictions as new attributes to train some learning method (meta-learning). In this work, we use two out of four ways (*a* and *d*). We also use different visual properties (color, texture, and shape) to each of the learning methods chosen to be simple classifiers. We follow the concept that two instances of the same class have similar classification outputs for the same set of classifiers [14].

In this regard, in this work, we investigate the combination of several learning methods and image descriptors aiming at creating more effective classifiers. We propose a framework for automatically combining the most discriminative classifiers using the support vector machine (SVM) technique, as well as exploring the use of diversity measures to select the less-correlated, yet effective, classifiers in three different selection strategies. We have performed experiments that demonstrate that the proposed framework for classifier fusion yields comparable results to the traditional fusion approaches but using less learning and description methods as well as not incurring in the curse of dimensionality problems, which are common to some fusion techniques. Another major advantage of the proposed method is that it yields good classification results using small training examples being more robust to the small sample size problem common in many classification techniques [1].

Our research hypothesis is that: *Appropriate classifier selection approaches can take advantage of classifier diversity to improve the accuracy performance of multiple classifier systems.*

The contributions and publications directly related to this thesis are: a framework for classifier fusion through a meta-learning approach using Support Vector Machines techniques [11]; a new classifier selection approach based on diversity measures consensus [9, 10]; a new classifier selection approach based on *Kendall* correlation analysis [12]; and a new classifier selection approach based on rank aggregation techniques [8]; a multimodal framework for automatic identification of fruit flies (Diptera: Tephritidae) [7].

3 The Classifier Fusion Framework

The objective of the fusion framework [10] is to exploit the degree of agreement/disagreement among classifiers, concept known as diversity, to select the most suitable ones to be used in a combination scheme.

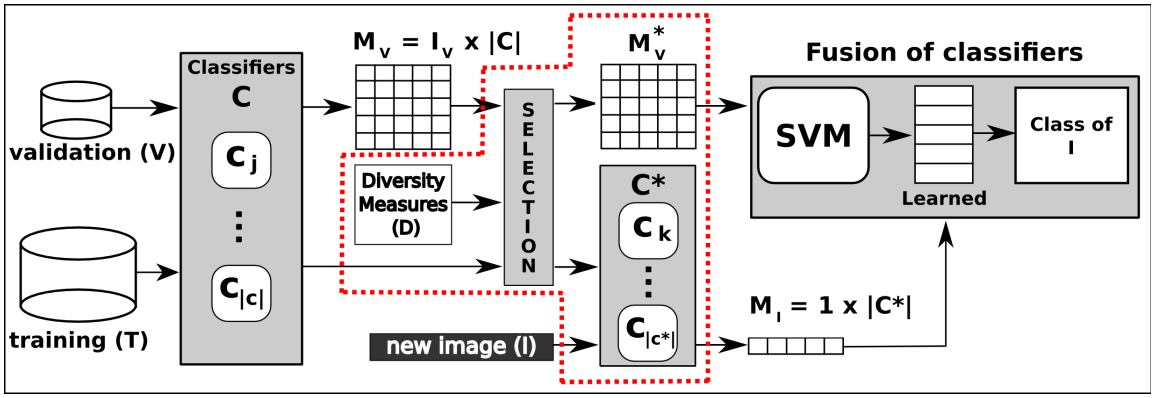


Fig. 1: Framework for classifier selection and fusion [10]. The classifier selection process is delimited by the dashed red line.

3.1 Formalization

Let $\mathcal{C}$ be the set of classifiers $\mathcal{C} = \{c_{11}, c_{12}, \ldots, c_{22}, \ldots, c_{|\mathcal{L}||\mathcal{F}|}\}$, with $c_{ij} = (l_i, f_j)$, where l_i is a learning method (e.g., Decision Tree, Naïve Bayes, kNN, etc.), and f_j is an image descriptor (e.g., Color Histogram). $|\mathcal{C}| = |\mathcal{L}| \times |\mathcal{F}|$, where $\mathcal{L}$ and $\mathcal{F}$ are sets of available learning methods and image descriptors, respectively. Initially, all classifiers $c_k \in \mathcal{C}$ $(1 < k \leq |\mathcal{C}|)$ are trained on a training set T. Next, classifier results on a validation set V are computed and stored into a matrix M_V, where $|M_V| = |V| \times |\mathcal{C}|$ and $|V|$ is the number of regions in a validation set V. The actual classes of training and validation data points are known *a priori*.

The objective of our framework is to select a set $\mathcal{C}^* \subset \mathcal{C}$ of classifiers that are good candidates to be combined. $\mathcal{C}^*$ is determined by using M_V as input in an approach that exploits diversity measures (see Section 3.2). Note that $\mathcal{C}^*$ can be used to compute a new matrix $M_V^* \subset M_V$. Each selected classifier in $\mathcal{C}^*$ is used to determine the class of an unknown instance. The outcomes of those classifiers are later combined by a novel fusion technique (majority voting, SVM, etc.), which is responsible for defining the class of the unknown instance. Fig. 1 illustrates the framework FSVM for combining classifiers.

3.2 Selection based on Consensus

Fig. 2 illustrates the adopted five-step approach for selecting classifiers based on diversity measures, previously introduced in [10]. First, diversity measures (set $\mathcal{D}$ in Fig. 2) are used to assess the degree of agreement among available classifiers in $\mathcal{C}$ by taking into account the M_V matrix previously computed. That step is represented by arrow (a) in Fig. 2. Pairs of classifiers are then ranked according to their diversity score. Each diversity measure defines a different ranked list

and, at the end of this step, a set $\mathcal{R}$ of ranked lists is produced (arrow (b)). In the following, a novel set of ranked lists $\mathcal{R}^t$ is computed by selecting the top t pairs of classifiers from each ranked list in $\mathcal{R}$ (arrow (c)), and a histogram $\mathcal{H}$ that counts the number of occurrences of a classifier in all ranked lists of $\mathcal{R}^t$ is computed (arrow (d)). Finally, the most frequent classifiers in $\mathcal{H}$, whose accuracy is greater that a given threshold $\mathcal{T}$, are combined by a fusion approach (arrow (e)). $\mathcal{T}$ is a threshold defined in terms of the average accuracy among all classifiers using the validation set V.

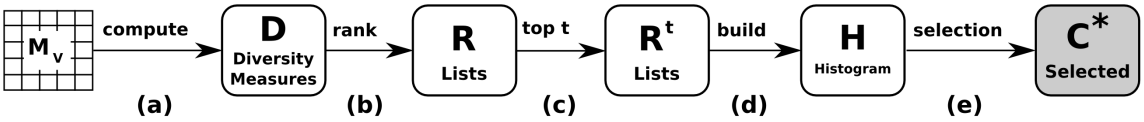


Fig. 2: The five steps for classifier selection are: (a) Computation of diversity measures from the validation matrix M_V; (b) Ranking of pairs of classifiers by their diversity measure scores; (c) Selection of the top $t = 100$ ranked pairs of classifiers; (d) Computation of a histogram $\mathcal{H}$ that counts the number of occurrences of each classifier; (e) Selection of classifiers $|\mathcal{C}^*|$ based on their occurrence in $\mathcal{H}$ and on a defined threshold $\mathcal{T}$ [10].

3.3 Selection based on Kendall Correlation

Let $\mathcal{C}$ be the set of classifiers created by the combination of learning methods and image descriptors. Let $\mathcal{P} = \{p_1, p_2, \ldots, p_{|\mathcal{C}\times\mathcal{C}|}\}$ be a set of all possible pairs of classifiers, i.e., $p_l = (c_i, c_j)$, where $(c_i, c_j) \in \mathcal{C} \times \mathcal{C}$. Let $\mathcal{D} = \{d_1, d_2, \ldots, d_{|\mathcal{D}|}\}$ be a set of diversity measures, such that each diversity measure $d_k \in \mathcal{D}$ defines a distance function $\rho : \mathcal{P} \rightarrow \mathbb{R}$, where $\mathbb{R}$ denotes real numbers. Equations described in paper [18] that define different criteria for implementing the function ρ. Consider $\rho(p_l) \geq 0$ for all $p_l \in \mathcal{P}$ and $\rho(p_l) = 0$, with $p_l = (c_i, c_j)$, if $c_i = c_j$. The distance $\rho(p_l)$ among all pairs of classifiers $p_l = (c_i, c_j) \in \mathcal{C} \times \mathcal{C}$ can be computed to obtain a $|\mathcal{C}| \times |\mathcal{C}|$ distance matrix A. Given a diversity measure $d_k \in \mathcal{D}$, we can compute a ranked list $\mathcal{R}_{d_l}$ by taking into account the distance matrix A. The ranked list $\mathcal{R}_{d_l}$=$\{p_1, p_2, \ldots, p_{|\mathcal{C}\times\mathcal{C}|}\}$ (where $p_l = (c_i, c_j)$ is a pair of classifiers) can be defined as a permutation of the collection $\mathcal{P}$, such that, if p_l is ranked at lower positions than p_m, i.e., p_l is ranked before p_m, then $\rho(p_l) < \rho(p_m)$. In this way, pairs of classifiers are ranked according to their agreement score defined in terms of a diversity measure.

We exploit the correlation of ranked lists of pairs of classifiers to select the more appropriate ones to be combined. In this thesis, we use the *Kendall* tau rank correlation coefficient (τ) [15] to measure the degree of concordance between two different ranked lists of the same set of observed samples. The *Kendall* correlation $\tau(\mathcal{R}_{d_i}, \mathcal{R}_{d_j})$ between two ranked lists $\mathcal{R}_{d_i}$ and $\mathcal{R}_{d_j}$ is defined in terms of the number of concordant pairs NC in $\mathcal{R}_{d_i}$ and $\mathcal{R}_{d_j}$, the number of discordant pairs ND, and the number of positions n in the ranked lists.

We propose a novel strategy, named *Kendall* classifier selection (KCS), to define appropriate classifiers to be used in the classification framework presented in [10]. KCS makes use of the degree of agreement of different diversity measures.

This agreement is measured in terms of the *Kendall* correlation among ranked lists of classifiers.

Let d_{H_1} and d_{H_2} be the diversity measures with the highest correlation scores, which are defined by the *Kendall* correlation. Let $\mathcal{R}_{d_{H_1}}$ and $\mathcal{R}_{d_{H_2}}$ be the ranked lists of pairs of classifies defined by d_{H_1} and d_{H_2}, respectively. KCS defines the top-ranked pairs of classifiers in $\mathcal{R}_{d_{H_1}}$ and $\mathcal{R}_{d_{H_2}}$ as the most appropriate ones to be used in the classification framework presented in [10]. We also tested in our experiments selected classifiers defined in terms of the lowest correlated diversity measures (d_{L_1} and d_{L_2}). In this case, we use classifiers defined in the top-ranked positions of $\mathcal{R}_{d_{L_1}}$ and $\mathcal{R}_{d_{L_2}}$.

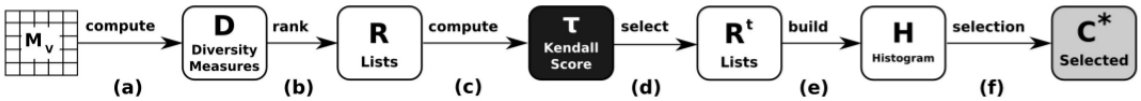


Fig. 3: The six steps for new classifier selection are: (a) Compute diversity measures from the validation matrix M_V; (b) Sort $\mathcal{R}$ lists by diversity measure scores; (c) Compute *Kendall* correlation coefficients among all ranked lists of classifiers $\mathcal{R}$; (d) Select $\mathcal{R}_{d_{H_1}}$ and $\mathcal{R}_{d_{H_2}}$ or $\mathcal{R}_{d_{L_1}}$ and $\mathcal{R}_{d_{L_2}}$ ranked lists to be used in the next step; (e) $\mathcal{R}^t$ lists with top $t = 100$; (f) Compute a histogram $\mathcal{H}$ that counts the number of occurrences of each classifier; (g) Select the most appropriate classifiers $|\mathcal{C}^*|$ based on their occurrence in $\mathcal{H}$ and a defined threshold $\mathcal{T}$ [12].

Figure 3 summarizes in six steps the new approach for selecting classifiers based on *Kendall* correlation. It is important to highlight that all steps regarding the selection of classifiers for fusion are performed during the training phase of the decision-making framework. Using a validation set separated during training allows us to evaluate different descriptors and learning techniques, assess their outcomes when classifying the validation examples, and properly selecting, by means of the proposed *Kendall*-based methodology, the most suitable classifiers for deployment during testing.

3.4 Selection based on Rank Aggregation

We propose to use multiple diversity and evaluation measures (Kappa, Tau, and accuracy) to determine which classifiers should be combined to improve the classification results in a given problem. Recall that different diversity measures would rank pairs of classifiers differently. In many situations, *rank aggregation* methods have been used as a way of obtaining a consensus ranking when multiple ranked lists are computed by different approaches. Rank aggregation has also been treated as the task of combining different ranked lists (or scores) in order to obtain a single, and more accurate, ranked list. For classification tasks, the combination with the lowest error occurs when the classifiers being combined are non-correlated (high diversity) and yields high accuracy rate [3]. In our approach, each considered measure (both diversity and evaluation measures) produces a ranked list of pairs of classifiers. A rank aggregation method combines all ranked lists, producing a single combined ranked list, which is used to identify pairs of

classifiers with good classification performance and high diversity. In the next section, we formally define the proposed rank aggregation approach.

Figure 4 summarizes the six-step approach for selecting classifiers based on rank aggregation.

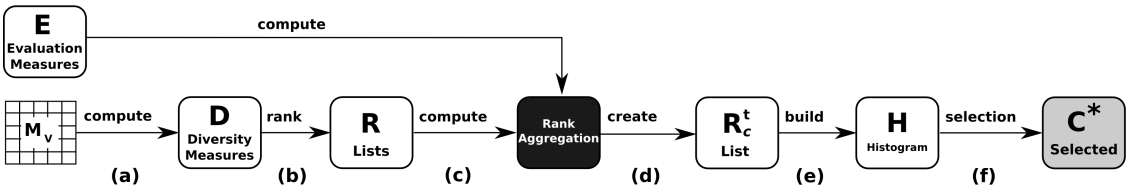


Fig. 4: The six steps of the new classifier selection are: (a) Compute diversity measures from the validation matrix M_V; (b) Sort $\mathcal{R}$ lists according to scores of diversity measures; (c) Compute rank aggregation using all ranked lists of classifiers ($\mathcal{R}$) and evaluation measures (E); (d) Create a single list $\mathcal{R}_c^t$, which list has the top $t = 100$; (e) Compute a histogram $\mathcal{H}$ that counts the number of occurrences of each classifier; (f) Select the most appropriate classifiers $|\mathcal{C}^*|$ that satisfy a defined threshold $\mathcal{T}$ [8].

4 Experiments and Discussion

This section presents some of several experiments that we performed to evaluate the robustness of our fusion framework with each selection process [8, 10, 12].

4.1 Effectiveness Analysis

In these experiments, six fusion techniques were compared: our approach using SVM (FSVM-PK-49) considering $|C| = 49$, two Adaboost approaches (BOOST-DEFAULT and BOOST-49), Bagging (BAGG-49), and Majority Voting (MV-49). Recall that using $|C| = 49$ means that all available classifiers (7 learning methods $\times$ 7 image descriptors) are employed in the fusion process. FSVM-PK means the SVM technique uses a polynomial kernel to combine different simple classifiers in our approach. Furthermore, we have included the best single classifier (no fusion) between all tested learning methods.

Table 1 presents the results obtained for each fusion technique and the best single classifier using one of four datasets considered in the work and considering three different evaluation measures (Accuracy, Kappa, and Tau). Notice that BOOST and BAGG techniques show up with the suffix ALL, which means the concatenation of the feature vectors produced by the seven different image descriptors considered. Thus BAGG-49-ALL and BOOST-49-ALL techniques refer to the use of 49 iterations and seven image descriptors (hybrid fusion).

In these experiments, our late fusion approach (FSVM-PK-49), which uses meta-learning on the outputs of all available classifiers yielded a slightly better classification result considering the three evaluation measures, when compared to other techniques in any tested datasets. However, the achieved results when considering the selection of the most appropriate descriptors and learning methods automatically during the fusion process are more interesting.

Important to note that BOOST and BAGG techniques use a fusion hybrid (feature and decision level fusion) to achieve similar results to our framework that uses only decision level fusion. We also computed the confidence intervals to verify if the results obtained by the proposed framework differ from those observed for the baselines. We could observe that FSVM-PK-49 has no statistical difference between the best baseline in the Caltech dataset. Furthermore, our late fusion approach (FSVM-PK-49) achieves similar results to BOOST-49-ALL (hybrid fusion).

Dataset	**Techniques**	**Measures**		
		Accuracy	**Kappa**	**TAU**
Caltech	**FSVM-PK-49**	**47.05%±1.77**	**0.45±0.02**	**0.46±0.02**
	BOOST-49-ALL	46.90%±0.63	0.45±0.01	0.46±0.01
	BAGG-49-ALL	43.01%±1.38	0.41±0.01	0.42±0.01
	SVM-PK-LAS	41.30%±0.41	0.39±0.00	0.40±0.00
	MV-49	41.02%±0.46	0.38±0.00	0.40±0.00
	BOOST-DEFAULT-ALL	39.92%±0.57	0.38±0.01	0.39±0.01

Table 1: Classification effectiveness of the proposed framework and baselines [10].

4.2 Training Set Size Impact

This section shows a behavioral study among the classifiers compared in Table 1 using reduced training sets. In our experiments, we conducted a study considering five different sizes for the training set (T): 8%, 16%, 33%, 67%, 100%, which represents 5%, 10%, 20%, 40% and 60% of the entire datasets, respectively. These subsets have been selected from original training set. We use again the 5-fold cross-validation protocol previously adopted in our experiments.

Fig. 5 shows the results for one of four datasets (Caltech) used in our work. The x-axis denotes the number of images in the training set and the y-axis represents the average accuracy in the testing set. The FSVM-PK-49 approach using a subset of 8% of training set achieves 39.52% of accuracy. In the same training set, BOOST-49-ALL yields 32.33%, which means that our approaches have a gain of more than 19% compared to the best baseline. In the subset 16%, our approaches are still better and achieve accuracy results of 40.67% (FSVM-PK-49) against 37.24% of the BOOST-49-ALL. That represents a gain of more than 7% in classification accuracy. From the subset 33% to 100%, the best baseline yields similar performance to our approach. In summary, we can see that the proposed approach are able to learn from small training sets.

4.3 Classifier Selection Approaches

This section discusses the results regarding the effectiveness and efficiency of the proposed framework using one of the two different datasets performed in our work. In this case, the Urban dataset has been used. In our experiments, we have used *Double-Fault Measure* (DFM), *Q-Statistic* ($QSTAT$), *Interrater Agreement* k (IA), *Correlation Coefficient* ρ (COR), and *Disagreement Measure* (DM). Our framework is denoted as FSVM-NORM-$|\mathcal{C}^*|$, where NORM means

the normalized polynomial SVM kernel used in our experiments and $|\mathcal{C}^*|$ is number of simple classifiers that will be combined by the SVM-based meta-learning technique.

Table 2 shows the average kappa indices for all performed experiments with Urban dataset. The columns refer to the number of classifiers $|\mathcal{C}^*|$, which have values range from 5 to 36, where 5 is the lowest number of classifiers selected and 36 is the total amount of possible classifiers that can be selected (six image descriptors and six learning methods result in 36 different simple classifiers).

In these experiments, we compare three selection strategies: *Consensus*, *Kendall*, and Rank Aggregation. *Consensus* refers to the strategy described in Section 3.2, which uses all the five diversity measures in the selection process. *Kendall*, in turn, refers to the strategy described in Section 3.3, which uses the two less correlated diversity measures (in the case, IA and $QSTAT$) in the selection process. These diversity measures were defined according to an *a priori* correlation analysis. Finally, Rank Aggregation refers to the use of the rank aggregation strategy described in Section 3.4. In Table 2, we highlight in blue the number minimum of classifier that each approach needs to achieve similar result than the FSVM-NORM-$|\mathcal{C}^*|$ using all classifiers ($|\mathcal{C}^*| = 36$). *Consensus* approach need to use $|\mathcal{C}^*| = 15$ classifiers. *Kendall* approach achieves similar result using $|\mathcal{C}^*| = 10$ classifiers. Finally, the rank aggregation approach with configuration $Kappa{+}DFM{+}IA{+}QSTAT$ is able to yield very effective results with only $|\mathcal{C}^*| = 5$ classifiers.

We also computed the confidence intervals to verify if the results obtained by the proposed fusion approach differ from those observed for the baselines. We could observe that our approach achieves similar results to those observed for almost all baselines compared, but with fewer classifiers. Please, refer to the associated thesis[1] for more details regarding performed experiments.

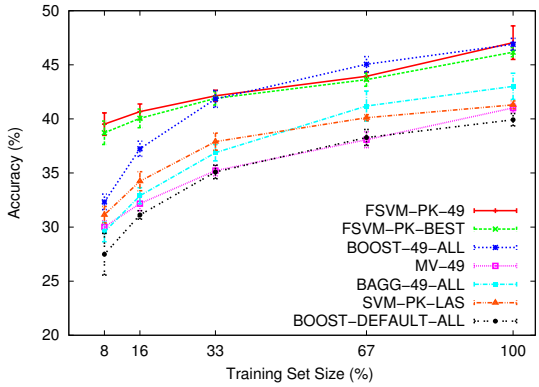


Fig. 5: Accuracy scores of all classifiers using training sets with different sizes [10].

Urban Dataset				
Approaches	**Number of Classifiers $\|\mathcal{C}^*\|$**			
	5	**10**	**15**	**36**
***Consensus* [10]**	0.564	0.570	**0.594**	0.612
***Kendall* [12]**	0.566	**0.592**	0.604	0.612
Rank Agg. [8]	**0.593**	0.592	0.602	0.612

Table 2: Kappa indices computed for each selection approach using different number of classifiers ($|\mathcal{C}^*|$) in the Urban dataset.

5 Conclusion

This work presented a framework for selection and fusion of simple classifiers using diversity measures and meta-learning on top of classifier outcomes. The

[1] `www.ic.unicamp.br/~ffaria/ffaria_final_thesis.pdf`

main novelty of this work relies on the use of diversity measures to determine which learning and image descriptor methods are more suitable to be combined in a given classification problem. Thus, three different strategies for classifier selection have been proposed (*Consensus*, *Kendall* correlation, and Rank Aggregation). This work resulted in papers in three important international journals [7,8,10] and three conference papers [9,11,12]. In addition, two articles have been submitted to international journals.For future work, we plan to investigate additional strategies and metrics for improving the classifier selection process, find the optimal diversity measures set for each application, test non-pairwise diversity measures, and perform experiments in other application domains.

References

1. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006)
2. Breiman, L.: Bagging predictors. Mach. Learn. 24(2), 123–140 (1996)
3. Croft, W.B.: Combining approaches to information retrieval. In: Croft, W.B. (ed.) Advances in Information Retrieval, vol. 7, pp. 1–36. Springer US (2002)
4. Didaci, L., Fumera, G., Roli, F.: Diversity in classifier ensembles: Fertile concept or dead end? In: MCS. pp. 37–48 (2013)
5. Du, P., Xia, J., Zhang, W., Tan, K., Liu, Y., Liu, S.: Multiple classifier system for remote sensing image classification: a review. Sensors 12(4), 4764 (2012)
6. Duin, R.P.W., Pekalska, E.: Open issues in pattern recognition. In: CORE. vol. 30, pp. 27–42 (2005)
7. Faria, F.A., P. Perre, P., Zucchi, R.A., Jorge, L.R., Lewinsohn, T.M., Rocha, A., Torres, R.d.S.: Automatic identification of fruit flies (diptera: Tephritidae). JVCIR. (2014)
8. Faria, F.A., Pedronette, D.C.G., dos Santos, J.A., Rocha, A., Torres, R.d.S.: Rank aggregation for pattern classifier selection in remote sensing images. JSTARS 7(4), 1103–1115 (April 2014)
9. Faria, F.A., dos Santos, J.A., Rocha, A., da S. Torres, R.: Automatic classifier fusion for produce recognition. In: SIBGRAPI. pp. 252–259 (2012)
10. Faria, F.A., dos Santos, J.A., Rocha, A., da S. Torres, R.: A framework for selection and fusion of pattern classifiers in multimedia recognition. PRL 39(0), 52 – 64 (2014)
11. Faria, F.A., dos Santos, J.A., da S. Torres, R., Rocha, A., Falcão, A.X.: Automatic fusion of region-based classifiers for coffee crop recognition. In: IGARSS. pp. 2221–2224 (2012)
12. Faria, F.A., dos Santos, J.A., Sarkar, S., Rocha, A., da S. Torres, R.: Classifier selection based on the correlation of diversity measures: When fewer is more. In: SIBGRAPI. pp. 16–23. Arequipa, Peru (august 2013)
13. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. ICML pp. 148–156 (1996)
14. Jurek, A., Bi, Y., Wu, S., Nugent, C.: Classification by cluster analysis: a new meta-learning based approach. In: MCS. pp. 259–268 (2011)
15. Kendall, M.G.: A New Measure of Rank Correlation. Biometrika 30(1/2), 81–93 (Jun 1938)
16. Kim, D., Baek, Y., Kim, W.: Reducing overfitting of adaboost by clustering-based pruning of hard examples. In: ICUIMC. p. 90 (2013)
17. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience (2004)
18. Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Mach. Learn. 51, 181–207 (2003)
19. Morais, E., Goldenstein, S., Ferreira, A., Rocha, A.: Automatic tracking of indoor soccer players using videos from multiple cameras. In: SIBGRAPI. pp. 174–181 (aug 2012)
20. Rocha, A., Papa, J.P., Meira, L.A.A.: How far do we get using machine learning black-boxes? IJPRAI 26(2), 1261001–1–1261001–23 (2012)
21. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR. pp. I–511–I–518 (2001)
22. Weber, R., Schek, H., Blott, S.: A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In: VLDB. pp. 194–205 (1998)

Affinity Analysis between Researchers using Text Mining and Differential Analysis of Graphs

Luís Trigo[1], Pavel Brazdil[1,2]

[1]LIAAD-INESC Tec, [2]FEP, Univ. of Porto

{lptrigo,pbrazdil}@inescporto.pt

Abstract. Finding people with similar skills within a domain may provide an important support for managing research centers. The academic production, albeit in an unstructured format, is easily accessible. Thus, we resort to sources on the web - academic and bibliographic databases - to uncover the affinities among researchers. What interests us most are affinities that are not yet evidenced by co-authorship. Besides, of interest are also other outputs of the method in the form of subgroups and the researchersthat play an important role in them.

Keywords: Web Mining, Text mining, Clustering, Social Network Analysis, Differential Analysis of Graphs.

1 Introduction

Researchers seek to discover other researchers with similar interests to follow their work and plan future collaborations. At management level, it enables identifying suitable researchers for a given task, which precedes the implementation of partnerships with other institutions and researchers policies. Another advantage of this analysis is that it goes beyond the formal hierarchical framework within the organization, thereby revealing its unknown connections that can be followed up.

The main scientific contribution is beyond re-using standard techniques of text mining to bibliographic databases, but rather using these techniques to obtain two kinds of graphs, co-authorship and affinity graphs, and exploring a differential analysis with the aim of identifying new useful knowledge.

Our aim is to focus on affinity analysis between certain research centers for various reasons: First, the outcome of the study may be useful to these centers. It may propose that certain collaborations be initiated. Besides, the outcomes of automatic analysis may be easily verified by some members of these centers. This research could be extended later to cover a larger set of centers.

Regarding the discovery of similarities between researchers, Price et al. (2010) developed a methodology for the Web, called *SubSift*, establishing profiles for researchers

on the basis of researchers' publications. Based on these profiles, a typical Information Retrieval task is performed aiming to compare the papers submitted to a scientific conference (playing the role of Query in IR) with different profiles, in order to optimize the task of distributing articles to review.

Rogosky and Goldstone (2002) have pointed out that, in the context of a conceptual network, the meaning of a concept - here "*researcher*" - depends on the relationship with the other concepts in the conceptual framework. Thus we analyze networks of affinities and of particular interest are those that are not covered by the simple co-authorship connections. To uncover these we have to resort to many different techniques, including web mining, text mining, social network analysis, sub-graph discovery and differential analysis of graphs and graph analysis.

The main steps of the method are described in the following.

2 Methodology

This section presents the main steps undertaken to uncover the unknown information regarding affinities. The method involves the following steps:

1. Query user to obtain names of institutions and websites;
2. Web mining to identify researchers' names;
3. Web / Text mining to process researchers' publications;
4. Elaboration of similarity matrix and visualization using graphs;
5. Application of sub-group discovery to the affinity graph;
6. Elaboration of a co-authorship graphs and differential analysis of graphs;
7. Identification of important nodes (researchers) in the graph.

The details about all these steps are given in the following.

2.1 Query user to obtain names of institutions

So far, our work is in a prototype stage and so we have applied the method to two closely related R&D units, INESC TEC[1] – LIAAD [2] and CRACS [3]. The total number of researchers does not exceed several dozen. Our plan is to apply the method to larger set of units in future, such as the whole INESC or some Faculties of the University of Porto or other Universities.

2.2 Web mining to identify researchers' names

Each research institution has normally a webpage listing their researchers. Lists of researchers can be extracted easily by building an expression in the *XPath* query language to obtain their names from the website.

Regarding tools to extract and process the data, we chose R with its *tm* package for part of text mining, the *XML* package for web mining, as well as *igraph* and *sna* packages for clustering and social network analysis.

2.3 Web / Text mining to process researchers' publications

Each researcher name can be inserted in the search URL for the DBLP [4] which enables direct access to each researcher list of publications. The retrieval of publications can be done automatically, using *XPath* expressions. However, a problem of *named entity identification* arises here. This is because researchers may have several variants of their name. Thus several entries may exist in the bibliographic database for the same researcher, each associated with a particular variant of his/her name. Typically, one of the variants will appear on the institution site. This name may not match the name used in the bibliographic database.

Another problem is that we may have several investigators with the same name in the bibliographic database. One of the techniques used by Bugla (2009) is the following. To determine whether a given publication of P in some bibliographic database should be attributed to person P' on a given site, a check is made whether both (i.e. P and P')have the same home institution.

Regarding the particular bibliographic database, we have chosen DBLP, because it is an open and comprehensive bibliographic database in the field of computer science. Currently, we are considering to use Authenticus instead, as its designwas based on Bugla's work within a project from the University of Porto, and has the advantage that it retrieves publications from several other bibliographic databases (incl. e.g. SCOPUS).

The publications titles are extracted into plain text files, each representing a particular author. The text files are retrieved and preprocessed in the usual manner. We use *BoW*representation, remove numbers, stop-words, punctuation and other spurious elements. After this task, the list of documents is transformed into a document-term vector representation with *tf-idf* weighting (Feldman and Sanger, 2007).

2.4 Elaboration of similarity matrix and visualization using graphs

The vector representation described in the previous step is used to generate the cosine similarity matrix. This matrix can be visualized in the form of a graph and is used as the basis for further processing. Fig. 2 shows an example of an affinity graph, where all links (similarities) below a given threshold have been considered irrelevant and hence removed.

2.5 Application of community discovery to the affinity graph

After transforming the similarity matrix into a graph format, we use the community discovery algorithm called *Walktrap* (Pons and Latapy, 2006). This technique finds densely connected sub-graphs, also defined as communities, through random walks. It assumes that short random walks tend to stay in the same community.

The hierarchical agglomerative approach is based on a measure of distance between vertices (node to node) and an example of the output of clustering can be visualized in the following figure (Fig. 1). An optimal level of modularity of the network, based on the weighted connections between internal and external community is used by the algorithm to identify non-hierarchic communities. In our example below, the method identified three communities, identified as L-ML, L-OR and Con the basis of data gathered in 2011.

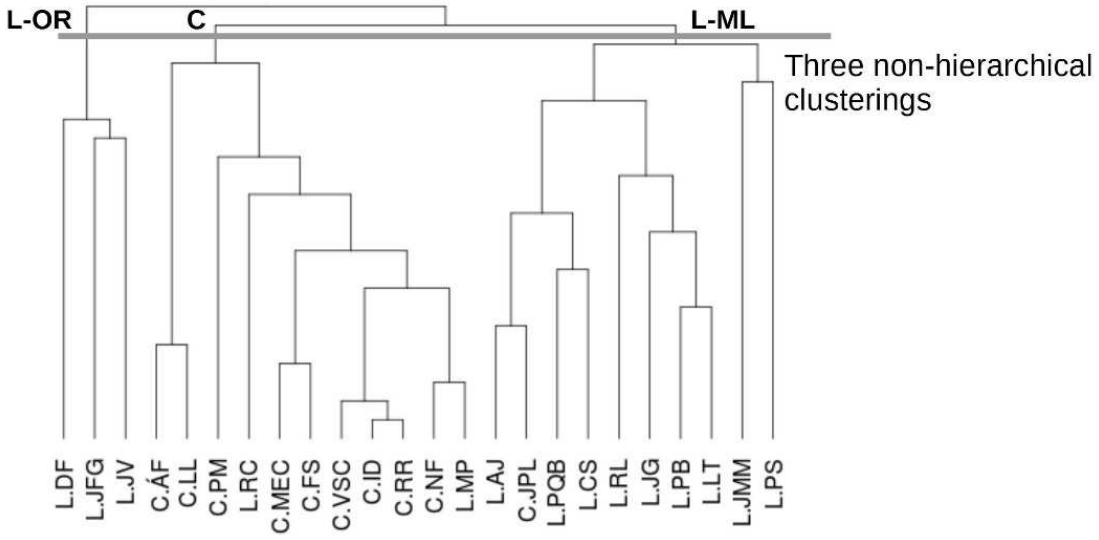


Fig.1. Dendrogram generated by the Walktrap clustering algorithm

Different discovered communities can be superimposed on the graph. The result can be seen in Fig. 1, where different communities uncovered by the Walktrap have been identified by ellipses.

The communities discovered correspond well to the organizational structure of the two studied entities. One of the interesting issues to study in the future is - what are the differences between the two organizational structures This differences can suggest that a possible re-organization could be considered by the management in future.

The Researcher Affinity Network graph (Fig. 2) that was generated enables to perform a visual analysis ofrelationships between researchers. The thickness of the edges

represents the similarity weight between pairs of researchers. Node dimension reflects the number of publications at DBLP for each author.

The same graph shows also several communities that were detected by Walktrap. Further analysis of the community structure is presented in section 2.7.

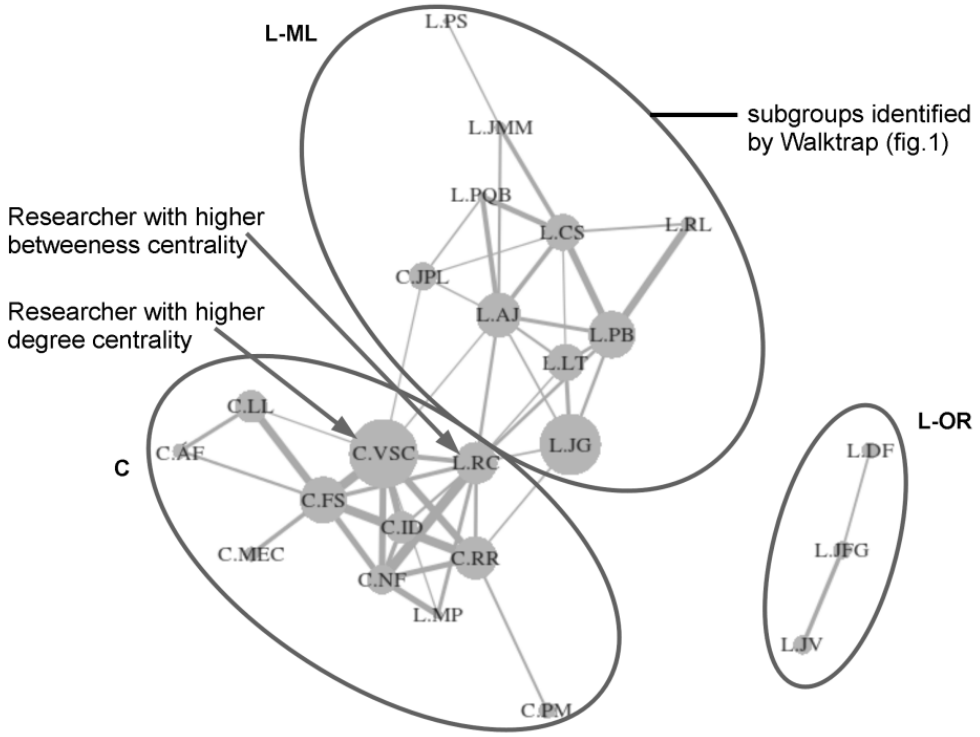


Fig.2. Researcher Affinity Network with communities identified by the *Walktrap* algorithm

2.6 Elaboration of a co-authorship graph and differential analysis of graphs

The generation of the co-authorship graph is a relatively simple matter. A link between authors A and B is introduced, if they are co-authors of at least one of the papers. After constructing the affinity graph (G1) and co-authorship graphs (G2), we can proceed to the next step which involves differential analysis. This involves constructing a graph that is basically a difference between G1 and G2. The next figure shows an example.

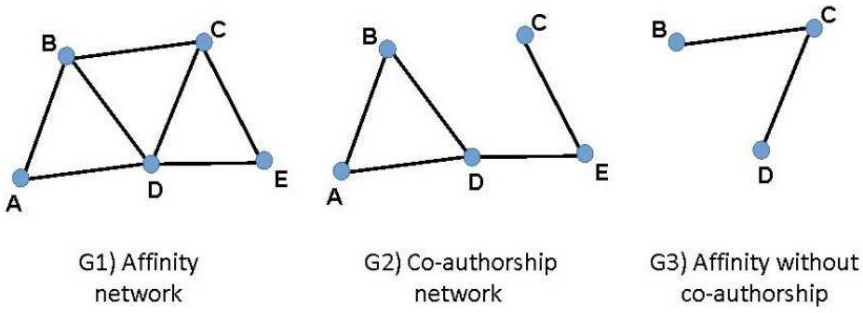


Fig.3. Example of differential analysis

2.7 Identification of important nodes (researchers) in the graph.

The affinity network enables to calculate certain measures of importance of the researchers within their community and in the context of different communities. This involves, for instance, *degree centrality*and *betweenness* centrality among others (Wasserman and Faust, 1994).

Some centrality measures can be computed to account for different weights of the connections, as shown in the table below. The degree centrality is based on the number of connections to a vertex. The *betweenness centrality*indicates the number of times a vertex joins two other vertices on the shortest path. The eigenvector centrality shows the importance of vertices that connect to a given vertex.

Table 1 shows an example. The black marked cells indicate that the largest *degree centrality* is located in CRACS, while that the largest *betweenness centrality* belongs to a member of LIAAD (which was clustered with CRACS researchers), as noted in the previous figure visually. It also seems that, as pointed out by the *eigenvector centrality*, the influence within the community from the most central authors in LIAAD is more tenuous than the influence of the most central authors in CRACS.

	L.PB	L.RC	L.AJ	L.JG	C.RR	C.FS	C.VSC
Degree centrality	3.4	4.8	3.1	1.6	4	**5.6**	4.7
Betweenness centrality	41	**179**	130	18	45	88	16
Eigenvector centrality	0.07	0.37	0.06	0.06	0.38	**0.45**	0.44

Table 1. Centrality measures for some of the most relevant researchers

3 Conclusions and future work

The current work explores Web/Text mining for matching researcher names with their publication titles. This permits to retrieve researchers' publications and process the text files to construct a similarity matrix and a network of affinities.

Further processing leads to quite interesting results in the form of sub-graphs / communities. These can be compared to the formal organization structure. Further work involves differential analysis of graphs on the basis of the affinity and co-authorship graphs. The resulting differential analysis enables to identify pairs of people that could potentially benefit from working together.

In future work, we plan to design an adaptive method capable of retrieving researcher's names from sites with unknown format, or sites that may have altered the format. The method will rely on a fact that at least one researcher's name is known. The HTML/XML source code of the page will be analyzed with the aim of identifying the researcher's name there and elaborating a convenient Xpath expression leading to this name. The command will be adapted so as to be able to retrieve all researchers' names.

We also intend to process the abstracts and consider a substantially higher number of research centers and/or researchers representing some challenges to the process of clustering. To overcome these, we plan to use an incremental / data-streaming approach for this task(Gama et al., 2010).

A validation step needsto be added to the methodology. A brief online survey will be carried out for the most central researchers about who could be their potential collaborator. The outcome will be compared to the results of differential analysis.

An important problem in the text mining phase is that researchers from different domains use different vocabulary/terminology to describe the same things. This problem is difficult to overcome. We will try to use, as some others did, Wordnet and DBpedia (Leal et al, 2012) to identify synonyms and related words, although this may be harder for some specific domains, which may require specific dictionaries.

Regarding related management needs, we intend to go beyond similarity analysis and study the complementary analysis to uncover potential collaboration between different individuals. The aim of this analysis would be to identify two or more researchers with complementary skills for a given task.

Acknowledgments

This work is partially funded by FCT/MEC through PIDDAC and ERDF/ON2 within project NORTE-07-0124-FEDER-000059 and through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project FCOMP-01-0124-FEDER-037281.

References

1. INESC TEC, http://www.inescporto.pt/
2. LIAAD, http://www.liaad.up.pt/
3. CRACS, http://cracs.fc.up.pt/
4. DBLP, http://www.informatik.uni-trier.de/~ley/db/
5. Bugla, S.: Name identification in scientific publications, Master's thesis, University of Porto (2009)
6. Feldman, R., Sanger, J.: Text Mining Textbook: Advanced Approaches in Analysing Unstructured Data, Cambridge Univ. Press (2007)
7. Gama, J., Rodrigues, P. P., Spinosa, E. J., Ferreira de Carvalho, A. C.: Knowledge discovery from data streams. Boca Raton: Chapman & Hall/CRC (2010)
8. Goldstone, R., Rogosky, B. J.: Using relations within conceptual systems to translate across conceptual systems, Cognition, 84. pp. 295–320 (2002)
9. Leal, J. P., Rodrigues, V., Queirós, R.: Computing semantic relatedness using dbpedia. In OASIcs-OpenAccess Series in Informatics (Vol. 21). SchlossDagstuhl-Leibniz-ZentrumfuerInformatik(2012)
10. Pons, P., Latapy, M.: Computing communities in large networks using random walks, Journal of Graph Algorithms and Applications, Vol. 10, no. 2, pp. 191-218 (2006)
11. Price, S., Flach, P. A., Spielgler, S., Bailey, C., Rogers, N.: SubSift web services and workflows for profiling and comparing scientists and their published works, Sixth IEEE International Conference on e–Science (2010)
12. Wasserman, S., Faust, K.:, Social network analysis: methods and applications. Cambridge University Press, New York (1994)

NASSAU: Description Length Minimization for Boolean Matrix Factorization

Sanjar Karaev

Max-Planck Institute for Informatics
Campus E 1 4, D-66123 Saarbrücken, Germany
skaraev@mpi-inf.mpg.de

Abstract. Boolean Matrix Factorization (BMF) is an important tool in data mining that in many cases allows to increase interpretability for binary data. In BMF one decomposes a given binary matrix into a Boolean product of binary factors such that some cost function is minimized.
In this work we consider the description length of the data as the cost, which has been proven effective in uncovering true structure of the data and removing the noise. The argument is that structured data is easier to compress than the noise, and hence simpler models should be favored. We introduce a new BMF algorithm, that we call `NASSAU`, which traces back its history and correct its earlier mistakes. As turned out in our experiments, this approach performs reasonably well for both real-world and synthetic data.

Keywords: Matrix factorization, Boolean matrices, MDL principle, Random walks

1 Introduction

The amount of data that needs to be analyzed in today's world is enormous, rendering it impossible to explain it all without making simplifying assumptions. Typically, we strive to find a comparatively small number of patterns that occur in the data frequently and provide a good explanation for it.

Matrix factorization is a very common tool in data mining, allowing to extract a small number of frequent patterns. The matrix factorization problem is, given a matrix, find its decomposition into two or more factors that satisfy some constraints. Perhaps the most famous type of matrix factorization is the Singular Value Decomposition (SVD) [2]. However, despite its many useful mathematical properties, its results can sometimes be hard to interpret. For example, explaining negative values in the factors of SVD might be a problem. A very common approach to increase interpretability is to require the factors to have the same type as the input data, like for example in Nonnegative Matrix Factorization (NMF) [3], which restricts the factors to be nonnegative real matrices.

A special class of matrix decomposition is the Boolean Matrix Factorization problem (BMF) [1]. In BMF both the input matrix and the factors are binary,

and the matrix product is Boolean. The motivation for this is analogous to the nonnegativity restriction in NMF – we force the factors to be binary because so is the input, and Boolean product is natural on binary data. There are different forms of BMF depending on the cost function used.

In this work we use the Minimum Description Length principle (MDL) [4], which is very useful in tackling the problem of the trade-off between fitting the data well and having a simple model. In general, the more complexity we allow in the model, the better we can fit the data. However, having high model order comes at the cost of fitting the noise.

In this paper we present `NASSAU`, a new BMF algorithm that is designed to minimize the description length. Unlike the majority of the previously proposed BMF algorithms, it can correct its previous mistakes. `NASSAU` is quite robust to subtractive noise, which is especially beneficial for real-world data as in many domains there could be zeros simply due to the lack of observation.

2 Related Work

In BMF one decomposes a given binary matrix into the Boolean product of two binary matrices such that some cost is minimized. Perhaps the most intuitive and frequently used scoring function is the one that counts how many 0 - 1 errors were made in the reconstructed data. This objective was used in one of the first algorithms proposed for solving the BMF problem – `Asso` [1]. However, this is not the only reasonable choice for the cost function. For example, Miettinen and Vreeken [5] suggested to use the description length of the data as an alternative cost. In this case the problem becomes an application of the Minimum Description Length principle (MDL) [4]. It postulates that one should favor a model yielding the shortest description of the data. The intuition is that structured data is much more easily compressible than noise, and thus models providing shorter description of the data better capture its essence. In [5] the authors combined the MDL principle with the `Asso` algorithm [1] to tackle the model order selection problem in BMF. However, the `Asso` algorithm minimizes the number of $0-1$ errors, and MDL was only run as a postprocessing step to compare the results for different ranks and select the best one.

Lucchese, et al. [6] proposed an algorithm that they call `PANDA+`, which is capable of optimizing with respect to several different objectives, including the description length. However, they use a different encoding of the data than the one considered in this work (for details see [6] and [5]). Also the way `PANDA+` works is very different from our approach. In particular, it expects solid core patterns is the data, which it then tries to extend.

`PANDA+` is an extended version of `PANDA` algorithm, which was introduced in the authors' previous work [7].

Another area of research that has a strong link to BMF is tiling transaction databases [13]. A tile is a submatrix consisting only of ones. The objective is to cover as many ones as possible with tiles, without covering any zeros. The

problem is similar to BMF in that the original data is covered with rank-1 matrices full of ones, but unlike BMF, covering zeros with ones is not allowed.

3 Notation

Let $B \in \{0,1\}^{n\times k}$ and $C \in \{0,1\}^{k\times m}$. We denote by $B \circ C$ the Boolean product of matrices B and C. We will also require some notation for manipulating rows and columns of matrices. For any matrix A, we denote its i-th row by A_i and its j-th column by A^j. The matrix obtained by removing A^j (A_j) is denoted by A^{-j} (A_{-j}). In addition, if A is of size n-by-m and there is a column vector c of size n-by-1 and a row vector r of size 1-by-m, then we denote by $[A, c]$ and $\begin{bmatrix} A \\ r \end{bmatrix}$ matrices obtained by joining A with c and r respectively.

Let $A \in \mathbb{R}^{n\times m}$, $B \in \mathbb{R}^{n\times k}$, and $C \in \mathbb{R}^{k\times m}$ be binary matrices, and let $\langle B, C\rangle$ be a Boolean decomposition of A. Then we call B and C *factors* of this decomposition and for any $1 \leq l \leq k$, the rank-1 matrix formed by the vector pair $\langle B^l, C_l\rangle$, a *block*.

Let again $\langle B, C\rangle$ be a Boolean decomposition of a binary matrix A. We denote by $L(A, B, C)$ the description length of A with factors B and C.

4 BMF with MDL

In this work we study the Boolean Matrix Factorization problem with description length as a scoring function. We start this section by introducing the Minimum Description Length principle, which we then use to formulate the BMF problem.

4.1 Minimum Description Length (MDL) and encoding BMF

Following the MDL principle [4], we use the description length of the data as our cost function. MDL is a formalization of the Occam's razor, which states that faced with two competing models that describe the data equally well, one should choose the simpler one. In MDL the complexity of the model is expressed as the code length needed for the lossless compression of the data with this model. It is known [14] that structured data usually compresses much better than the noise, which makes MDL a usefool tool for noise removal.

The idea to use MDL as a scoring function for BMF comes from [5], where the authors introduces several possible encodings for the BMF problem. In this work we use the encoding that was deemed the best in the above paper, which the authors call *data to model encoding*. Here we will describe the main ideas behind this encoding scheme. More detailed explanation can be found in [5].

Assume that we are given a binary n-by-m matrix A and its decomposition $A \approx B \circ C$, where $B \in \{0,1\}^{k\times m}$ and $C \in \{0,1\}^{k\times m}$. The description length of this factorization can be represented as follows

$$L(A, B, C) = \alpha + L(B, C) + L(A \,|\, B, C).$$

Here α is a constant term that does not depend on a particular factorization (see [5] for details on how it is computed), $L(B, C)$ is the description length of the model (factors B and C) and $L(A \mid B, C)$ is the description length of the data (matrix A) given the model. $L(B, C)$ can be split into two parts $L(B, C) = L(B) + L(C)$, where each summand corresponds to one factor. Each column of B is independently encoded as a binary vector. Since there are $\binom{n}{|B^i|}$ binary strings that have the same length and number of ones as B^i, it can now be identified by two integers: one encoding the number of nonzero elements in B^i, (maximum n) and the other encoding the index of B^i among all binary strings having the same profile (maximum $\binom{n}{|B^i|}$). Thus, encoding B^i requires $\log n + \log \binom{n}{|B^i|}$ bits [5]. Since B has k column we have

$$L(B) = k \log(n) + \sum_{i=1}^{k} \log \binom{n}{|B^i|} . \tag{1}$$

C can be encoded analogously to the above by encoding its rows as binary strings.

It now remains to encode the description of the data given the model, or the error matrix $E = XOR(A, B \circ C)$.

It can be split it into positive and negative parts, E^+ and E^-, where $E^+_{ij} = 1$ if and only if $A_{ij} > (B \circ C)_{ij}$ and $E^-_{ij} = 1$ if and only if $A_{ij} < (B \circ C)_{ij}$ [5]. In order not to assume any structure in the error matrix, E^+ and E^- are encoded as binary strings in the same way as columns of B. This yields

$$L(E^+) = \log(mn - |B \circ C|) + \log \binom{mn - |B \circ C|}{|E^+|}$$

and

$$L(E^-) = \log(|B \circ C|) + \log \binom{|B \circ C|}{|E^-|} .$$

4.2 Boolean Matrix Factorization (BMF)

In this work we study the following problem.

Definition (**Boolean matrix factorization**). *Given a binary matrix $A \in \{0, 1\}^{n \times m}$, the Boolean Matrix Factorization (BMF) problem is to find binary factor matrices B and C such that the total description length of the data when represented as a Boolean product of B and C is minimized. We denote the description length of a Boolean decomposition of matrix A into factors B and C by $L(A, B, C)$.*

Note that a more traditional way of defining the BMF problem is to look for a decomposition of a given rank (see e.g. [5]). We do not require the rank as an input, but rather aim to find the best decomposition regardless of it, when finding the right rank becomes a byproduct.

5 Algorithm

In this section we present a new algorithm, which we call `NASSAU` (Algorithm 1), for solving the BMF problem. In a nutshell it works as follows: start with an empty factorization, then iteratively add blocks until there is no more gain in the description length, and then go through the earlier blocks and attempt to improve them.

The basic building block for `NASSAU` is a routine called `FindBlock` (algorithm 3) which, given the input matrix A and current factors B and C, finds (approximately) an optimal block to be added to the factors. It requires a set of candidate column vectors on the input, which it uses to start building potential blocks to be added to the current factorization. It then chooses the one that yields the best change to the description length. Subsection 5.3 describes this process in detail.

We run `FindBlock` repeatedly until the description length does not decrease anymore. Periodically we update all the blocks found up to this point to fix some of the suboptimal decisions made in the past. This is done using the routine `CyclicUpdates` (Algorithm 2), which goes though the current factors updating one block at a time with `FindBlock`. Finally, when adding a new block does not improve the cost, we already have a reasonable approximation of the optimal rank of the data. Next, we run `CyclicUpdates` yet again to make final fixes to the obtained blocks.

`NASSAU` accepts several parameters that control its execution. The first parameter t represents the initial temperature for the `CyclicUpdates` function (it is explained in Subsection 5.2). It is updated using another parameter τ. Parameter θ is used within `FindBlock` routine and is explained in Subsection 5.3. Finally, M determines how frequently we update current blocks.

5.1 Finding candidates

The candidate vectors for `FindBlock` are found using restarted random walks on the bipartite graph corresponding to the input matrix. A restarted random walk starts from a certain node in the graph and on each iteration either returns to the origin with probability ϵ or visits one of the neighbors of the current node with probability $(1 - \epsilon)/d$, where d is the number of neighbours. Candidate vectors are then obtained by thresholding the stationary solution of the corresponding Markov chain.

5.2 `CyclicUpdates`

`CyclicUpdates` is used in `NASSAU` to improve the found factors by backtracking the decision history and updating previous blocks with `FindBlock`. The motivation for this is that some of the blocks might have become redundant after we have found new ones. Note that the objective is not strictly decreasing – `FindBlock` is a heuristics and is not guaranteed to improve the current block. However, we might still want to keep the update to avoid getting stuck in a local

minimum. We apply a technique similar to simulated annealing – we always accept an update if it decreases the cost, otherwise we might still accept it with a probability proportional to current temperature. This is different from the standard simulated annealing in that we use a deterministic approach to find the next step and only use randomization to decide whether to actually apply it.

5.3 `FindBlock`

`FindBlock` finds one more block to be added to the current factorization. It aims to minimize the number of 0 - 1 errors rather than the description length. The reason why we use this objective is that direct rank-1 optimization of the description length is a complicated task due to its complex nature, and 0 - 1 error proved to be a good proxy for it in these settings.

`FindBlock` starts with a set of candidate column vectors. For each candidate it finds a corresponding row vector such that together they would form a good block in terms of 0 - 1 error (line 9). Observe that once an elements has been covered, it does not matter if we cover it again. Hence, we should disregard already covered elements when computing the score. This is done by introducing a binary weight matrix W (line 4) that has ones only at the positions corresponding to not yet covered elements of A. Note that setting element $c_l = 1$ corresponds to using b to cover l-th column of A, whereas setting $c_l = 0$ corresponds to covering it with all zeros. `FindBlock` accepts parameter θ that controls when the algorithm would cover a column of A with vector b. We set c_l to 1 if and only if using b to cover A^l would result in an error that is better than the error when using a vector of all zeros by a factor of at least θ. We then fix c and find b in the same fashion (line 10). These alternating updates are repeated until convergence conditions are satisfied. We then compare the blocks obtained starting from different candidates and choose the one yielding the best cost.

6 Experiments

We performed synthetic and real-world tests with the proposed algorithm, and compare it with two of the most successful BMF algorithms `Asso` [1] and `PANDA+` [6]. In addition we ran the same experiments with a truncated version of the `NASSAU` algorithm that does not perform updates to the found blocks (that is it stops when adding blocks does not improve the score). This version is denoted by `NASSAUnc` throughout this section.

6.1 Synthetic Data

We evaluated `NASSAU`, as well as its truncated form `NASSAUnc`, on synthetic data and also compared the results to those of `Asso` [1] and `PANDA+` [6] algorithms. `Asso` depends on a user provided parameter to threshold the association matrix. To obtain more accurate results, we ran it with the parameter ranging from 0 to 1 with step 0.05, and then chose the best solution.

Algorithm 1 NASSAU

Input: matrix $A \in \{0,1\}^{n \times m}$, $0 < t < 1$, $0 < \tau < 1$, $0 < \theta < 1$, $M > 0$ – integer
Output: Factors $B \in \{0,1\}^{n \times k}$ and $C \in \{0,1\}^{k \times m}$
function NASSAU(A, t, τ, θ, M)
    Initialize: $B \leftarrow 0^{n \times 0}$, $C \leftarrow 0^{0 \times m}$,
    $Candidates \leftarrow GetCandidates(A)$ ▷ Random walks.
    $[b, c] \leftarrow FindBlock(A, B, C, Candidates, \theta)$
    $Bnew \leftarrow [B, b]$, $Cnew \leftarrow \begin{bmatrix} C \\ c \end{bmatrix}$
    while $L(A, Bnew, Cnew) < L(A, B, C)$ **do**
        $[b, c] \leftarrow FindBlock(A, B, C, Candidates)$
        $B \leftarrow Bnew$, $C \leftarrow Cnew$
        $Bnew \leftarrow [B, b]$, $Cnew \leftarrow \begin{bmatrix} C \\ c \end{bmatrix}$
        if M rounds since last update **then**
            $[Bnew, Cnew] \leftarrow CyclicUpdates(A, Bnew, Cnew, Candidates, 0, \theta)$
    while not converged **do**
        $[B, C] \leftarrow CyclicUpdates(A, B, C, Candidates, t, \theta)$
        $t \leftarrow t * \tau$
    return B, C

Algorithm 2 CyclicUpdates

Input: matrices $A \in \{0,1\}^{n \times m}$, $B \in \{0,1\}^{n \times k}$, $C \in \{0,1\}^{k \times m}$, $Candidates \in \{0,1\}^{n \times s}$, $t > 0$, $0 < \theta < 1$
Output: Factors $Bbest \in \{0,1\}^{n \times k}$ and $Cbest \in \{0,1\}^{k \times m}$
function CYCLICUPDATES(A, B, C, $Candidates$, t, θ)
    $Bbest \leftarrow B$, $Cbest \leftarrow C$
    for $l = 1$ **to** k **do**
        $[b, c] \leftarrow FindBlock(A, B^{-l}, C_{-l}, Candidates, \theta)$
        $Bnew \leftarrow [B^{-l}, b]$, $Cnew \leftarrow \begin{bmatrix} C_{-l} \\ c \end{bmatrix}$ ▷ Replace current block.
        $d = L(A, Bnew, Cnew)$
        if $d < L(A, B, C)$ **then**
            $B \leftarrow Bnew$, $C \leftarrow Cnew$
        else
            $B \leftarrow Bnew$, $C \leftarrow Cnew$ with probability t
        if $d < L(A, Bbest, Cbest)$ **then**
            $Bbest \leftarrow B$, $Cbest \leftarrow C$
    return $Bbest$, $Cbest$

Algorithm 3 FindBlock

Input: matrices $A \in \{0,1\}^{n \times m}$, $B \in \{0,1\}^{n \times k}$, $C \in \{0,1\}^{k \times m}$, $Candidates \in \{0,1\}^{n \times s}$, $0 < \theta < 1$
Output: block $\langle bbest, cbest \rangle$ with $bbest \in \{0,1\}^{n \times 1}$ and $cbest \in \{0,1\}^{1 \times m}$
function FindBlock($A, B, C, Candidates, \theta$)
    $W \leftarrow \mathbf{1} - B \circ C$ ▷ Weight matrix.
    $bbest \leftarrow 0^{n \times 1}$, $cbest \leftarrow 0^{1 \times m}$
    for $i = 1$ **to** s **do**
        $b \leftarrow Candidates^i$ ▷ Initialize b with i-th candidate column.
        repeat
            $c \leftarrow$ row vector with $c_l = 1$ iff $\sum_{W_{jl}=1} |A_{jl} - b_j)| < \theta \sum_{W_{jl}=1} A_{jl}$
            $b \leftarrow$ column vector with $b_l = 1$ iff $\sum_{W_{lj}=1} |A_{lj} - c^j| < \theta \sum_{W_{lj}=1} A_{lj}$
        until stopping criteria are satisfied
        if $L(A, [B, b], \left[\begin{smallmatrix} C \\ c \end{smallmatrix}\right]) < L(A, [B, bbest], \left[\begin{smallmatrix} C \\ cbest \end{smallmatrix}\right])$ **then**
            $bbest \leftarrow b$, $cbest \leftarrow c$
    return $bbest$, $cbest$

The experimental setup was as follows. We first generated random binary factors, then multiplied them using the Boolean matrix product to obtain matrices of size 600 by 400 with inner dimension of 15 and density 0.08. Then after adding some noise, we ran the algorithms on the noisy matrices and measured the obtained description length.

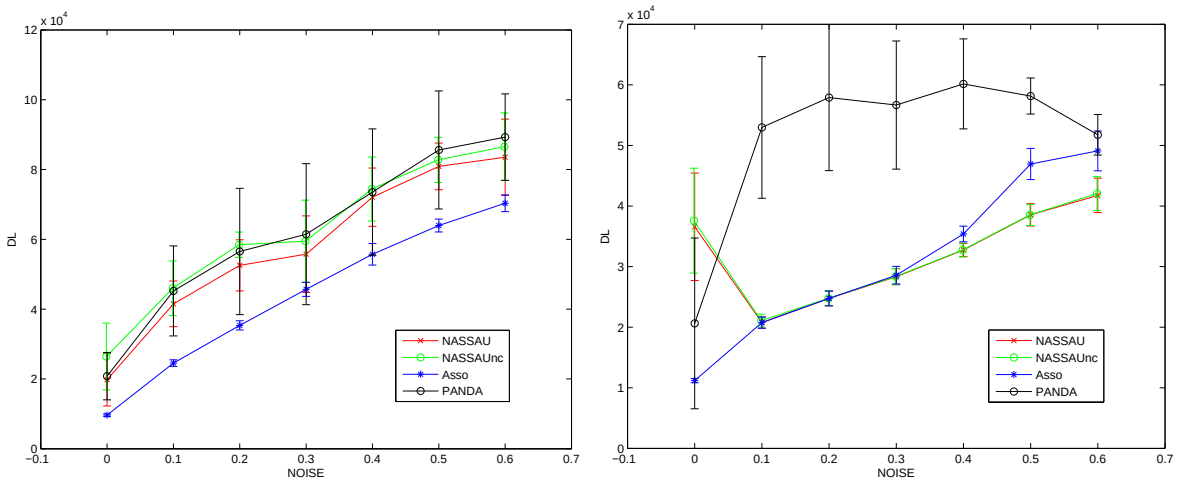
Additive noise. The purpose of this test is to find how robust the algorithms are to additive noise, that is when 0s are turned to 1s. We used various noise levels, ranging from 0 to 60% with respect to the number of ones in the input matrix. We also added a very low level (3%) of destructive noise, which was kept constant throughout the test. The results are presented in Figure 1a.

Destructive noise. The varying destructive noise test has the identical setup to the above , except that we now turn 1s to 0s. The level of noise is again measured relative to the number of ones in the input data. Analogously to the previous test, we added 3% of additive noise. The results are shown in Figure 1b.

From the plots it is obvious that `Asso` is more robust to additive noise than other algorithms. On the other hand, `NASSAU` outperforms all other methods for high levels of destructive noise. `PANDA+` performs slightly worse than `NASSAU` for the additive noise test. However, high levels of subtractive noise deteriorates its results relatively quickly compared to other methods. Comparing performances of `NASSAU` and its `NASSAUnc`, we see that while updates were clearly useful for additive noise, in case of destructive noise the improvement was very small.

6.2 Real-World Data

We performed experiments on real-world datasets and compared the results of `NASSAU`, `NASSAUnc`, `Asso`, and `PANDA+` algorithms. We tested all the algorithms



(a) Varying additive noise.

(b) Varying destructive noise.

Fig. 1. Varying noise test. The vertical axis represents the obtained description length and the horizontal axis stands for the level of noise. Markers represent the means over 10 instances and the error bars have width of twice the standard deviation.

on the following datasets: Paleo – fossil records[1], Dialects – presence data of dialects across Finnish municipalities [9], [10], Newsgroups – an excerpt from the 20Newsgroups dataset[2] containing news posts in a bag-of-words representation, and Mammals – a presence data of different mammals within geographical areas of 50×50 kilometers in Europe [8]. For `Asso`, same as in the synthetic experiments, we chose the best scoring thresholding parameter. The obtained results are collected in Table 1. It can be seen that `NASSAU` obtains lower costs than the other two methods for all the datasets tested. A possible explanation for that is that it is less vulnerable to destructive noise, which is very likely to occur in many of the real-world datasets. For example in the Mammals dataset many of 0s could actually represent the fact that some species have not been observed in some area due to, for instance low population, rather than the absence of this species. In other words, in this case the data contains many false negatives, and `NASSAU` is quite robust against them. Also for two out of four datasets updates run by `NASSAU` gave it a substantial edge over its truncated form.

7 Conclusions

In this work we introduced a new algorithm for the BMF problem that directly optimizes the description length of the data. The algorithm we propose is nonhierarchical and is capable of fixing errors it has previously made. Based on experiments with both real-world and synthetically generated data, we can

[1] NOW public release 030717, available at http://www.helsinki.fi/science/now/ [Fortelius et al. 2003].

[2] http://people.csail.mit.edu/jrennie/20Newsgroups/

Table 1. Comparison between the description length obtained for various real-world datasets by `Asso`, `PANDA+`, `NASSAU`, and `NASSAUnc` algorithms.

Algorihtm	Paleo	Dialects	Newsgroups	Mammals
Asso	18556	209713	65955	215209
PANDA+	19728	292120	67120	234710
NASSAU	17831	176017	65680	179939
NASSAUnc	17931	230120	66198	196970

conclude that is is competitive with the best existing BMF methods. In particular, it is very robust to high levels of destructive noise (more so than all other methods that we tested). Moreover, it obtained better results on all real-world experiments that we conducted.

References

1. Miettinen, P.: Matrix decomposition methods for data mining: Computational complexity and algorithms. Ph.D. thesis, University of Helsinki (2009)
2. Golub, G. H. and Van Loan, C. F.: Matrix computations. JHU Press. (1996)
3. Lee D. D. and Seung, H. S.: Learning the parts of objects by non-negative matrix factorization. Nature, 401:788-791. (1999)
4. Rissanen, J.: Modeling by shortest data description. Automatica 14, 1, 465-471. (1978)
5. Miettinen, P. and Vreeken, J.: Model order selection for Boolean matrix factorization. Proceedings of the 17th ACM SIGKDD international conference on Knowledge Discovery and Data Mining. (2011)
6. Lucchese, C., Orlando, S., and Perego R.: A unifying framework for mining approximate top-k binary patterns. IEEE Transactions on Knowledge and Data Engineering. (2013), to appear
7. Lucchese, C., Orlando, S., and Perego, R.,: Mining top-k patterns from binary datasets in presence of noise. SDM, SIAM, pp. 165-176. (2010)
8. Mitchell-Jones, A., Amori, G., Bogdanowicz, W., Krystufek, B., Reijnders, P. H., Spitzenberger, F., Stubbe, M., Thissen, J., Vohralik, V., and Zima, J.: The atlas of european mammals. Academic Press. (1999)
9. Embleton, S. M. and Wheeler, E. S.: Finnish dialect atlas for quantitative studies. Journal of Quantitative Linguistics 4, 1-3, 99-102. (1997)
10. Embleton, S. M. and Wheeler, E. S.: Computerized dialect atlas of finnish: dealing with ambiguity. Journal of Quantitative Linguistics 7, 3, 227-231. (2000)
11. Cover, T. M. and Thomas, J. A.: Elements of information theory. Wiley-Interscience, New York. (2006)
12. Vereshchagin, N. and Vitanyi, P.: Kolmogorov's structure functions and model selection. IEEE Transactions on Information Technology 50, 12, 3265-3290. (2004)
13. Geerts, F., Goethals, B., and Mielikinen, T.: Tiling databases. Discovery Science. Springer Berlin Heidelberg. (2004)
14. Grünwald, P.: A tutorial introduction to the minimum description length principle. MIT Press. (2005)

Author Index