4th Workshop on

Machine Learning in Life Sciences

23 September 2016
Riva del Garda, Italy

Organized under
European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2016

ENGINE Center, Wroclaw University of Science and Technology
Wrocław, Poland, 2016

This work was supported by EC under FP7, Coordination and Support Action, Grant Agreement Number 316097, ENGINE – European Research Centre of Network Intelligence for Innovation Enhancement.

Editorial layout and cover design Pawel Ksieniewicz

European Research Centre of Network Intelligence for Innovation Enhancement
Wroclaw University of Science and Technology
Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland

ISBN 978-83-943803-1-1

Preface

Life sciences, ranging from medicine, biology and genetics to biochemistry and pharmacology have developed rapidly in previous years. Computerization of those domains allowed to gather and store enormous collections of data. Analysis of such vast amounts of information without any support is impossible for human being. Therefore recently machine learning and pattern recognition methods have attracted the attention of broad spectrum of experts from life sciences domain.

The aim of this Workshop is to stress the importance of interdisciplinary collaboration between life and computer sciences and to provide an international forum for both practitioners seeking new cutting-edge tools for solving their domain problems and theoreticians seeking interesting and real-life applications for their novel algorithms. We are interested in novel machine learning technologies, designed to tackle complex medical, biological, chemical or environmental data that take into consideration the specific background knowledge and interactions between the considered problems. We look for novel applications of machine learning and pattern recognition tools to contemporary life sciences problems, that will shed light on their strengths and weaknesses. We are interested in new methods for data visualization and methods for accessible presentation of results of machine learning analysis to life scientists. We welcome new findings in the intelligent processing of non-stationary medical, biological and chemical data and in proposals for efficient fusion of information coming from multiple sources. Papers on efficient analysis and classification of bid data (understood as both massive volumes and high-dimensionality problems) will be of special interest to this Workshop.

September 2016

Michal Wozniak
Bartosz Krawczyk
Workshop Chair
MLLS 2016

Organization

Machine Learning in Life Sciences 2016 is organized under European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2016.

Workshop Chairs

Bartosz Krawczyk	Wroclaw University of Science and Technology, PL
Michal Wozniak	Wroclaw University of Science and Technology, PL
Krzysztof Cios	Virginia Commonwealth University, USA
Igor Podolak	Jagiellonian University, PL
Tomislav Smuc	Rudjer Boskovic Institute, HR

Program Commitee

Boguslaw Cyganek	AGH University of Science and Technology, PL
Wojciech Czarnecki	Jagiellonian University, PL
Saso Dzeroski	Jozef Stefan Institute, SI
Mikel Galar	Public University of Navarra, ES
Dragan Gamberger	Rudjer Boskovic Institute, HR
Manuel Grana	University of the Basque Country, ES
Dragi Kocev	Jozef Stefan Institute, SI
Lukasz Kurgan	Virginia Commonwealth University, USA
Katarzyna Wegrzyn-Wolska	ESIGETEL, FR
Mohamed Medhat Gaber	Robert Gordon University, UK
Jose Antonio Saez	University of Granada, ES
Fran Supek	Centre for Genomic Regulation, ES
Jacek Tabor	Jagiellonian University, PL
Isaac Triguero	Ghent University, BE
Pawel Ksieniewicz	Wroclaw University of Science and Technology, PL

Table of Contents

An Efficient Training Algorithm for Kernel Survival Support Vector Machines

Sebastian Pölsterl[1](✉), Nassir Navab[2,3], and Amin Katouzian[4]

[1] The Knowledge Hub Team, The Institute of Cancer Research, London, UK,
[2] Chair for Computer Aided Medical Procedures
Technische Universität München, Munich, Germany
[3] Johns Hopkins University, Baltimore MD, USA
[4] IBM Almaden Research Center, San Jose CA, USA
sebastian.poelsterl@icr.ac.uk, nassir.navab@tum.de, akatouz@us.ibm.com

Abstract. Survival analysis is a fundamental tool in medical research to identify predictors of adverse events and develop systems for clinical decision support. In order to leverage large amounts of patient data, efficient optimisation routines are paramount. We propose an efficient training algorithm for the kernel survival support vector machine (SSVM). We directly optimise the primal objective functionand employ truncated Newton optimisation and order statistic trees to significantly lower computational costs compared to previous training algorithms, which require $O(n^4)$ space and $O(pn^6)$ time for datasets with n samples and p features. Our results demonstrate that our proposed optimisation scheme allows analysing data of a much larger scale with no loss in prediction performance. Experiments on synthetic and 5 real-world datasets show that our technique outperforms existing kernel SSVM formulations if the amount of right censoring is high ($\geq 85\%$), and performs comparably otherwise.

Keywords: survival analysis · support vector machine · optimisation · kernel-based learning

1 Introduction

In clinical research, the primary interest is often the time until occurrence of an adverse event, such as death or reaching a specific state of disease progression. In *survival analysis*, the objective is to establish a connection between a set of features and the time until an event of interest. It differs from traditional machine learning, because parts of the training data can only be partially observed. In a clinical study, patients are often monitored for a particular time period, and events occurring in this particular period are recorded. If a patient experiences an event, the exact time of the event can be recorded – the time of the event is *uncensored*. In contrast, if a patient remained event-free during the study period, it is unknown whether an event has or has not occurred after the study ended – the time of an event is *right censored* at the end of the study.

Cox's proportional hazards model (CoxPH) is the standard for analysing time-to-event data. However, its decision function is linear in the covariates,

which can lead to poor predictive performance if non-linearities and interactions are not modelled explicitly. Depending on the level of complexity, researchers might be forced to try many different model formulations, which is cumbersome. The success of kernel methods in machine learning has motivated researchers to propose kernel-based survival models, which ease analysis in the presence of non-linearities, and allow analysing complex data in the form of graphs or strings by means of appropriate kernel functions (e.g. [23, 35]). Thus, instead of merely describing patients by feature vectors, structured and more expressive forms of representation can be employed, such as gene co-expression networks [36].

A kernel-based CoxPH model was proposed in [24, 2]. Authors in [19, 28] cast survival analysis as a regression problem and adapted support vector regression, whereas authors in [11, 31] cast it as a learning-to-rank problem by adapting the rank support vector machine (Rank SVM). Eleuteri *et al.* [10] formulated a model based on quantile regression. A transformation model with minimal Lipschitz smoothness for survival analysis was proposed in [33]. Finally, Van Belle *et al.* [34] proposed a hybrid ranking-regression model.

In this paper, we focus on improving the optimisation scheme of the non-linear ranking-based survival support vector machine (SSVM). Existing training algorithms [11, 31] perform optimisation in the dual and require $O(pn^6)$ time – excluding evaluations of the kernel function – and $O(n^4)$ space, where p and n are the number of features and samples. Recently, an efficient training algorithm for linear SSVM with much lower time complexity and linear space complexity has been proposed [26]. We extend this optimisation scheme to the non-linear case and demonstrate its superiority on synthetic and real-world datasets. Our implementation of the proposed training algorithm is available online at https://github.com/tum-camp/survival-support-vector-machine.

2 Methods

Given a dataset $\mathcal{D}$ of n samples, let $\mathbf{x}_i$ denote a p-dimensional feature vector, $t_i > 0$ the time of an event, and $c_i > 0$ the time of censoring of the i-th sample. Due to right censoring, it is only possible to observe $y_i = \min(t_i, c_i)$ and $\delta_i = I(t_i \leq c_i)$ for every sample, with $I(\cdot)$ being the indicator function and $c_i = \infty$ for uncensored records. Hence, training a survival model is based on a set of triplets: $\mathcal{D} = \{(\mathbf{x}_i, y_i, \delta_i)\}_{i=1}^n$. After training, a survival model ought to predict a risk score of experiencing an event based on a set of given features.

2.1 The Survival Support Vector Machine

The SSVM is an extension of the Rank SVM [13] to right censored survival data [31, 11]. Consequently, survival analysis is cast as a learning-to-rank problem: patients with a lower survival time should be ranked before patients with longer survival time. In the absence of censoring – as it is the case for traditional Rank SVM – all pairwise comparisons of samples are used during training. However, if samples are right censored, some pairwise relationships are invalid. When

comparing two censored samples i and j ($\delta_i = \delta_j = 0$), it is unknown whether the i-th sample should be ranked before the j-th sample or vice versa, because the exact time of an event is unknown for both samples. The same applies if comparing an uncensored sample i with a censored sample j ($\delta_i = 1$ and $\delta_j = 0$) if $y_i < y_j$. Therefore, a pairwise comparison (i, j) is only valid if the sample with the lower observed time is uncensored. Formally, the set of valid comparable pairs $\mathcal{P}$ is given by

$$\mathcal{P} = \{(i, j) \mid y_i > y_j \wedge \delta_j = 1\}_{i,j=1}^{n},$$

where it is assumed that all observed time points are unique [31, 11]. Training a linear SSVM (based on the hinge loss) requires solving the following optimisation problem:

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \gamma \sum_{(i,j)\in\mathcal{P}} \max(0, 1 - \boldsymbol{w}^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)), \tag{1}$$

where $\boldsymbol{w} \in \mathbb{R}^p$ are the model's coefficients and $\gamma > 0$ controls the degree of regularisation.

Without censoring, all possible pairs of samples have to be considered during training, hence $|\mathcal{P}| = n(n-1)/2$ and the sum in (1) consists of a quadratic number of addends with respect to the number of training samples. If part of the survival times are censored, the size of $\mathcal{P}$ depends on the amount of uncensored records and the order of observed time points – censored and uncensored. Let q_e denote the percentage of uncensored time points, then $|\mathcal{P}|$ is at least $q_e n(q_e n - 1)/2$. This situation arises if all censored subjects drop out before the first event was observed, hence, all uncensored records are incomparable to all censored records. If the situation is reversed and the first censored time point occurs after the last time point of an observed event, all uncensored records can be compared to all censored records, which means $|\mathcal{P}| = q_e n^2 - q_e n(q_e n + 1)/2$. In both cases, $|\mathcal{P}|$ is of the order of $O(q_e n^2)$ and the number of addends in optimisation problem (1) is quadratic in the number of samples. In [31, 11], the objective function (1) is minimised by solving the corresponding Lagrange dual problem:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \boldsymbol{\alpha}^\top \mathbb{1}_m - \frac{1}{2}\boldsymbol{\alpha}^\top \boldsymbol{A}\boldsymbol{X}\boldsymbol{X}^\top \boldsymbol{A}^\top \boldsymbol{\alpha} \\ \text{subject to} \quad & 0 \leq \alpha_{ij} \leq \gamma, \quad \forall (i, j) \in \mathcal{P}, \end{aligned} \tag{2}$$

where $m = |\mathcal{P}|$, $\mathbb{1}_m$ is a m-dimensional vector of all ones, $\boldsymbol{\alpha} \in \mathbb{R}^m$ are the Lagrangian multipliers, and $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is a sparse matrix with $\boldsymbol{A}_{k,i} = 1$ and $\boldsymbol{A}_{k,j} = -1$ if $(i, j) \in \mathcal{P}$ and zero otherwise. It is easy to see that this approach quickly becomes intractable, because constructing the matrix $\boldsymbol{A}\boldsymbol{X}\boldsymbol{X}^\top\boldsymbol{A}^\top$ requires $O(n^4)$ space and solving the quadratic problem (2) requires $O(pn^6)$ time.

Van Belle *et al.* [32] addressed this problem by reducing the size of $\mathcal{P}$ to $O(n)$ elements: they only considered pairs (i, j), where j is the largest uncensored sample with $y_i > y_j$. However, this approach effectively simplifies the objective function (1) and usually leads to a different solution. In [26], we proposed an efficient optimisation scheme for solving (1) by substituting the hinge loss for the

squared hinge loss, and using truncated Newton optimisation and order statistics trees to avoid explicitly constructing all pairwise comparisons, resulting in a much reduced time and space complexity. However, the optimisation scheme is only applicable to training a linear model. To circumvent this problem, the data can be transformed via Kernel PCA [27] before training, which effectively results in a non-linear model in the original feature space [26, 4]. The disadvantage of this approach is that it requires $O(n^2 p)$ operations to construct the kernel matrix – assuming evaluating the kernel function costs $O(p)$ – and $O(n^3)$ to perform singular value decomposition. Kuo *et al.* [21] proposed an alternative approach for Rank SVM that allows directly optimising the primal objective function in the non-linear case too. It is natural to adapt this approach for training a non-linear SSVM, which we will describe next.

2.2 Efficient Training of a Kernel Survival Support Vector Machine

The main idea to obtain a non-linear decision function is that the objective function (1) is reformulated with respect to finding a function $f\colon \mathcal{X} \to \mathbb{R}$ from a reproducing kernel Hilbert space $\mathcal{H}_k$ with associated kernel function $k\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that maps the input $\boldsymbol{z} \in \mathcal{X}$ to a real value (usually $\mathcal{X} \subset \mathbb{R}^p$):

$$\min_{f \in \mathcal{H}_k} \quad \frac{1}{2}\|f\|^2_{\mathcal{H}_k} + \frac{\gamma}{2} \sum_{(i,j) \in \mathcal{P}} \max(0, 1 - (f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)))^2,$$

where we substituted the hinge loss for the squared hinge loss, because the latter is differentiable. Using the representer theorem [20, 3, 21], the function f can be expressed as $f(\boldsymbol{z}) = \sum_{i=1}^{n} \beta_i k(\boldsymbol{x}_i, \boldsymbol{z})$, which results in the objective function

$$R(\boldsymbol{\beta}) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \beta_i \beta_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \frac{\gamma}{2} \sum_{(i,j) \in \mathcal{P}} \max\left(0, 1 - \sum_{l=1}^{n} \beta_l (k(\boldsymbol{x}_l, \boldsymbol{x}_i) - k(\boldsymbol{x}_l, \boldsymbol{x}_j))\right)^2,$$

where the norm $\|f\|^2_{\mathcal{H}_k}$ can be computed by using the reproducing kernel property $f(\boldsymbol{z}) = \langle f, k(\boldsymbol{z}, \cdot)\rangle$ and $\langle k(\boldsymbol{z}, \cdot), k(\boldsymbol{z}', \cdot)\rangle = k(\boldsymbol{z}, \boldsymbol{z}')$. The objective function can be expressed in matrix form through the $n \times n$ symmetric positive definite kernel matrix $\boldsymbol{K}$ with entries $\boldsymbol{K}_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$:

$$R(\boldsymbol{\beta}) = \frac{1}{2} \boldsymbol{\beta}^\top \boldsymbol{K} \boldsymbol{\beta} + \frac{\gamma}{2} \left(\mathbb{1}_m - \boldsymbol{A}\boldsymbol{K}\boldsymbol{\beta}\right)^\top \boldsymbol{D}_{\boldsymbol{\beta}} \left(\mathbb{1}_m - \boldsymbol{A}\boldsymbol{K}\boldsymbol{\beta}\right), \tag{3}$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n)^\top$ are the coefficients and $\boldsymbol{D}_{\boldsymbol{\beta}}$ is a $m \times m$ diagonal matrix that has an entry for each $(i, j) \in \mathcal{P}$ that indicates whether this pair is a support pair, i.e., $1 - (f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)) > 0$. For the k-th item of $\mathcal{P}$, representing the pair (i, j), the corresponding entry in $\boldsymbol{D}_{\boldsymbol{\beta}}$ is defined as

$$(\boldsymbol{D}_{\boldsymbol{\beta}})_{k,k} = \begin{cases} 1 & \text{if } f(\boldsymbol{x}_j) > f(\boldsymbol{x}_i) - 1 \Leftrightarrow \boldsymbol{K}_j \boldsymbol{\beta} > \boldsymbol{K}_i \boldsymbol{\beta} - 1, \\ 0 & \text{else,} \end{cases}$$

where $\boldsymbol{K}_i$ denotes the i-th row of kernel matrix $\boldsymbol{K}$. Note that in contrast to the Lagrangian multipliers $\boldsymbol{\alpha}$ in (2), $\boldsymbol{\beta}$ in (3) is unconstrained and usually dense.

The objective function (3) of the non-linear SSVM is similar to the linear model discussed in [26]; in fact, $R(\boldsymbol{\beta})$ is differentiable and convex with respect to $\boldsymbol{\beta}$, which allows employing truncated Newton optimisation [7]. The first- and second-order partial derivatives have the form

$$\frac{\partial R(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \boldsymbol{K}\boldsymbol{\beta} + \gamma \boldsymbol{K}^\top \left(\boldsymbol{A}^\top \boldsymbol{D}_{\boldsymbol{\beta}} \boldsymbol{A} \boldsymbol{K} \boldsymbol{\beta} - \boldsymbol{A}^\top \boldsymbol{D}_{\boldsymbol{\beta}} \mathbb{1}_m \right), \tag{4}$$

$$\frac{\partial^2 R(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} = \boldsymbol{K} + \gamma \boldsymbol{K}^\top \boldsymbol{A}^\top \boldsymbol{D}_{\boldsymbol{\beta}} \boldsymbol{A} \boldsymbol{K}, \tag{5}$$

where the generalised Hessian is used in the second derivative, because $R(\boldsymbol{\beta})$ is not twice differentiable at $\boldsymbol{\beta}$ [18].

Note that the expression $\boldsymbol{A}^\top \boldsymbol{D}_{\boldsymbol{\beta}} \boldsymbol{A}$ appears in eqs. (3) to (5). Right multiplying $\boldsymbol{A}^\top$ by the diagonal matrix $\boldsymbol{D}_{\boldsymbol{\beta}}$ has the effect that rows not corresponding to support pairs – pairs $(i, j) \in \mathcal{P}$ for which $1 - (\boldsymbol{K}_i \boldsymbol{\beta} - \boldsymbol{K}_j \boldsymbol{\beta}) < 0$ – are dropped from the matrix $\boldsymbol{A}$. Thus, $\boldsymbol{A}^\top \boldsymbol{D}_{\boldsymbol{\beta}} \boldsymbol{A}$ can be simplified by expressing it in terms of a new matrix $\boldsymbol{A}_{\boldsymbol{\beta}} \in \{-1, 0, 1\}^{m_{\boldsymbol{\beta}}, n}$ as $\boldsymbol{A}^\top \boldsymbol{D}_{\boldsymbol{\beta}} \boldsymbol{A} = \boldsymbol{A}_{\boldsymbol{\beta}}^\top \boldsymbol{A}_{\boldsymbol{\beta}}$, where $m_{\boldsymbol{\beta}}$ denotes the number of support pairs in $\mathcal{P}$. Thus, the objective function and its derivatives can be compactly expressed as

$$\begin{aligned}
R(\boldsymbol{\beta}) &= \frac{1}{2} \boldsymbol{\beta}^\top \boldsymbol{K} \boldsymbol{\beta} + \frac{\gamma}{2} \left(m_{\boldsymbol{\beta}} + \boldsymbol{\beta}^\top \boldsymbol{K} \left(\boldsymbol{A}_{\boldsymbol{\beta}}^\top \boldsymbol{A}_{\boldsymbol{\beta}} \boldsymbol{K} \boldsymbol{\beta} - 2 \boldsymbol{A}_{\boldsymbol{\beta}}^\top \mathbb{1}_{m_{\boldsymbol{\beta}}} \right) \right), \\
\frac{\partial R(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= \boldsymbol{K} \boldsymbol{\beta} + \gamma \boldsymbol{K} \left(\boldsymbol{A}_{\boldsymbol{\beta}}^\top \boldsymbol{A}_{\boldsymbol{\beta}} \boldsymbol{K} \boldsymbol{\beta} - \boldsymbol{A}_{\boldsymbol{\beta}} \mathbb{1}_{m_{\boldsymbol{\beta}}} \right), \\
\frac{\partial^2 R(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} &= \boldsymbol{K} + \gamma \boldsymbol{K} \boldsymbol{A}_{\boldsymbol{\beta}}^\top \boldsymbol{A}_{\boldsymbol{\beta}} \boldsymbol{K}.
\end{aligned}$$

The gradient and Hessian of the non-linear SSVM share properties with the corresponding functions of the linear model. Therefore, we can adapt the efficient training algorithm for linear SSVM [26] with only small modifications, thereby avoiding explicitly constructing the matrix $\boldsymbol{A}_{\boldsymbol{\beta}}$, which would require $O(q_e n^2)$ space. Since the derivation for the non-linear case is very similar to the linear case, we only present the final result here and refer to [26] for details.

In each iteration of truncated Newton optimisation, a Hessian-vector product needs to be computed. The second term in this product involves $\boldsymbol{A}_{\boldsymbol{\beta}}$ and becomes

$$\gamma \boldsymbol{K} \boldsymbol{A}_{\boldsymbol{\beta}}^\top \boldsymbol{A}_{\boldsymbol{\beta}} \boldsymbol{K} \boldsymbol{v} = \gamma \boldsymbol{K} \begin{pmatrix} (l_1^+ + l_1^-) \boldsymbol{K}_1 \boldsymbol{v} - (\sigma_1^+ + \sigma_1^-) \\ \vdots \\ (l_n^+ + l_n^-) \boldsymbol{K}_n \boldsymbol{v} - (\sigma_n^+ + \sigma_n^-) \end{pmatrix}, \tag{6}$$

where, in analogy to the linear SSVM,

$$\begin{aligned}
\mathrm{SV}_i^+ &= \{ s \mid y_s > y_i \wedge \boldsymbol{K}_s \boldsymbol{\beta} < \boldsymbol{K}_i \boldsymbol{\beta} + 1 \wedge \delta_i = 1 \}, & l_i^+ &= |\mathrm{SV}_i^+|, \\
\mathrm{SV}_i^- &= \{ s \mid y_s < y_i \wedge \boldsymbol{K}_s \boldsymbol{\beta} > \boldsymbol{K}_i \boldsymbol{\beta} - 1 \wedge \delta_s = 1 \}, & l_i^- &= |\mathrm{SV}_i^-|,
\end{aligned}$$

$\sigma_i^+ = \sum_{s \in \mathrm{SV}_i^+} \boldsymbol{K}_s \boldsymbol{v}$, and $\sigma_i^- = \sum_{s \in \mathrm{SV}_i^-} \boldsymbol{K}_s \boldsymbol{v}$. The values l_i^+, l_i^-, σ_i^+, and σ_i^- can be obtained in logarithmic time by first sorting according to the predicted scores $f(\boldsymbol{x}_i) = \boldsymbol{K}_i \boldsymbol{\beta}$ and subsequently incrementally constructing one order statistic tree to hold SV_i^+ and SV_i^-, respectively [26, 22, 21]. Finally, the risk score of experiencing an event for a new data point $\boldsymbol{x}_{\mathrm{new}}$ can be estimated by $\hat{f}(\boldsymbol{x}_{\mathrm{new}}) = \sum_{i=1}^{n} \hat{\beta}_i k(\boldsymbol{x}_i, \boldsymbol{x}_{\mathrm{new}})$, where $\hat{\boldsymbol{\beta}} = \operatorname{argmin} R(\boldsymbol{\beta})$.

2.3 Complexity

The overall complexity of the training algorithm for a linear SSVM is

$$[O(n \log n) + O(np + p + n \log n)] \cdot \bar{N}_{\mathrm{CG}} \cdot N_{\mathrm{Newton}},$$

where $\bar{N}_{\mathrm{CG}}$ and N_{Newton} are the average number of conjugate gradient iterations and the total number of Newton updates, respectively [26]. For the non-linear model, we first have to construct the $n \times n$ kernel matrix $\boldsymbol{K}$. If $\boldsymbol{K}$ cannot be stored in memory, computing the product $\boldsymbol{K}_i \boldsymbol{v}$ requires n evaluations of the kernel function and n operations to compute the product. If evaluating the kernel function costs $O(p)$, the overall complexity is $O(n^2 p)$. Thus, computing the Hessian-vector product in the non-linear case consists of three steps, which have the following complexities:

1. $O(n^3 p)$ to compute $\boldsymbol{K}_i \boldsymbol{v}$ for all $i = 1, \ldots, n$,
2. $O(n \log n)$ to sort samples according to values of $\boldsymbol{K}_i \boldsymbol{v}$,
3. $O(n^2 + n + n \log n)$ to calculate the Hessian-vector product via (6).

This clearly shows that, in contrast to training a linear model, computing the sum over all comparable pairs is no longer the most time consuming task in minimising $R(\boldsymbol{\beta})$ in eq. (3). Instead, computing $\boldsymbol{K}\boldsymbol{v}$ is the dominating factor.

If the number of samples in the training data is small, the kernel matrix can be computed once and stored in memory thereafter, which results in a one-time cost of $O(n^2 p)$. It reduces the costs to compute $\boldsymbol{K}\boldsymbol{v}$ to $O(n^2)$ and the remaining costs remain the same. Although pre-computing the kernel matrix is an improvement, computing $\boldsymbol{K}\boldsymbol{v}$ in each conjugate gradient iteration remains the bottleneck. The overall complexity of training a non-linear SSVM with truncated Newton optimisation and order statistics trees is

$$O(n^2 p) + \left[O(n \log n) + O(n^2 + n + n \log n)\right] \cdot \bar{N}_{\mathrm{CG}} \cdot N_{\mathrm{Newton}}. \tag{7}$$

Note that direct optimisation of the non-linear objective function is preferred over using Kernel PCA to transform the data before training, because it avoids $O(n^3)$ operations corresponding to the singular value decomposition of $\boldsymbol{K}$.

3 Comparison of Survival Support Vector Machines

3.1 Datasets

Synthetic Data Synthetic survival data of varying size was generated following [1]. Each dataset consisted of one uniformly distributed feature in the interval

Table 1. Overview of datasets used in our experiments.

Dataset	n	p	Events	Outcome
AIDS study [14]	1,151	11	96 (8.3%)	AIDS defining event
Coronary artery disease [25]	1,106	56	149 (13.5%)	Myocardial infarction or death
Framingham offspring [17]	4,892	150	1,166 (23.8%)	Coronary vessel disease
Veteran's Lung Cancer [16]	137	6	128 (93.4%)	Death
Worcester Heart Attack Study [14]	500	14	215 (43.0%)	Death

[18; 89], denoting age, one binary variable denoting sex, drawn from a binomial distribution with probability 0.5, and a categorical variable with 3 equally distributed levels. In addition, 10 numeric features were sampled from a multivariate normal distribution $\mathcal{N}_{10}(\boldsymbol{\mu}, \boldsymbol{I})$ with mean $\boldsymbol{\mu} = (0,\ 0,\ 0.3,\ 0.15,\ 0.8,\ 0.67,\ 0.2,\ 0,\ 0.12,\ 0.3)^\top$. Survival times t_i were drawn from a Weibull distribution with $k = 1$ (constant hazard rate) and $\lambda = 0.9$ according to the formula presented in [1]: $t_i = [(-\log u_i)/(\lambda \exp(f(\boldsymbol{x}_i)))]^{1/k}$, where u_i is uniformly distributed within $[0; 1]$, $f(\cdot)$ denotes a non-linear model that relates the features to the survival time (see below), and $\boldsymbol{x}_i \in \mathbb{R}^{14}$ is the feature vector of the i-th subject. The censoring time c_i was drawn from a uniform distribution in the interval $[0; \tau]$, where τ was chosen such that about 20% of survival times were censored in the training data. Survival times in the test data were not subject to censoring to eliminate the effect of censoring on performance estimation. The non-linear model $f(\boldsymbol{x})$ was defined as

$$f(\boldsymbol{x}) = 0.05x_{\text{age}} + 0.8x_{\text{sex}} + 0.03x_{\text{N1}}^2 + 0.3x_{\text{N2}}^{-2} - 0.1x_{\text{N7}} + 0.6x_{\text{N4}}/x_{\text{N2}} + x_{\text{N1}}/x_{\text{N8}} - 0.9\tanh(x_{\text{N6}})/x_{\text{N9}} + 0.09x_{\text{C1}}/x_{\text{sex}} + 0.03x_{\text{C2}}/x_{\text{sex}} + 0.3x_{\text{C3}}/x_{\text{sex}},$$

where C1, C2, and C3 correspond to dummy codes of a categorical feature with three categories and N1 to N10 to continuous features sampled from a multivariate normal distribution. Note that the 3^{rd}, 9^{th} and 10^{th} numeric feature are associated with a zero coefficient, thus do not affect the survival time. We generated 100 pairs of train and test data of 1,500 samples each by multiplying the coefficients by a random scaling factor uniformly drawn from $[-1; 1]$.

Real Data In the second set of experiments, we focused on 5 real-world datasets of varying size, number of features, and amount of censoring (see table 1). The Framingham offspring and the coronary artery disease data contained missing values, which were imputed using multivariate imputation using chained equations with random forest models [9]. To ease computational resources for validation and since the missing values problem was not the focus, one multiple imputed dataset was randomly picked and analysed. Finally, we normalised continuous variables to have zero mean and unit variance.

3.2 Prediction Performance

Experiments presented in this section focus on the comparison of the predictive performance of the survival SVM on 100 synthetically generated datasets as well as 5 real-world data sets against 3 alternative survival models:

1. Simple SSVM with hinge loss and $\mathcal{P}$ restricted to pairs (i, j), where j is the largest uncensored sample with $y_i > y_j$ [32],
2. Minlip survival model by Van Belle *et al.* [33],
3. linear SSVM based on the efficient training scheme proposed in [26],
4. Cox's proportional hazards model [5] with ℓ_2 penalty.

The regularisation parameter γ for SSVM and Minlip controls the weight of the (squared) hinge loss, whereas for Cox's proportional hazards model, it controls the weight of the ℓ_2 penalty. Optimal hyper-parameters were determined via grid search by evaluating each configuration using ten random 80%/20% splits of the training data. The parameters that on average performed best across these ten partitions were ultimately selected and the model was re-trained on the full training data using optimal hyper-parameters. We used Harrell's concordance index (c-index) [12] – a generalisation of Kendall's τ to right censored survival data – to estimate the performance of a particular hyper-parameter configuration. In the grid search, γ was chosen from the set $\{2^{-12}, 2^{-10}, \ldots, 2^{12}\}$. The maximum number of iterations of Newton's method was 200.

In our cross-validation experiments on real-world data, the test data was subject to censoring too, hence performance was measured by Harrell's and Uno's c-index [12, 29] and the integrated area under the time-dependent, cumulative-dynamic ROC curve (iAUC; [30, 15]). The latter was evaluated at time points corresponding to the $10\%, 20\%, \ldots, 80\%$ percentile of the observed time points in the complete dataset. For Uno's c-index the truncation time was the 80% percentile of the observed time points in the complete dataset. For all three measures, the values 0 and 1 indicate a completely wrong and perfectly correct prediction, respectively. Finally, we used Friedman's test and the Nemenyi post-hoc test to determine whether the performance difference between any two methods is statistically significant at the 0.05 level [8].

Synthetic Data The first set of experiments on synthetic data served as a reference on how kernel-based survival models compare to each other in a controlled setup. We performed experiments using an RBF kernel and the clinical kernel [6]. Figure 1 summarises the results on 100 synthetically generated datasets, where all survival times in the test data were uncensored, which leads to unbiased and consistent estimates of the c-index. The experiments revealed that using a clinical kernel was advantageous in all experiments (see fig. 1). Using the clinical kernel in combination with any of the SSVM models resulted in a significant improvement over the corresponding model with RBF kernel and linear model, respectively. Regarding the RBF kernel, it improved the performance over a linear model, except for the simple SSVM, which did not perform significantly

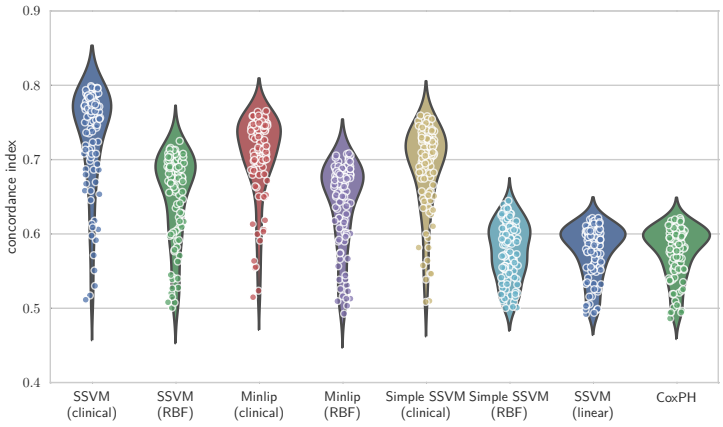


Fig. 1. Performance of the proposed ranking-based survival support vector machine compared against other kernel-based survival models and Cox's proportional hazards model on 100 synthetically generated datasets. In brackets: the kernel function used.

better than the linear SSVM. The simple SSVM suffers from using a simplified objective function with a restricted set of comparable pairs $\mathcal{P}$, despite using an RBF kernel. This clearly indicates that reducing the size of $\mathcal{P}$ to address the complexity of training a non-linear SSVM, as proposed in [32], is inadequate. Although, the Minlip model is based on the same set of comparable pairs, the change in loss function is able to compensate for that – to some degree. Our proposed optimisation scheme and the Minlip model performed comparably (both for clinical and RBF kernel).

Real Data In this section, we will present results on 5 real-world datasets (see table 1) based on 5-fold cross-validation and the clinical kernel [6], which is preferred if feature vectors are a mix of continuous and categorical features as demonstrated above. Table 2 summarises our results. In general, performance measures correlated well and results support our conclusions from experiments on synthetic data described above. The simplified SSVM performed poorly: it ranked last in all experiments. In particular, it was outperformed by the linear SSVM, which considers all comparable pairs in $\mathcal{P}$, which is evidence that restricting $\mathcal{P}$ is an unlikely approach to train a non-linear SSVM efficiently. The Minlip model was outperformed by the proposed SSVM on two datasets (AIDS study and coronary artery disease). It only performed better on the veteran's lung cancer data set and was comparable in the remaining experiments. The linear SSVM achieved comparable performance to the SSVM with clinical kernel on all datasets, except the coronary artery disease data. Finally, Cox's proportional hazard model often performed very well on the real-world datasets, although it does not model non-linearities explicitly. The performance difference between

Table 2. Average cross-validation performance on real-world datasets. iAUC: integrated area under the time-dependent, cumulative-dynamic ROC curve.

		SSVM (ours)	SSVM (simple)	Minlip	SSVM (linear)	Cox
AIDS study	Harrel's c	0.759	0.682	0.729	0.767	0.770
	Uno's c	0.711	0.621	0.560	0.659	0.663
	iAUC	0.759	0.685	0.724	0.766	0.771
Coronary artery disease	Harrel's c	0.739	0.645	0.698	0.706	0.768
	Uno's c	0.780	0.751	0.745	0.730	0.732
	iAUC	0.753	0.641	0.703	0.716	0.777
Framingham offspring	Harrel's c	0.778	0.707	0.786	0.780	0.785
	Uno's c	0.732	0.674	0.724	0.699	0.742
	iAUC	0.827	0.742	0.837	0.829	0.832
Lung cancer	Harrel's c	0.676	0.605	0.719	0.716	0.716
	Uno's c	0.664	0.605	0.716	0.709	0.712
	iAUC	0.740	0.630	0.790	0.783	0.780
WHAS	Harrel's c	0.768	0.724	0.774	0.770	0.770
	Uno's c	0.772	0.730	0.778	0.775	0.773
	iAUC	0.799	0.749	0.801	0.796	0.796

our SSVM and the Minlip model can be explained when considering that they not only differ in the loss function, but also in the definition of the set $\mathcal{P}$. While our SSVM is able to consider all (valid) pairwise relationships in the training data, the Minlip model only considers a small subset of pairs. This turned out to be problematic when the amount of censoring is high, as it is the case for the AIDS and coronary artery disease studies, which contained less than 15% uncensored records (see table 1). In this setting, training a Minlip model is based on a much smaller set of comparable pairs than what is available to our SSVM, which ultimately leads to a Minlip model that generalises badly. Therefore, our proposed efficient optimisation scheme is preferred both with respect to runtime and predictive performance. When considering all experiments together, statistical analysis [8] suggests that the predictive performance of all five survival models is comparably.

3.3 Conclusion

We proposed an efficient method for training non-linear ranking-based survival support vector machines. Our algorithm is a straightforward extension of our previously proposed training algorithm for linear survival support vector machines. Our optimisation scheme allows analysing datasets of much larger size than previous training algorithms, mostly by reducing the space complexity from $O(n^4)$ to $O(n^2)$, and is the preferred choice when learning from survival data with high amounts of right censoring. This opens up the opportunity to build

survival models that can utilise large amounts of complex, structured data, such as graphs and strings.

Acknowledgments. This work has been supported by the CRUK Centre at the Institute of Cancer Research and Royal Marsden (Grant No. C309/A18077), The Heather Beckwith Charitable Settlement, The John L Beckwith Charitable Trust, and the Leibniz Supercomputing Centre (LRZ, www.lrz.de). Data were provided by the Framingham Heart Study of the National Heart Lung and Blood Institute of the National Institutes of Health and Boston University School of Medicine (Contract No. N01-HC-25195).

References

1. Bender, R., Augustin, T., Blettner, M.: Generating survival times to simulate Cox proportional hazards models. Stat. Med. 24(11), 1713–1723 (2005)
2. Cai, T., Tonini, G., Lin, X.: Kernel machine approach to testing the significance of multiple genetic markers for risk prediction. Biometrics 67(3), 975–986 (2011)
3. Chapelle, O.: Training a support vector machine in the primal. Neural Comput. 19(5), 1155–1178 (2007)
4. Chapelle, O., Keerthi, S.S.: Efficient algorithms for ranking with SVMs. Information Retrieval 13(3), 201–205 (2009)
5. Cox, D.R.: Regression models and life tables. J. R. Stat. Soc. Series B Stat. Methodol. 34, 187–220 (1972)
6. Daemen, A., Timmerman, D., Van den Bosch, T., Bottomley, C., Kirk, E., Van Holsbeke, C., Valentin, L., Bourne, T., De Moor, B.: Improved modeling of clinical data with kernel methods. Artif. Intell. Med. 54, 103–114 (2012)
7. Dembo, R.S., Steihaug, T.: Truncated newton algorithms for large-scale optimization. Math. Program. 26(2), 190–212 (1983)
8. Demšar, J.: Statistical Comparisons of Classifiers over Multiple Data Sets. J. Mach. Learn. Res. 7, 1–30 (2006)
9. Doove, L.L., van Buuren, S., Dusseldorp, E.: Recursive partitioning for missing data imputation in the presence of interaction effects. Comput. Stat. Data. Anal. 72(0), 92–104 (2014)
10. Eleuteri, A., Taktak, A.F.: Support vector machines for survival regression. In: Computational Intelligence Methods for Bioinformatics and Biostatistics. LNCS, vol. 7548, pp. 176–189. Springer (2012)
11. Evers, L., Messow, C.M.: Sparse kernel methods for high-dimensional survival data. Bioinformatics 24(14), 1632–1638 (2008)
12. Harrell, F.E., Califf, R.M., Pryor, D.B., Lee, K.L., Rosati, R.A.: Evaluating the yield of medical tests. J. Am. Med. Assoc. 247(18), 2543–2546 (1982)
13. Herbrich, R., Graepel, T., Obermayer, K.: Large Margin Rank Boundaries for Ordinal Regression, chap. 7, pp. 115–132. MIT Press, Cambridge, MA (2000)
14. Hosmer, D., Lemeshow, S., May, S.: Applied Survival Analysis: Regression Modeling of Time to Event Data. John Wiley & Sons, Inc., Hoboken, NJ (2008)
15. Hung, H., Chiang, C.T.: Estimation methods for time-dependent AUC models with survival data. Can. J. Stat. 38(1), 8–26 (2010)
16. Kalbfleisch, J.D., Prentice, R.L.: The Statistical Analysis of Failure Time Data. John Wiley & Sons, Inc. (2002)

17. Kannel, W.B., Feinleib, M., McNamara, P.M., Garrision, R.J., Castelli, W.P.: An investigation of coronary heart disease in families: The Framingham Offspring Study. Am. J. Epidemiol. 110(3), 281–290 (1979)
18. Keerthi, S.S., DeCoste, D.: A modified finite newton method for fast solution of large scale linear SVMs. J. Mach. Learn. Res. 6, 341–361 (2005)
19. Khan, F.M., Zubek, V.B.: Support vector regression for censored data (SVRc): A novel tool for survival analysis. In: 8th IEEE International Conference on Data Mining. pp. 863–868 (2008)
20. Kimeldorf, G.S., Wahba, G.: A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. Ann. Math. Stat. 41(2), 495–502 (1970)
21. Kuo, T.M., Lee, C.P., Lin, C.J.: Large-scale kernel RankSVM. In: SIAM International Conference on Data Mining. pp. 812–820 (2014)
22. Lee, C.P., Lin, C.J.: Large-scale linear RankSVM. Neural Comput. 26(4), 781–817 (2014)
23. Leslie, C., Eskin, E., Noble, W.S.: The spectrum kernel: A string kernel for SVM protein classification. In: Pac. Symp. Biocomput. pp. 564–575 (2002)
24. Li, H., Luan, Y.: Kernel Cox regression models for linking gene expression profiles to censored survival data. In: Pac. Symp. Biocomput. pp. 65–76 (2003)
25. Ndrepepa, G., Braun, S., Mehilli, J., Birkmeier, K.A., Byrne, R.A., Ott, I., Hösl, K., Schulz, S., Fusaro, M., Pache, J., Hausleiter, J., Laugwitz, K.L., Massberg, S., Seyfarth, M., Schömig, A., Kastrati, A.: Prognostic value of sensitive troponin T in patients with stable and unstable angina and undetectable conventional troponin. Am. Heart J. 161(1), 68–75 (2011)
26. Pölsterl, S., Navab, N., Katouzian, A.: Fast training of support vector machines for survival analysis. In: Machine Learning and Knowledge Discovery in Databases. LNCS, vol. 9285, pp. 243–259. Springer (2015)
27. Schölkopf, B., Smola, A., Müller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput. 10(5), 1299–1319 (1998)
28. Shivaswamy, P.K., Chu, W., Jansche, M.: A support vector approach to censored targets. In: 7th IEEE International Conference on Data Mining. pp. 655–660 (2007)
29. Uno, H., Cai, T., Pencina, M.J., D'Agostino, R.B., Wei, L.J.: On the C-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data. Stat. Med. 30(10), 1105–1117 (2011)
30. Uno, H., Cai, T., Tian, L., Wei, L.J.: Evaluating prediction rules for t-year survivors with censored regression models. J. Am. Stat. Assoc. 102, 527–537 (2007)
31. Van Belle, V., Pelckmans, K., Suykens, J.A.K., Van Huffel, S.: Support vector machines for survival analysis. In: 3rd International Conference on Computational Intelligence in Medicine and Healthcare. pp. 1–8 (2007)
32. Van Belle, V., Pelckmans, K., Suykens, J.A.K., Van Huffel, S.: Survival SVM: a practical scalable algorithm. In: 16th European Symposium on Artificial Neural Networks. pp. 89–94 (2008)
33. Van Belle, V., Pelckmans, K., Suykens, J.A.K., Van Huffel, S.: Learning transformation models for ranking and survival analysis. J. Mach. Learn. Res. 12, 819–862 (2011)
34. Van Belle, V., Pelckmans, K., Van Huffel, S., Suykens, J.A.K.: Support vector methods for survival analysis: a comparison between ranking and regression approaches. Artif. Intell. Med. 53(2), 107–118 (2011)
35. Vishwanathan, S., Schraudolph, N.N., Kondor, R., Borgwardt, K.M.: Graph kernels. J. Mach. Learn. Res. 11, 1201–42 (2011)

36. Zhang, W., Ota, T., Shridhar, V., Chien, J., Wu, B., Kuang, R.: Network-based survival analysis reveals subnetwork signatures for predicting outcomes of ovarian cancer treatment. PLoS Comput. Biol. 9(3), e1002975 (2013)

Learning in silico communities to perform flow cytometric identification of synthetic bacterial communities

Peter Rubbens[1], Ruben Props[1] Nico Boon[1], and Willem Waegeman[1]

[1]Ghent University, Ghent, Belgium,
Peter.Rubbens@UGent.be

Abstract. Flow cytometry is able measure up to 50.000 cells in various dimensions in seconds of time. This large amount of data gives rise to the possibility of making predictions at the single-cell level, however, applied to bacterial populations a systemic investigation lacks. In order to combat this deficiency, we cultivated twenty individual bacterial populations and measured them through flow cytometry. By creating *in silico communities* we are able to use supervised machine learning techniques in order to examine to what extent single-cell predictions can be made; this can be used to identify the community composition. We show that for more than half of the communities consisting out of two bacterial populations we can identify single cells with an accuracy >90%. Furthermore we prove that in silico communities can be used to identify their in vitro counterpart communities. This result leads to the conclusion that in silico communities form a viable representation for synthetic bacterial communities, opening up new opportunities for the analysis of bacterial flow cytometric data and for the experimental study of low-complexity communities.

Keywords: flow cytometry, microbiology, in silico communities, synthetic bacterial communities, linear discriminant analysis, random forests

1 Introduction

Flow cytometry (FCM) is an experimental technique which characterizes individual cells in terms of fluorescence and scatter signals; this results in a multidimensional description of every cell. As the analysis of cells is increasing (up to 50.000 of cells per second), alongside with the dimensionality of the data (up to 50 dimensions will be available soon), the field of FCM *bioinformatics* is growing accordingly [11]. A promising approach of analyzing FCM data is the use of supervised machine learning techniques in order to identify single cells, an approach which has been used in for example the recognition of leukemia [17] or to identify various populations of phytoplankton [3], [12].

However, applied to bacterial populations this approach seems to be lacking a thorough investigation. Two reports exist, of which the first analyzed the effect of various cocktails of fluorescent staining [6], and the second the extent to which

individual cells can be classified using multiple but only scatter signals [13]. However the number of populations used in latter studies is small, remaining only to the binary setting.

To investigate this issue more thoroughly, we cultivated twenty different bacterial populations and measured them individually through FCM. We propose the use of *in silico communities*, communities we created by aggregating the data coming from these individual cultures. This approach leads to two advantages; first, we are able to use supervised machine learning methods as we know the individual label of every cell. Second, we have the ability to create a wide spectrum of bacterial communities ranging from low complexity to high complexity, and ranging from low evenness (i.e., unevenly distributed populations) to high evenness communities. For example, for a population richness of $S = 2$ we already have the possibility of analyzing 190 different bacterial communities, only considering the population richness.

In the first section we perform a thorough analysis regarding the possibility of making single-cell predictions. We will show that for a binary setting we are able to achieve high accuracies for a majority of possible bacterial communities. Next, we consider a multiclass setting as well, showing that FCM data should be feasible for increasing population richness. We chose methods which extend to a multiclass setting in a natural way. For now we opted to use Linear Discriminant analysis (LDA), which is an established method in microbial ecology to perform multivariate analyses [14], and Random Forests, known for its high performance in various applications with only one hyperparameter to tune [8].

In the second section of the paper we show that we can use the statistical properties of in silico communities in order to classify individual cells contained in resembling (i.e., containing the same bacterial populations) in vitro communities. This is not self-evident for two reasons; first, flow cytometric measurements suffer from technical variations and second, it has been proven that bacterial populations exhibit heterogeneous behavior which is reflected in FCM data [18]. In order to test this hypothesis, we created so-called *abundance gradients*; we define an abundance gradient as a set of in vitro communities which contain the same two bacterial populations, but in varying abundances. We will show that we are able to retrieve these relative abundances using classifiers which are trained on an evenly distributed in silico community. This result forms a strong argument that flow cytometric in silico communities form a proper representation for synthetic bacterial communities, and thus can be used for further study; furthermore, it enables researchers the use of supervised methods combined with FCM in order to analyze low-complexity communities.

2 Exploratory analysis of in silico communities

In order to systematically investigate the possibility of making single-cell predictions, we have cultivated twenty bacterial populations and measured them through FCM; a full list can be found in Tab. 1.

Table 1. List of cultivated bacterial populations measured through FCM (dataset 1).

Bacterial population	Culture collection reference
Agrobacter rhizogenes	UFZ requested [16]
Bacillus subtilis	LMG 7135
Burkholderia ambifaria	LMG 19182
Citrobacter freundii	DSMZ 15979
Cupriavidus necator	LMG 1201
Cupriavidus pinatubonensis	LMG 1197
Edwardsialla ictaluri	LMG 7860
Enterobacter aerogenes	DSMZ 30053
Escherichia coli	DSMZ 2840
Janthinobacterium sp. B3	UFZ requested [16]
Klebsiella oxytoca	LMG 3055
Lactobacillus plantarum	LMG 9211
Micrococcus luteus	UFZ requested [16]
Pseudomonas fluorescens	R 23898
Pseudomonas putida	R 17801
Rhizobium radiobacter	LMG 287
Shewanella oneidensis	LMG 19005
Sphingomonas aromaticivorans	LMG 18303
Streptococcus salivarius	LMG 11489
Zymomonas mobilis subsp. mobilis	LMG 460

For $S = 2$ we have analyzed all possible pairwise combinations (this number equals 190). We created in silico communities sampling an equal amount of 5.000 cells for every population; this means that an in silico community consists out of 10.000 cells. We trained a classifier using LDA and Random Forests on 70% of the in silico community; hereafter we predicted the population to which cells belong to contained in the 30% held out test set. We note that there was no need to tune the Random Forest classifier; using the preset $\sqrt{K}$, with K being the total number of available features to choose from at every split ($K = 12$, of which eight are fluorescence signals and four are scatter signals), gives rise to (near-)optimal results, in accordance with [2]. We note that after having performed ten-fold cross-validation for K on twenty randomly picked in silico communities, the increase in accuracy was 0.007 at the utmost. We expressed our performance for every in silico community in terms of the *area under the receiver operating characteristic curve* (AUC) and the *accuracy* (Fig. 1).

We note that the ensemble of pairwise combinations of populations give rise to performance accuracies ranging from 0.99 to near random guessing predictions; in other words, we were not biased towards highly discriminative populations. We have further summarized our results in Tab. 2, reporting the mean AUC and accuracy, along with their standard deviations, and the percentage of communities giving rise to performances higher than 0.90. Based on these numbers, we conclude that we are able to achieve high accuracies for a significant amount of possible communities. We note that a combination of *E. ictaluri* -

Table 2. Summary of analysis using LDA and Random Forests (RF) for $S = 2$. We denote the mean AUC (μ_{AUC}) and accuracy (μ_{acc}), along with the standard deviation ($\sigma_{\text{AUC/acc}}$) and the percentage of communities reporting a performance of 0.90 or higher.

	μ_{AUC}	μ_{acc}	σ_{AUC}	σ_{acc}	AUC > 0.90	acc > 0.90
LDA	0.90	0.83	0.089	0.088	62%	27%
RF	0.95	0.90	0.071	0.085	82%	65%

S. aromaticivorans results in the highest AUC of 0.999, a combination of *K. oxytoca* - *Z. m. subsp. mobilis* results in the highest accuracy, being 0.996.

Generally using Random Forests results in better performances than LDA, however, this is not always the case. 45% of the possible in silico communities report an increase in AUC of less than 0.03, 17% report an increase in accuracy less than 0.03.

In order to assess the fruitfulness of analyzing bacterial communities in a multiclass setting, we created 150 randomly chosen in silico communities for every increment of S, for which populations are evenly sampled (again 5.000 cells per population); this means that the total amount of cells contained in an in silico community N_{tot} equals $N_{\text{tot}} = S \times 5.000$. We used the same approach as described previously to perform LDA and Random Forests (i.e., creating a training set using 70% of the data and a test set using the other 30% of the data). We calculated the accuracy for every test set, after which we calculated the mean accuracy accompanied with its confidence-interval (CI) for every S (Fig. 2.)

For all values of S one is able to make single-cell predictions significantly better than random guessing. As S increases, both the mean accuracy and the size of the CI decline. This is due to the fact that for growing population richness, the degree in overlap between populations in the multidimensional 'FCM-space' starts growing. Therefore it is harder for classifiers to make a distinction between populations, which results in performances that are lower and more centered.

The difference in performance between the two classifiers increases as S increases. This means that for communities with a low richness ($S = 2, 3$) LDA might every so often be a sufficient method to perform single-cell predictions, however this is not always the case. This also means that although for low S a linear combination of variables already discriminates populations quite well, predictions can be improved by choosing classifiers which are able to combine variables in a non-linear way, especially for higher complexity communities.

3 Identifying bacterial populations in synthetic communities using in silico communities

We created three abundance gradients in order to verify to what extent an in silico community is able to identify its an in vitro community containing the same bacterial populations. We chose combinations which initially (according

Table 3. Three different combinations (Comb.) of bacterial populations used to create abundance gradients (dataset 2).

Comb.	Population 1	Population 2
1	*P. fluorescens*	*P. Putida*
2	*A. rhizogenes*	*Janthinobacterium sp. B3*
3	*M. luteus*	*S. oneidensis*

to the analysis described above) reported a low (Comb. 1), medium (Comb. 2) and high performance (Comb. 3), respectively (Tab. 3). As we do not know the individual labels of the cells contained in these communities, we predicted the relative abundance of populations present in a community, which can be derived by summing the predicted labels of individual cells. We express the performance in terms of the *root mean squared error* (RMSE):

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(p_i - \hat{p}_i)^2}{n}}, \tag{1}$$

with p being the target relative abundance to predict, $\hat{p}$ the predicted relative abundance and n the the total number of bacterial communities constituting the abundance gradient; $n = 13$ in this case[1]. Because we measured the populations individually beforehand, we are able to construct the same abundance gradient in silico. This enables us to not only carefully examine to what extent these abundances can be retrieved, but also to compare the in silico results with the in vitro results. The resulting RMSE is summarized in Tab. 4, along with the mean AUC calculated for the ensemble of communities constituting in silico abundance gradients; the predicted abundance gradients are visualized in Fig. 3.

Comb. 2 (Figs. 3CD) and 3 (Figs. 3EF) are well-predicted as opposed to Comb. 1 (Figs. 3AB), which is reflected in the RMSE; the mean AUC however reports quite a high AUC for Comb. 1 when using Random Forests, albeit still lower than for Comb. 2 and 3. The results for the in vitro analysis of Comb. 2 and 3 give rise to a similar RMSE, although we expected from initial performances that these values would be different. To investigate this issue, we added additional results in Tab. 5. We report the performance of a classifier in terms of the accuracy and the AUC trained on 70% and evaluated on 30% of the new in silico communities; in other words, classifiers were trained in the same way as in the previous section, so that a succinct comparison is possible with the originally reported values (*).

We note that although the performances are similar for Comb. 3, this is not the case for Comb. 1 and 2. Whereas the performances for Comb. 1 initially reported higher, the performances for Comb. 2 initially reported lower. This explains why the results for the in vitro analysis for Comb. 2 and 3. report similar

[1] We have constructed communities with the following relative abundances (population 1/population 2): 1%/99%, 5%/95%, 10%/90%, 20%/80%, 30%/70%, 40%/60%, 50%/50%, 60%/40%, 70%/30%, 80%/20%, 90%/10%, 95%/5% and 99%/1%.

Table 4. RMSE and mean AUC (μ_{AUC}) for predicted abundance gradients. RMSE has been calculated between the predicted gradients and the target gradients, both in silico and in vitro, having used LDA and a Random Forest classifier. μ_{AUC} has been calculated by calculating the AUC for every in silico community, and averaging over all in silico communities constituting the respective abundance gradient.

	Comb. 1	Comb. 2	Comb. 3
RMSE LDA in silico	0.29	0.0060	0.10
RMSE RF in silico	0.21	0.0036	0.022
RMSE LDA in vitro	0.51	0.036	0.096
RMSE RF in vitro	0.48	0.036	0.032
μ_{AUC} LDA in silico	0.64	1.0	0.93
μ_{AUC} RF in silico	0.88	1.0	1.0
σ_{AUC} LDA in silico	0.022	0.00070	0.0054
σ_{AUC} RF in silico	0.055	0.00039	0.00088

Table 5. Comparison of performances using LDA and Random Forests using datasets 1 as opposed to dataset 2. Performance using LDA and a Random Forest classifier for in-silico communities created with the same populations as in Comb. 1, 2 and 3, using the data reported in the previous section (dataset 1, denoted with *) and the abundance gradient data (dataset 2). These in silico communities are constructed and analyzed in exactly the same way, that is, they are evenly distributed communities consisting out of the same number of cells. Classifiers are trained on 70% of the data and evaluated on the opposite 30% test data.

	Comb. 1	Comb. 2	Comb. 3
AUC LDA*	0.64	0.82	0.96
AUC LDA	0.62	1.0	0.93
acc LDA*	0.62	0.77	0.92
acc LDA	0.59	0.99	0.91
AUC RF*	0.82	0.94	1.0
AUC RF	0.70	1.0	0.99
acc RF*	0.75	0.87	0.99
acc RF	0.64	1.0	0.97

results. However, this implies that although our approach is fruitful to analyze synthetic communities, performances are not yet exactly reproducible when individual bacterial populations are measured at different time points through FCM.

Furthermore, we emphasize the similar behavior between the in silico analysis (Fig. 3, left panel) and in vitro analysis (Fig. 3, right panel). We see that results are almost identical using either LDA or a Random Forest classifier analyzing Comb. 2. Moreover, inspecting Comb. 3, we note that using Random Forests increases the performance significantly, both for the in silico communities and in vitro communities; LDA suffers from a systematic bias, which is almost entirely (but not in full) reduced when one uses Random Forests.

4 Conclusion

After a thorough survey we can state that it is possible to predict the population to which bacterial single cells belong based on FCM data for low-complexity communities. Furthermore we have shown in a rigorous manner that in silico communities can be used to identify their in vitro counterpart communities. This leads to the conclusion that in silico communities form a viable representation for synthetic bacterial communities, and thus, we are allowed to use these in silico communities for further study. Supervised machine learning methods become therefore available to study issues as FCM data transformation [9] or feature selection, a topic studied in the field of immunology [10], but which seems te be lacking thus far in the field of microbiology. For low-complexity communities 'off-the-shelf' classifiers will most of the time already suffice to identify bacterial single cells. The outcome of this research therefore complies with the motivation to integrate supervised machine learning methods into standard FCM software [4].

A natural extension of this research would be to find the optimal multiclass method to analyze FCM data; a number of possibilities exist, ranging from binary classifiers which are naturally extendable to a multiclass setting or a combination of binary classifiers using a *one-versus-one (OVO)* or *one-versus-all (OVA)* approach [1]. However, it has to be noted that the performance of classifiers is not yet reproducible. The reason behind this is that bacterial populations exhibit heterogeneous behavior, which is reflected in FCM data [18]. However, FCM has been suggested to further quantify bacterial heterogeneity [5],[7], research in which in silico communities in combination with supervised machine learning methods might prove its value.

References

1. Aly, M.: Survey on multi-class classification methods. Tech. rep. (2005)
2. Bernard, S., Heutte, L., Adam, S.: Influence of hyperparameters on random forest accuracy. In: Benediktsson, JA and Kittler, J and Roli, F (ed.) Multiple Classifier Systems, Proceedings. Lecture Notes in Computer Science, vol. 5519, pp. 171–180. Int Assioc Patern Recognit & Tech Comm 1; IEEE Geosci & Remote Sensing Soc, IEEE Iceland Sect; Univ Cagliari; Univ Surrey (2009), 8th International Workshop on Multiple Classifier Systems, Univ Iceland, Reykjavik, ICELAND, JUN 10-12, 2009
3. Boddy, L., Morris, C., Wilkins, M., Al-Haddad, L., Tarran, G., Jonker, R., Burkill, P.: Identification of 72 phytoplankton species by radial basis function neural network analysis of flow cytometric data. Mar Ecol Prog Ser 195, 47–59 (2000)
4. Davey, H.M.: Prospects for the automation of analysis and interpretation of flow cytometric data. Cytometry A 77A(1), 3–5 (JAN 2010)
5. Davey, H.M., Winson, M.K., et al.: Using flow cytometry to quantify microbial heterogeneity. Curr Issues Mol Biol 5(1), 9–15 (2003)
6. Davey, H., Jones, A., Shaw, A., Kell, D.: Variable selection and multivariate methods for the identification of microorganisms by flow cytometry. Cytometry 35(2), 162–168 (FEB 1 1999)

7. Fernandes, R.L., Nierychlo, M., Lundin, L., Pedersen, A.E., Tellez, P.E.P., Dutta, A., Carlquist, M., Bolic, A., Schapper, D., Brunetti, A.C., Helmark, S., Heins, A.L., Jensen, A.D., Nopens, I., Rottwitt, K., Szita, N., van Elsas, J.D., Nielsen, P.H., Martinussen, J., Sorensen, S.J., Lantz, A.E., Gernaey, K.V.: Experimental methods and modeling techniques for description of cell population heterogeneity. Biotechnol Adv 29(6), 575–599 (NOV-DEC 2011)
8. Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems? J Mach Learn Res 15, 3133–3181 (2014)
9. Finak, G., Perez, J.M., Weng, A., Gottardo, R.: Optimizing transformations for automated, high throughput analysis of flow cytometry data. BMC Bioinformatics 11 (NOV 4 2010)
10. Hassan, S.S., Ruusuvuori, P., Latonen, L., Huttunen, H.: Flow cytometry-based classification in cancer cesearch: a view on feature selection. Cancer Informatics pp. 75–85 (04 2016)
11. O'Neill, K., Aghaeepour, N., Spidlen, J., Brinkman, R.: Flow cytometry bioinformatics. PLoS Comput Biol 9(12) (DEC 2013)
12. Pereira, G.C., Ebecken, N.F.F.: Combining in situ flow cytometry and artificial neural networks for aquatic systems monitoring. Expert Syst Appl 38(8), 9626–9632 (AUG 2011)
13. Rajwa, B., Venkatapathi, M., Ragheb, K., Banada, P.P., Hirleman, E.D., Lary, T., Robinson, J.P.: Automated classification of bacterial particles in flow by multiangle scatter measurement and support vector machine classifier. Cytometry A 73A(4), 369–379 (APR 2008), 24th International Congress of the International-Society-for-Analytical-Cytology, Budapest, HUNGARY, MAY 17-21, 2008
14. Ramette, A.: Multivariate analyses in microbial ecology. FEMS Microbiol Ecol 62(2), 142–160 (NOV 2007), Joint Symposium of the Environmental-Microbiology-Group/British-Ecological-Society/Society-for- General-Microbiology, Univ York, York, ENGLAND, SEP 13, 2006
15. Rubbens, P., Props, R., Boon, N., Waegeman, W.: Flow cytometric single-cell identification of populations in synthetic bacterial communities. Under review
16. Saleem, M., Fetzer, I., Dormann, C.F., Harms, H., Chatzinotas, A.: Predator richness increases the effect of prey diversity on prey yield. Nat Commun 3 (DEC 2012)
17. Toedling, J., Rhein, P., Ratei, R., Karawajew, L., Spang, R.: Automated in-silico detection of cell populations in flow cytometry readouts and its application to leukemia disease monitoring. BMC Bioinformatics 7 (JUN 5 2006)
18. Vives-Rego, J., Resina, O., Comas, J., Loren, G., Julia, O.: Statistical analysis and biological interpretation of the flow cytometric heterogeneity observed in bacterial axenic cultures. J Microbiol Methods 53(1), 43–50 (APR 2003)

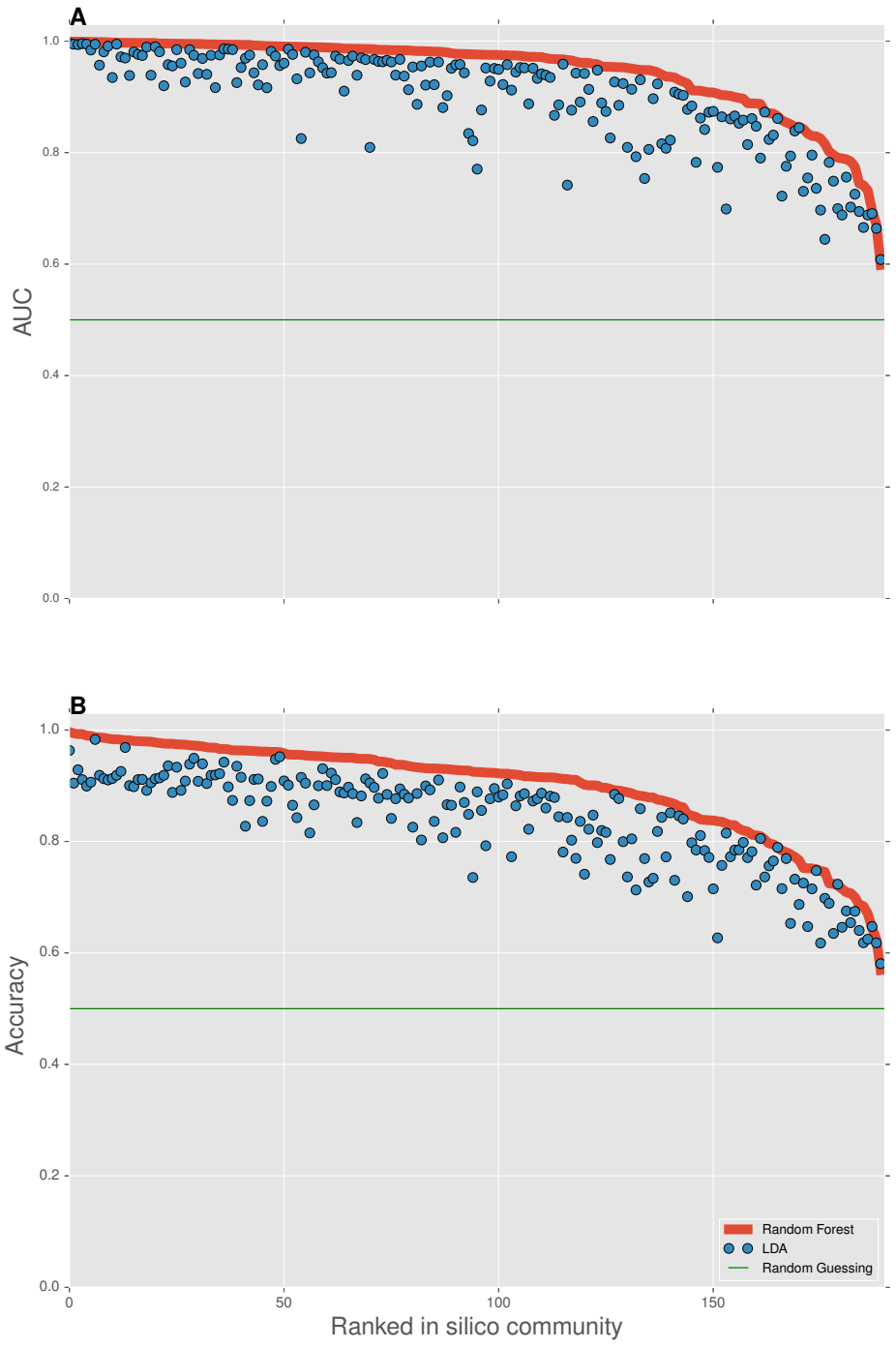


Fig. 1. Ranked performances using LDA and Random Forests for $S = 2$. A Ranked AUC. **B** Ranked accuracy. Performances are visualized for allevenly distributed 190 in silico communities and have been ranked in descending order according to the performances resulting from using Random Forests, accompanied with performances resulting from using LDA on the same in silico community. The performances have been calculated on a 30% held-out test set; figure taken from [15].

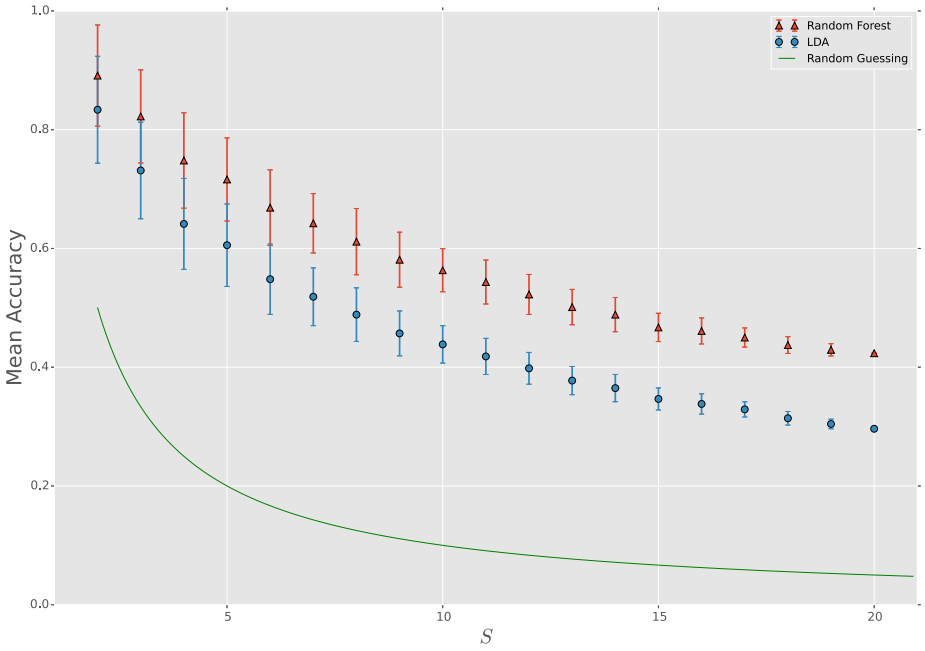


Fig. 2. Mean accuracy for increasing population richness using LDA and Random Forests. Mean accuracy along with a 68%-CI is displayed, resulting from an analysis using LDA and Random Forests for 150 randomly chosen in silico communities for $S = 2, ..., 18$ (for $S = 19$ and $S = 20$ this number becomes 20 and 1 respectively); every in silico community is evenly distributed. The accuracy has been calculated on a 30% held-out test set, after which the mean accuracy is calculated for the ensemble of silico communities for every increment of S; figure taken from [15].

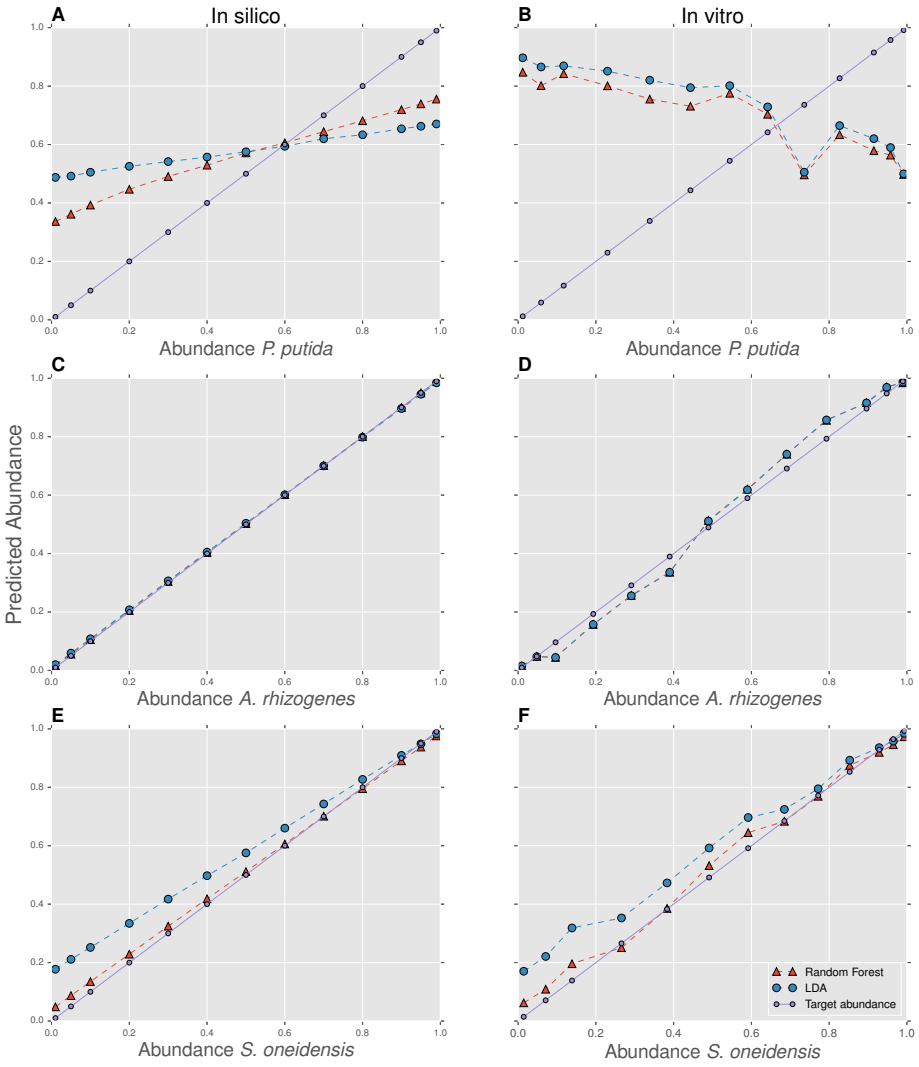


Fig. 3. Predicted abundance gradients. AB Comb. 1: *P. putida* – *P. fluorescens*; **CD** Comb. 2: *S. oneidensis* – *M. luteus*; **EF** Comb. 3: *A. rhizogenes* – *Janthinobacterium sp. B3*. Both the in silico (left panel) and in vitro (right panel) abundance gradients are visualized. The predicted relative abundance gradient is plotted against its target (i.e., designed in silico and in vitro) relative abundance for the first population of the three combinations. It follows that the relative abundance of the opposite population equals one minus the relative abundance of the first population (as $S = 2$); figure taken from [15].

Natural language processing methods in biological activity prediction

Szymon Nakoneczny[1] and Marek Śmieja[1]

Faculty of Mathematics and Computer Science, Jagiellonian University
Lojasiewicza 6, 30-348 Kraków, Poland
{szymon.nakoneczny, marek.smieja}@ii.uj.edu.pl

Abstract. Virtual screening is a process in which databases of chemical compounds are searched in order to find structures characterized with a high biological activity, possible drug candidates [12]. Our goal is to combine the natural language processing methods with SMILES, a text representation of compounds, in the classification of active compounds. Since SMILES encodes a graph structure of a compound into a sequence of symbols, it is not reasonable to build a language model directly from SMILES. In this paper we propose various strategies to adjust the underlying representation to create the best possible language model of compounds. The introduced modifications are verified in an extensive experimental study. The results show that the proposed approach outperforms classical fingerprint representations and does not require any specialized chemical knowledge.

Keywords: n-gram language model, bag-of-n-grams, support vector machine, virtual screening, SMILES

1 Introduction

Only a small percentage of a huge number of organic compounds can serve as drugs. As it is not possible to synthesize all of them in a laboratory, the first move is to find biological active compounds by means of a computer analysis. Before machine learning, a common approach was to simulate a compound to target protein docking and estimate a ligand activity [13]. Currently, one of the most popular methods is to represent chemical compounds as a vector called fingerprint and analyze them with machine learning approaches. Each position in this representation refers to some substructure of a compound's graph and often in order to use the most meaningful substructures, a huge number of them is being handcrafted by chemists [11].

Our goal is to apply natural language processing methods with a text representation called SMILES. In this representation, a compound graph is flattened into a sequence of symbols which makes the structure information much harder to understand. Fortunately, working with a representation which computational needs matches the use of fingerprints while still having all of the information

encoded inside of the representation opens new possibilities for applying machine learning methods. The effort is taken to use and improve n-gram language model and bag-of-words representation known from NLP. In scope of improvements, SMILES modifications with an aim to increase its informativity will be tested. The best modifications were chosen with experimental analysis which also proves that our solution can achieve higher scores than standard fingerprint representations without a use of any specialized chemical knowledge.

The paper is organized as follows. Next section gives a brief review of related SMILES based approaches. Section 3 provides a more detailed description of SMILES representation and recalls n-gram language model. The proposed modifications, which allow to combine SMILES with NLP, are placed at the end of this section. Experimental results are included in Section 4. Finally, the conclusion is given.

2 Related work

The SMILES-based approaches to chemical compounds analysis are rather rare. A problem of bioactivity prediction with text representation was tackled by Apilak Worachartcheewan et al. [16]. The idea of this approach is to define a set of features based on a presence of some abstract subsequences in SMILES, which should be relevant for activity analysis, and then optimizing the model with Monte Carlo approach. It is worth to mention that this approach is very similar to building a structural fingerprint.

David Vidal et al. [14] proposed to build a general vector representation based on all subsequences of a given length found in SMILES. This representation can be then used in any task concerning chemical compounds analysis.

In comparison to those ideas, our solution is mostly motivated by natural language processing achievements. Due to the n-gram language model characteristics, subsequences used are varying in length and their set is not limited by any domain knowledge. It is a model objective to learn the value of subsequences given the problem of bioactivity prediction. Thereby, our approach does not require much of organic chemistry knowledge.

3 Natural language processing with SMILES

In this section, we provide more detailed description of SMILES representation and recall the n-gram language model. We also propose here various strategies how to combine these two tools to create an efficient representation for applying machine learning methods.

3.1 SMILES representations

SMILES (Simplified Molecular Input Line Entry System) is a simple, easy to understand language in which molecules and chemical reactions can be written with

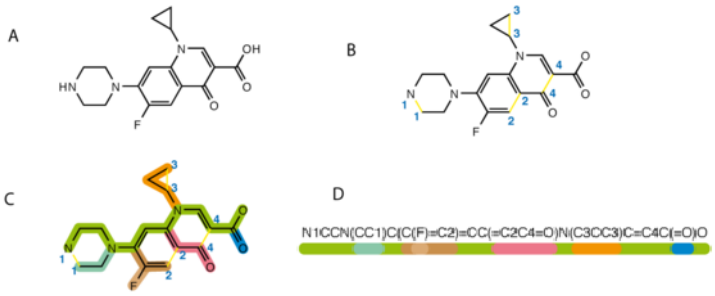


Fig. 1: Example of SMILES representation.

ASCII characters representing atoms and bonds [3]. Its most important advantage is an ability to uniquely identify chemical compound, which is something that simple molecular formula does not provide, while taking approximately 50%-70% less space than corresponding matrix representation. SMILES is created by traversing a compound graph and writing down atoms and bonds along the way.

Basic rules which govern this process are (see Figure 1 for example):

- Atoms are represented by their atomic symbol. If an atom does not belong to subset of organic elements or it is an isotope or its charge has to be specified, it is putted with all this information into square brackets. Besides the isotope case, hydrogen atoms are always implied by compound structure in which case they are being omitted.
- Single, double and triple bonds are represented by symbols '-', '=' and '#' respectively. Th default bond is a single one and '-' symbol can be omitted.
- Branches are putted inside brackets.
- Cyclic structures are represented by removing one of the bonds in compound graph in order to create a tree. Removed bond is then marked with one and same digit following an atom which opens and closes this bond. Digits can be reused after a bond closing and in rare cases when number higher than 9 is used, it has to be preceded with '%' symbol.
- Not connected structures are written individually and separated with a '.' symbol.

Extensions to SMILES covers aromaticity and unique representation. In case of aromaticity, the idea is to encode this information with small atomic symbols, thanks to which it can be easily detected without any algorithm and SMILES is being even further simplified. The example is aromatic ring C1=COC=C1 which will be now written as c1cocc1. The rules described above, starting with an order in which a graph is being traversed, are not deterministic. The idea with canonical SMILES is to create additional set of rules and default behaviors in order to create only one unique SMILES for each of the compounds.

3.2 N-gram language model

One of the first problems tackled by natural language processing methods was a missing last word in sentence [7]. N-gram language model, which builds a probabilistic language model based on n previous words, turns out to be a perfect solution to this problem [1]. Having a sentence of words $w_1, \ldots, w_N$, the probability of word w_i occurring after the last n words is given by[1]:

$$P(w_i|w_{i-n+1} \ldots w_{i-1}) = \frac{|w_{i-n+1} \ldots w_i|}{|w_{i-n+1} \ldots w_{i-1}|}$$

Therefore, the probability of a sequence can be calculated as:

$$P(w_1, \ldots, w_N) = \prod_{i=1}^{N} P(w_i|w_{i-n+1} \ldots w_{i-1})$$

Because longer sequences will automatically get lower probability, the perplexity is used to calculate how well a given sentence fits to a created model.

Definition 1. *Perplexity of a $w = w_1, \ldots, w_N$ sequence of elements, is defined as*

$$PPL(w) = P(w_1, \ldots, w_N)^{-\frac{1}{N}}$$

This measure is not dependent on a sequence length and is minimized by the best fitting sequence.

A problem with an n-gram model arises when a sequence contains such n-grams which were not present in a training set, therefore their probability equals 0. Smoothing techniques were introduced to deal with this and other n-gram model problems. One of such methods is Jelinek-Mercer interpolation [17], which idea is to use information of smaller contexts to interpolate the probability of a longer one. It is given by:

$$\hat{P}(w_i|w_{i-n+1} \ldots w_{i-1}) =$$
$$\lambda_{w_{i-n+1}^{i-1}} \hat{P}(w_i|w_{i-n+1} \ldots w_{i-1}) + (1 - \lambda_{w_{i-n+1}^{i-1}}) \hat{P}(w_i|w_{i-n+2} \ldots w_{i-1})$$

where $\lambda \in [0, 1]$ parameters can be grouped and then fitted with an expectation-maximization algorithm.

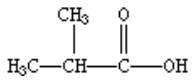
Applications of n-gram model go beyond sentence modeling with words. One can use the perplexity to construct a decision function which allows to classify a given sentence to one of underlying classes. For a simplicity, let us consider a binary classification problem and assume that separate n-gram models were constructed for two groups of texts. Then the probability of assigning a new text x to class c_i, for $i = 1, 2$ equals:

$$P(c_i|x) = \frac{PPL_{c_i}(x)}{PPL_{c_1}(x) + PPL_{c_2}(x)}. \tag{1}$$

N-gram model provides good results in many areas and its biggest advantages are simplicity and scalability. With increase of data, the space requirement grows slower as more n-grams are being repeated.

[1] we replace w_{-i}, for $i = 0, 1, \ldots$ by an empty symbol $< s >$

Table 1: Adding a context to SMILES 3-gram model

compound	SMILES	SMILES with context added
	CC(C)C(=O)O	CC(C)eCCC(=O)eCCO

3.3 Combining SMILES with NLP

The first step of constructing n-gram model relies on the tokenization of text into base symbols (words). This is motivated with balancing information carried by different tokens. It resembles a situation from natural language processing in which word lengths are completely ignored by an n-gram model.

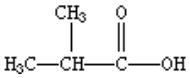
Two approaches are proposed to tokenize SMILES:

- **Baseline.** The simplest way to apply n-gram model to SMILES is to split SMILES into single characters and build a model upon such elements.
- **Tokenization.** Let us observe that it might not be reasonable to split an atom symbol 'Br' int o 'B' and 'r'. Following that idea, elements longer than one symbol, like 'Br', are gathered into single tokens. Also, numbers which are present in square brackets or after '%' symbol are matched together to distinct them from single digits which stand for circular structures.

SMILES provides a text representation of chemical compound structure. However, to build a reasonable N-gram model from SMILES one has to keep in mind that this representation encodes a graph structure into a sequence of symbols. To deal with that problem, the following SMILES modifications are proposed:

- **Simplification.** Idea proposed in [16] and [14] is to simplify circular structure information by replacing all the digits with '0' and to simplify aromaticity information by replacing all the letters with their capitals.
- **Context.** While traversing a branch in graph, ending it and going back to where it starts makes the n-gram model to still read the elements on the end on that branch which can actually be placed far away from a current position. The idea is to duplicate the last $(n-1)$ elements which were read before the branch started and are connected to the current path. Also proceed them with $(n-2)$ symbols 'e' to fully disconnect the structures which are not connected in compound graph. Table 1 shows an example of such SMILES modification.
- **Short paths.** In order not to multiply the information contained in the branching (as in the above context addition), the idea is to only add $(n-2)$ symbols 'e' which results in disconnecting some parts of a graph and working on shorter paths. Similarly to an NLP situation in which individual parts of a sentence are separated with a coma symbol, in case of SMILES such separation is always done with going back to the beginning of a branch represented by symbol ')'.

Table 2: An example of new SMILES creation

compound	SMILES	new reversed SMILES
	CC(C)C(=O)O	CC(C(O)=O)C

– **New SMILES.** Because there is more than one way of traversing a graph, to allow n-gram model to use more of the paths available one can create new SMILES by changing the order of graph traversing. Both sequences can be then separated with a special symbol in order to create one representation. Because the number of all possible SIMILES is extremely high and it makes our model more complex, there is a need to solve an information-complexity trade-off by maximizing the information gain when adding new SMILES. Starting with creation of one additional representation, it was done by reversing the order in which branches are read so that the first branch is read as the last one. Table 2 shows an example. As you can see, non of the paths used before is being repeated in the new SMILES which maximizes the information gain.

4 Experiments

In this section we present a complete experimental analysis, which verifies methods described in previous section. First, we describe the preparation of data. Next, we test how the modifications introduced in Section 3.3 affect the results obtained by applying a perplexity classifier (1). Finally, we compare different classifiers on a vector representation created from the n-gram model.

4.1 Data preparation

Chemical compounds data were downloaded from the ChEMBL database [4]. The target proteins are called receptors and each of those defines different activation value. In order to reliably test our approaches, a set of 6 serotonin receptors was chosen: 5-HT1a, 5-HT6, 5-HT7, 5-HT2a, 5-HT2b and 5-HT2c. They all define separated datasets in which the same compounds may or may not occur. Because activity test on human and rat provides similar results, both of those sets were downloaded. Activity function is measured by an inhibition constant K_i [15]. Compounds with an inhibition constant less than or equal to 100 nM were considered active; ligands with K_i higher than 1000 nM were used as inactive. The range (100, 1000) is removed from data as not clear enough.[2]

[2] From a machine learning point of view, removing those compounds creates an artificial absence of data. However, to make the problem easier and be consistent with the methodology used by chemists, the given approach is used.

Table 3: Number and ratio of different compounds for receptors

	5-HT1a	5-HT6	5-HT7	5-HT2a	5-HT2b	5-HT2c
actives	4376	1597	888	2041	410	1289
inactives	1057	379	365	995	335	1023
ZINC	39384	14373	7992	18369	3640	11601
act. / inact.	4.14	4.21	2.43	2.05	0.32	1.26
act. / ZINC	0.11	0.11	0.11	0.11	0.11	0.11

Unfortunately, bioactivity data can be very noised as varying laboratory conditions can highly change test results. In case of duplicated records, in order to reduce an influence of outliers, a median value is calculated.

The last problem with ChEMBL database is an unbalanced amount of active and inactive elements. Because the majority of molecules in the real world are the inactive ones, their discovery is nothing special and they are not published to databases. It results in a completely opposite compounds ratio. To deal with that problem, ZINC compounds are being used [6]. Those are artificially generated compounds with a high probability of being inactive. Because those compounds are significantly different than the ones obtained from ChEMBL, mixing ZINC with ChEMBL compounds would result in a yet another problem different than the real one. The most common strategy is to create two unbalanced datasets, one containing all the compounds from ChEMBL and another with active molecules from ChEMBL and inactive ones in the form of ZINC. An advantage of the first dataset is a high similarity between active and inactive compounds which makes the problem much harder, while second dataset describes better the real ratio between compounds. Having those datasets and 6 different receptors, it gives us total of 12 problems to solve. The number of compounds and their ratio in all of the datasets is summarized in the Table 3.

The models are tested with 10-fold cross-validation and the error is measured with the Matthews correlation coefficient [9] defined as:

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}},$$

where:

TP - number of examples correctly classified as positive

TN - number of examples correctly classified as negative

FP - number of examples incorrectly classified as positive

FN - number of examples incorrectly classified as negative

MCC results vary in range [-1, 1]. The reason for choosing it is the fact that it is one of the most reliable errors in case of unbalanced problems.

Table 4: Mean MCC score of 6-gram model in active and ZINC compounds classification

	5-HT1a	5-HT6	5-HT7	5-HT2a	5-HT2b	5-HT2c
baseline	0.944	0.969	0.943	0.930	0.897	0.928
tokenization	0.944	0.969	0.944	0.931	0.894	0.929
context	0.926	0.963	0.929	0.914	0.879	0.903
short paths	0.926	0.958	0.925	0.909	0.877	0.900
simplification	0.926	0.959	0.930	0.916	0.885	0.915
new SMILES	**0.960**	**0.974**	**0.952**	**0.952**	**0.912**	**0.939**

Table 5: Mean MCC score of 6-gram model in active and inactive compounds classification

	5-HT1a	5-HT6	5-HT7	5-HT2a	5-HT2b	5-HT2c
baseline	0.611	0.737	0.671	0.657	0.562	0.674
tokenization	0.614	0.733	0.668	0.656	**0.564**	0.676
context	0.591	0.731	0.633	0.641	0.553	0.655
short paths	0.601	**0.759**	0.650	0.649	0.542	0.665
simplification	0.593	0.720	**0.672**	0.637	0.552	0.653
new SMILES	**0.621**	0.749	0.666	**0.657**	0.559	**0.697**

4.2 N-gram language model

In this section we verify the usefulness of SMILES modifications proposed in Section 3.3. The goal of experiments will be to make the n-gram model extract from SMILES as much information relevant to bioactivity as possible.

We use a perplexity classifier (1) applied on these models. To refer the particular types of modifications, we use the names given in Section 3.3. In experiments, the KenLM implementation of n-gram model is used [5]. Both 3 and 6-grams models were tested, however 6-gram model performed much better in all of the experiments and will be used to present the results.

It is clear from Tables 4 and 5 that the tokenization made the results slightly better than the baseline. The experiments reveal the interesting thing that addition of the context does not help the bioactivity prediction. It may be due to a fact that during a ligand to target protein connection, branches are much more important than chemical compounds center. In above approach however, it is centers of a compound that are being multiplied and result in adding noise to the representation. We also observed a slight deterioration if the classification results when the short paths or the simplification of SMILES was considered. On the other hand, the creation of new SMILES worked well in case of active and ZINC compounds. With active and inactive compounds, the results are not so clear but in general it is the best approach developed.

Table 6: Representations for 5-HT1a receptor

				nonzeros	
		dataset size	length	mean	density
active + inactive	3-gram	5433	1975	99,71	0,050
	6-gram	5433	43562	308,51	0,007
	KRFP	5433	4860	64,08	0,013
active + ZINC	3-gram	43760	2599	103,72	0,040
	6-gram	43760	103933	318,19	0,003
	KRFP	43760	4860	62,38	0,013

4.3 Bag-of-n-grams

In order to enable natural language processing with standard methods of machine learning, one has to create a vector representation. The simplest representation of this kind is called bag-of-words and its idea is to create a vector of length N where every position shows a number of occurrence of a word from a dictionary. Dictionary can be extracted from a training set. This representation can be extended to the case of n-grams (bag-of-n-grams), where every position of a vector corresponds to occurrence of a given n-gram. This representation will be used in the present experiment.

Modifications introduced previously should also have a positive effect on a bag-of-n-grams representation build upon modified and tokenized SMILES. A popular Klekota-Roth fingerprint (KRFP) will be used to compare the results [8]. It is based on counting substructures which were designed to increase the bioactivity information. Table 6 shows statistics of created representations. KRFP has a constant length of 4860 features while between 3 and 6-gram representations one can observe a huge difference due to an exponential grow of n-grams. However, our representation scales well with the amount of data and for active and ZINC compounds, which set is about 8 times bigger than active and inactive compounds, the representation is only about twice longer.

The classification of vector representations was then solved with Support Vector Machine (SVM) [2] which is a reliable model of the machine learning. When using SVM, one has to choose a kernel and the most important parameter of SVM which is the regularization value C. The kernel chosen is radial basis function, while regularization was fitted with cross-validation method after performing features standardization by removing the mean and scaling to unit variance. The parameter space searched was 10^x for $x \in \{-3, -2, \cdots, 4\}$ and for all the data sets and representations, the best C value equals 10^3. Because the classification problem comes from mapping the activation function into a set of labels {-1, 1}, one can also treat this problem as a regression. Both approaches were tested with a use of SVM implementation from scikit-learn library [10]. Answers $\hat{y}_i$ of regression models are mapped into the set of labels with a function:

$$f(\hat{y}) = \begin{cases} 0 \text{ if } \hat{y} \leq 0.5 \\ 1 \text{ if } \hat{y} > 0.5 \end{cases}$$

Table 7: Mean MCC score of vector classification for active and ZINC compounds.

		5-HT1a	5-HT6	5-HT7	5-HT2a	5-HT2b	5-HT2c
	6-gram model	0.960	0.974	0.952	0.952	0.912	0.943
6-gram	SVM cls.	0.983	0.987	**0.977**	**0.975**	0.944	**0.970**
	SVM rgs.	**0.984**	**0.988**	**0.977**	**0.975**	**0.947**	**0.970**
KRFP	SVM cls.	0.971	0.975	0.956	0.953	0.892	0.943
	SVM rgs.	0.969	0.946	0.955	0.913	0.887	0.918

Table 8: Mean MCC score of vector classification for active and inactive compounds.

		5-HT1a	5-HT6	5-HT7	5-HT2a	5-HT2b	5-HT2c
	6-gram model	0.621	0.749	0.666	0.657	0.559	0.697
6-gram	SVM cls.	0.689	**0.780**	0.701	0.711	**0.589**	**0.703**
	SVM rgs.	**0.695**	0.774	0.686	0.695	0.576	0.697
KRFP	SVM cls.	0.655	0.726	0.733	0.722	0.529	0.677
	SVM rgs.	0.618	0.741	**0.744**	**0.727**	0.529	0.678

Tables 7 and 8 show results of classification for sets containing inactive and ZINC compounds respectively. The 'cls.' and 'rgs.' annotations stand for classification and regression approaches. First of all, the bag-of-n-grams classification gives much better results than n-gram model. SVM regression approach works better than the classification. In general, it achieves better results with 6-gram representation than with KRFP fingerprint (two exceptions are 5-HT7 and 5-HT2a receptors in the case of actives-inactives separation).

To have more detailed analysis of the classification, the results for receptors 5-HT1a and 5-HT6 (actives and inactives) are shown in Figure 2. For all of the classification models, 6-gram representation works better than KRFP and allows the SVM regression to achieve a high mean MCC scores equal to 0.7 and 0.78.

5 Conclusion

In this paper, the application of NLP methods to bioactivity prediction was presented. The best modifications were chosen during experimental analysis which shows that virtual screening with a use of textual representation was successful.

In case of the n-gram model, a general tokenization approach and new SMILES generation algorithm were designed. This lead to a creation of bag-of-n-grams representation which is a big success of this work. It allows SVM to achieve higher scores than when using KRFP . Considering the fact that in comparison to KRFP in order to create a bag-of-n-grams representation no chemical domain knowledge is required, it proves high capabilities of this representation.

Open door are also left for further study on SMILES. One of the biggest improvements was new SMILES generation. It would be definitely worth to find

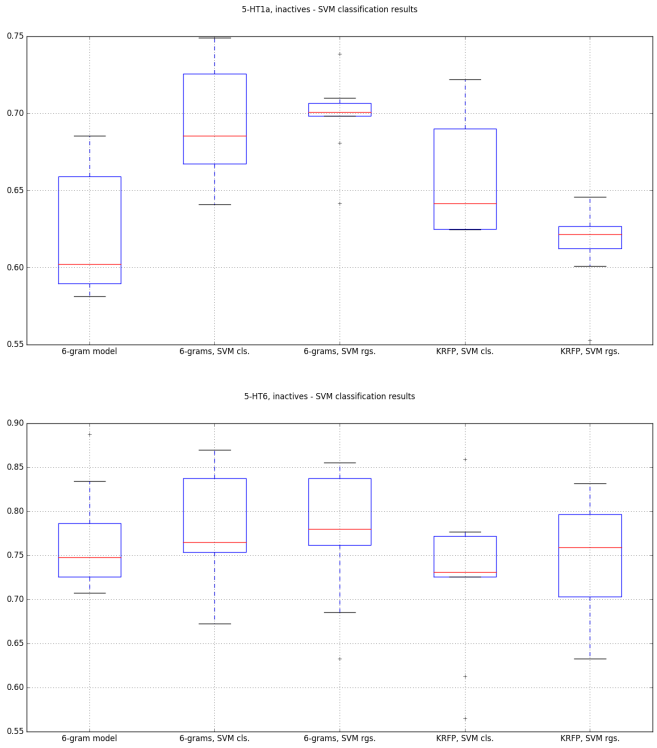


Fig. 2: Classification results of active and inactive compounds for 5-HT1a and 5-HT6, respectively

an optimum between results improvement and time complexity due to a new SMILES generation. In case of bag-of-n-grams representation, it is a common approach in NLP to reduce its dimensionality by removing from a dictionary words which occur in too many or too few documents, as they may be not relevant to a problem. Similar techniques based on n-grams frequency could be applied here which would result in reducing the dimensionality of the representation. Lastly, satisfactory results of bag-of-n-grams representation and n-gram model which works directly on SMILES representation lead to a conclusion that learning compounds representation directly from SMILES with deep learning methods could be a promising approach to undertake.

Acknowledgement

This research was partially supported by the National Science Centre (Poland) grant no. 2014/13/N/ST6/01832 and grant no. 2014/13/B/ST6/01792.

References

1. Brown, P.F.: Class-based n-gram models of natural language. Computational Linguistics 18, 467–479 (1992)
2. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20, 273297 (1995)
3. Daylight Chemical Information Systems: Daylight (2008), `http://www.daylight.com`
4. Gaulton, A.: ChEMBL: a large-scale bioactivity database for drug discovery. Nucleic Acids Research 40, D1100–D1107 (2016)
5. Heafield, K.: KenLM: faster and smaller language model queries. WMT '11 Proceedings of the Sixth Workshop on Statistical Machine Translation pp. 187–197 (2011)
6. Irwin, J.J., Shoichet, B.K.: ZINC - A free database of commercially available compounds for virtual screening. J. Chem. Inf. Model. 45, 177–182 (2005)
7. Jurafsky, D., Martin, J.H.: Speech and Language Processing. Prentice Hall, Upper Saddle River, New Jersey 07458, 2nd edition edn. (2008)
8. Klekota, J., Roth, F.P.: Chemical substructures that enrich for biological activity. Bioinformatics 24, 2518–2525 (2008)
9. Matthews, B.W.: Comparison of the predicted and observed secondary structure of T4 phage lysozyme. Biochimica et Biophysica Acta (BBA) - Protein Structure 405, 442–451 (1975)
10. 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, 2825–2830 (2011)
11. Raevsky, O.A.: Molecular structure descriptors in the computer-aided design of biologically active compounds. Russ. Chem. Rev. 68, 505–524 (1999)
12. Shoichet, B.K.: Virtual screening of chemical libraries. Nature 432, 862865 (2004)
13. Sousa, S.F., Fernandes, P.A., Ramos, M.J.: Proteinligand docking: Current status and future challenges. Proteins: Structure, Function and Bioinformatics 65, 15–26 (2006)
14. Vidal, D., Thormann, M., Pons, M.: LINGO, an efficient holographic text based method to calculate biophysical properties and intermolecular similarities. J. Chem. Inf. Model. 45, 386393 (2005)
15. Warszycki, D., Mordalski, S., Kristiansen, K., Kafel, R.and Sylte, I.C.Z., Bojarski, A.J.: A linear combination of pharmacophore hypotheses as a new tool in search of new active compounds–an application for 5-HT1A receptor ligands. PloS ONE 8(12), e84510 (2013)
16. Worachartcheewan, A., Mandi, P., Prachayasittikul, V., Toropova, A., Toropov, A., Nantasenamat, C.: Large-scale QSAR study of aromatase inhibitors using SMILES-based descriptors. Chemometrics and Intelligent Laboratory Systems 138, 120–126 (2014)
17. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 334–342. ACM (2001)

Evaluation of Fusion Approaches in Large-scale Bio-annotation Setting*

Vedrana Vidulin[1§], Maria Brbić[1] Fran Supek[1,2], and Tomislav Šmuc[1]

[1] Ruđer Boškovic Institute, Bijenička cesta 54, 10000 Zagreb, Croatia
[§]vedrana.vidulin@irb.hr,
[2] EMBL/CRG Systems Biology Unit, Centre for Genomic Regulation, Dr. Aiguader 88, 08003 Barcelona, Spain

Abstract. In this work we compare different information fusion approaches in the context of large-scale multi-label classification problems, typical today in bio-domains: early fusion, late fusion and hybrid fusion approach. The experiments are performed on two novel large-scale classification datasets for gene function prediction and prokaryotic phenotype prediction. Both datasets are based on descriptors coming from a number of different representations of biological entities. The results reveal that the fusion approaches exploiting complementarity are best suited for difficult annoation problems featured in complex datasets from bio-domains for which individual classifiers perform well only locally.

Keywords: information fusion, ensemble classifier design, diversity, annotation, genomics

1 Introduction

Combining classifiers for information fusion [1], [2], [3] is an important topic in the era of information overload present in the majority of technological and scientific domains, from multimedia systems to life-sciences. In particular, in molecular and systems biology multitude of omics approaches are used in order to explain complex roles and associations between cell constituents, and machine learning methods are used as tools for knowledge discovery, either for annotating entities with typically non-exclusive roles (e.g. protein function prediction), or in searching for the important patterns, interactions and representations of entities for the particular problem at hand. The importance of machine learning is best seen through a number of predictive challenges, some of which are repetitive regular events such as CASP and CAFA. Here are some features accompanying discovery problems in genomics:

1. Predictive performance is typically optimized to maximize predictions at certain precision threshold, and the final model can refrain from making a

* The first two authors should be regarded as joint first authors.

decision when certain precision is not guaranteed. This predictive setting is actually close to that of information retrieval in which metrics uch as F-measure, Area Under Precision Recall Curve (AUPRC) or recall@precision are used.
2. Predictive problems often involve multi-label classification or structured output prediction, where the output structure can be in the form of taxonomy or ontology showing the need for tools that exploit relations between labels.
3. Complex problems in this domain often include a number of different representations drawn from at least partially independent views. Overall dimensionality of these representations, in terms of the number of features, their relevance for the target and their mutual independence are the major challenges with respect to the annotation problems. Therefore, it is important to have efficient and reliable methods that can exploit individual contexts and their interactions.

The last feature can be handled using classifier fusion approaches that exploit complementarity on a feature or classifier level. In this work we evaluate different fusion approaches in the context of two large scale predictive problems: gene function prediction (GFP) and prokaryotic phenotype prediction (PP), addressing all of the specific requirements mentioned above. Evaluation methodology provide us a detailed picture of driving mechanisms behind the predictive performance of different approaches. The main contribution of this work are insights related to the importance of fusion mechanisms and their capability to exploit advantages of different representations used to describe entities of underlying problems.

2 Related Work

Multi-classifier systems, classifier ensembles and meta-learning approaches have been important topics in the machine learning field for several decades. Information fusion is closely related to these topics. The evolution and adaptation of the well known techniques and approaches in this field is of high importance in the context of complex, distributed and streaming data environments. Concepts of fusion and multi-classifier combination techniques are covered extensively in the reviews [1], [3], [4], and have been important topics at the relevant conferences in the field of data mining and machine learning.

Different aspects are important when optimizing combination of modalities, but one most considered is the level at which fusion of representations is performed: in early fusion predictive model is built using all feature sets or representations together to make a single decision model; contrary to that, late fusion approaches deal with combining models learnt separately on different features sets or representations. Each of these approaches has their own advantages. Early fusion approach can make use of interactions between basic features from different representations, improving both understanding and predictive performance, while late fusion approaches can combine outputs of different classifiers exploiting the same representation on a decision level. Moreover, combining individual

decisions from possibly complex individual representations offers scalability and allows us to use the most suitable methods for analyzing each single representation, thus providing more flexibility than the early fusion.

An obvious extension that combines the strengths of both approaches is the hybrid fusion, which in the simplest variant, combines early fusion model with the individual representation models using late fusion approaches.

In the majority of analysis reported in literature empirical results demonstrate better performance of late fusion methods in comparison to early fusion. Late fusion methods exploit additional level of complexity through another layer of decision making, i.e. combination of multiple baseline (feature subset) models. Although the power of ensembles approaches has been both theoretically and empirically proven to be related to at least partial independence of models, the strong and definite connection between different ensemble diversity and accuracy of the ensemble seems to be lacking [5].

Recently, in paper by [6] the authors illustrate the connection between improvement of ensemble classifiers based on the relative independence of their false-positive prediction patterns. In their study they used recall at predefined precision level as a measure to compare classifiers' performances. This measure is very well suited for information retrieval, but it is also aligned with the knowledge discovery tasks. We show in this work that the power of late fusion approaches is most effective in the setting where strong predictive performance of individual classifiers (or descriptor sets) is of very local character. Furthermore, we show that in this setting that there is a strong correlation between recall at predefined precision threshold and the diversity of classifiers calculated using their individual performances.

3 Fusion approaches

In the early fusion setting feature sets from the individual representations are all joined into one dataset from which a single classifier is constructed. This type of fusion should generally better exploit interactions between features from different representations.

On the other hand, late fusion is performed by constructing a separate classifier for each of the individual representations and then fusing their predictions. Our pipeline implements five different late fusion approaches: one vote, two votes, three votes, consensus [7] and weighted voting [8].

Let $C = (c_i)_{i=1}^{N}$ be a sequence of confidence scores of N individual classifiers and let $S = (s_i)_{i=1}^{N} = sort(C)$ denote a sequence arranged in ascending order i.e. $s_i \leq s_{i+1}$. The one/two/three votes approaches calculate the fused confidence scored of the class y_j:

$$c_{1vote}(y_j) = s_N(y_j); \quad c_{2votes}(y_j) = s_{N-1}(y_j); \quad c_{3votes}(y_j) = s_{N-2}(y_j) \tag{1}$$

Consensus score of a label y_j is calculated using the following formula:

$$c_{\text{cons}}(y_j) = 1 - \prod_{i=1}^{N}(1 - c_i(y_j)) \tag{2}$$

The weighted voting score of a label y_j is calculated as follows:

$$c_{\text{wv}}(y_j) = \sum_{i=1}^{N} w_i c_i(y_j), \tag{3}$$

where w_i denotes weight assigned to the classifer i and:

$$\sum_{i=1}^{N} w_i = 1 \tag{4}$$

We calculated weight as Area Under Precision-Recall Curve (AUPRC) estimated in the cross-validation setting and normalized to sum to 1.

Finally, hybrid fusion performs late fusion on the predictions from the individual classifiers and the early fusion classifier.

4 Experimental Setup

In this section we start by introducing problems and respective representations used in this work. Further, we explain the general scheme of experiments designed to give us an answer to our main question: which fusion approaches work best for the types of problems we solve in this work, while also providing intuition when and why. Next, we present the experimental methodology and discuss the evaluation measures used in the experiments.

4.1 Datasets

Gene function prediction. Five datasets represent GFP methods based on semantically distinct feature sets (Table 1, details in [9]). All datasets have common set of 15,308 instances representing eggNOG clusters of genes [10] and are labeled with 935 gene functions from Gene Ontology (GO) [11].

Phenotype prediction. Phenotype prediction datasets are constructed using six different representations (Table 2, details in [12]). Each representation represents one dataset with a set of 703 instances and 72 labels. Each instance corresponds to one prokaryotic organism labelled with a set of phenotypic traits.

4.2 Methodology

The input to our computational pipeline is the group of datasets that describe the same concept but with distinct feature sets. First, in order to reduce the dimensionality of the feature sets we applied principal component analysis (PCA) on each of the datasets separately and retained principal components (PC) that explain 90% of the variance. We divided our data into training (consisting of 2/3 learning instances) and test sets (1/3 instances) using stratified sampling.

Table 1. Gene function prediction datasets.

Dataset	# features	# PC	Features description
Phyletic profiles (PP)	2071	352	Features are genomes and feature values represent presence/absence of cluster member genes in genomes.
Empirical kernel map (EKM)	8447	1552	Features are gene clusters and feature values represent minimal distance between gene sequence pairs, where one sequence is from instance and another from feature cluster. Distance is measured as e-value.
Conserved gene neighborhood (CGN)	5891	1411	Features are gene clusters and feature values represent average log-distance (in nucleotides) between genes from instance and feature cluster pairs averaged over genomes.
Translation efficiency profiles (TEP)	2071	1449	Features are genomes and values represent maximal predicted level of cluster member genes expression.
Biophysical and protein sequence properties (BPS)	1170	296	Features represent various gene sequence statistics as described in [13]. Statistics are computed on a gene level and averaged between cluster member genes.

Table 2. Phenotype prediction datasets.

Dataset	# features	# PC	Features description
Text-mining	95663	438	Bag-of-words representations of documents describing bacterial/archael species collected from the scientific literature and the broader World Wide Web
Amino acid content	420	3	Amino acid and di-amino acid frequencies of a proteome
Pairwise co-occurrences	1235	33	Pairwise co-occurrences of species in metagenomes [14]
Phyletic profiles	80576	393	Presence/absence of the clusters of orthologous groups (COGs) of proteins
Conserved gene neighborhood	44850	366	Log pairwise chromosomal distance in nucleotides between pairs of 300 frequently occurring COG gene families
Translation efficiency profiles	990	263	Codon usage biases of COG gene families across 606 genomes, measured using the MILC method [15]

In the early fusion setting, all feature sets were combined and given as an input to a single early fusion model (EF). In the late fusion setting, individual models are constructed for each of the feature sets (FS) separately. In order to access performance of indvidual classifiers necessary for the weighted voting approach, we used a cross-validation setting. The hybrid fusion model was built in the same way as the late fusion one, but also using the early fusion model as a classifier. Finally, all models are deployed on the test set instances. Figure 4.2 gives a general outline of computational experiments performed on each of the problems discussed in this work.

Governed by the principle of the best trade-off of accuracy, efficiency and robustness, we used random forests [16] as a classification algorithm. Random forests have been used on biological data and the evaluation shows that they are able to produce state-of-the-art annotation results [9] [23] [24]. Since label spaces of gene function prediction (GFP) and phenotype prediction (PP) problems have different properties, we used different versions of the algorithm. In order to exploit hierarchical structure of the labels in the GFP problem, we used random forests version of CLUS-HMC [17], [18]. CLUS-HMC is the algorithm for hierarchical multilabel classification based on the framework of Predictive Clustering Trees [19]. CLUS-HMC was run with the default parameters, except

for these settings: decision tree pre-pruning was used to prevent the algorithm to form a leaf node when the number of instances in the node is <5; forest size was set to 200 trees; size of a feature subset for random forests was set to square root of the total number of features. For the PP problem where the multi-label target side is flat, we used FastRF, a fast and efficient implementation of the random forest algorithm in WEKA [20]. The number of trees was set to 500. For this setting we applied binary relevance method that corresponds to learning one classifier for each label separately. This leads to a notable difference from CLUS-HMC which is able to produce one global model over the whole hierarchy of labels. Finally, it is important to note that diversity of individual models in the fusion ensemble is the consequence of the use of different feature sets given in Tables 1 and 2 in building individual classifiers of the fusion ensemble.

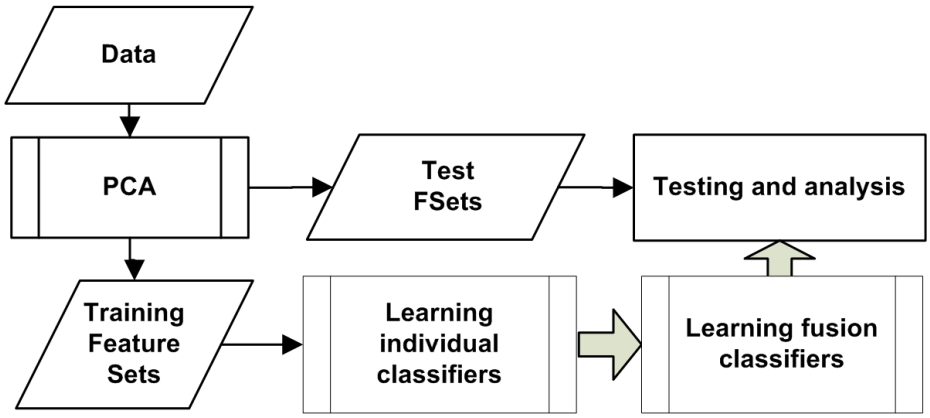


Fig. 1. General schema of computational experiments performed in this work.

4.3 Evaluation

Predictive performance of different fusion approaches was measured on a separate test set, composed of one third of the instances. We used two performance measures that rely on a precision-recall curve: (i) Area Under Precision Recall Curve (AUPRC), and (ii) recall at some predefined precision level (R@P) which represents the part of the precision-recall curve at some (high) precision level. The latter measure was used to emphasize the importance of having predictions with high level of precision which is especially important in the context of annotation for omics data.

A single PR curve was computed by averaging label-specific curves, which corresponds to the averaging procedure known as macro-averaging in a multi-label machine learning setting. It is common to report micro beside macro-averaged measures, but micro-averaging is not appropriate in the setup with

highly unbalanced classes where interesting classes are typically those with the least positive examples. For example, a specific label predicted from the bottom of the GO ontology is more interesting for a domain expert than the label predicted higher in the ontology. In such settings, micro-averaging would equally weight examples and thus, averaged performance scores would mainly represent the performance on less interesting general labels. In contrast, macro-averaging equally weights labels, enabling interesting specific classes to influence the average performance score.

Statistical comparison of fusion approaches was performed by using the corrected Friedman test and the post-hoc Nemenyi test [25]. The Friedman test is a non-parametric test for multiple hypothesis testing. For each label, this test ranks the fusion approaches according to AUPRCs measured for this specific label. The best approach is ranked as the first and in the case of ties an average rank is assigned. The test compares approaches by comparing ranks averaged over all labels and calculates Friedman statistic distributed according to the χ^2_F with $k-1$ degrees of freedom, k being the number of fusion approaches. In cases where at least one approach performed significantly different than the rest we performed Nemenyi test that shows where that difference lies. We present the results of Nemenyi test using average ranks diagrams [25].

Diversity in classifier ensembles is measured as a disagreement between classifiers in terms of correct/incorrect predictions. Different pairwise diversity measures have been proposed in literature [5], while the averaged statistic is typically calculated by averaging over all pairs of classifiers. As the most appropriate measure to characterize diversity in our setting, we chose the disagreement measure [21] defined as the ratio between the number of predictions on which classifiers disagree and the total number of predictions:

$$Disagreement(i,j) = \frac{N^{10} + N^{01}}{N^{11} + N^{10} + N^{01} + N^{00}}, \tag{5}$$

where N^{10} denotes the number of predictions on which classifer i is correct and classifier j is incorrect; the same applies vice versa. Since we deal with highly unbalanced classes, the diversity measure was calculated only on positive examples.

In order to investigate the complementarity in the context of fusion performance in more detail, we assessed the performance of fusion approaches in the respect to the generality of labels. The generality was measured using information content (IC) [22] computed from label frequency. Higher IC is related to the more specific labels which are usually more difficult to predict, but more valuable to the domain experts.

4.4 Experiments

In order to investigate which of the fusion approaches performs best and how is the performance related to the diversity of individual classifiers, we structured our experiments in the following manner: (i) we computed macro-averaged performance measures for each of the individual classifiers and fusion approaches

across three levels of difficulty; (ii) we looked at the relationship between difficulty and diversity of individual classifiers for EF, LF and HF approaches; and (iii) to examine complementarity of EF, LF and HF we measured improvement over a baseline. Since in our experiments LF-three votes approach is a proxy of majority voting and a solution that fosters consensus of classifiers rather than complementarity, we used it as a baseline.

5 Results and discussion

Performances of individual classifiers and different fusion approaches for GFP and PP problem are shown in Table 3 and Table 4, respectively. Performances are measured using average AUPRC and recall at 70% precision threshold. Results are structured to show performance for different label generality levels defined by problem the specific IC intervals (general, medium and specific; equal number of labels per bin). One obvious difference between the GFP and PP problems is that the average performance of individual classifiers for the GFP problem is much lower than for the PP problem. This is not unexpected since the GFP annotation is more difficult problem with more than an order of magnitude larger label space. Average results in Tables 3 and Table 4 are accompanied with statistical comparison of fusion approaches through average ranks diagrams across label generality levels shown in Figure 2.

More detailed analysis reveals different trends between fusion approaches across label generality levels. For the GFP problem, weighted voting approach and consensus scheme are clear winners over all generality levels, while late fusion and hybrid fusion approaches are consistently higher than the early fusion approach. The situation is different for the PP problem: early fusion gives slightly better results than late fusion and hybrid fusion approaches. For the general and medium-specific labels of the PP problem, three votes fusion seems to be a better choice than the weighted voting or consensus scheme, albeit not significantly. Also, differences in ranking between one and three vote schemes for the two problems suggest that the improvements through fusion is obtained in a different manner: by exploiting complementarity in GFP, and consensus in PP problem. In relative terms, improvements obtained by fusion approaches are much more significant for the GFP problem, as are also the differences between fusion schemes, as depicted in Figure 2.

5.1 Classifier diversity and performance of fusion approaches

The relationship between diversity and performance of different fusion approaches is shown on Figure 3. For both problems, higher diversity seems to be clearly related with higher performance, similarly for AUPRC and R@P measure. It seems that diversity spreads to larger values for GFP than for PP problem. Before making further inferences about the nature of the relationship between performance of classifiers and diversity we need to look into basic characteristics

Table 3. Gene function prediction

		GENERAL		MEDIUM-SPECIF		SPECIFIC	
		AUPRC	RC@0.7PR	AUPRC	RC@0.7PR	AUPRC	RC@0.7PR
Late fusion	Early fusion	0.543	0.417	0.196	0.192	0.081	0.155
	One vote	0.573	0.440	0.238	0.201	0.113	0.149
	Two votes	0.453	0.335	0.176	0.170	0.103	0.147
	Three votes	0.241	0.127	0.118	0.100	0.044	0.096
	Consensus	0.571	0.447	0.252	0.227	0.131	0.169
	Weighted voting	0.590	0.455	0.257	0.228	0.122	0.165
Hybrid fusion	One vote	0.575	0.439	0.245	0.201	0.117	0.151
	Two votes	0.561	0.434	0.251	0.216	0.131	0.175
	Three votes	0.485	0.373	0.210	0.200	0.107	0.165
	Consensus	0.582	0.449	0.275	0.233	0.153	0.187
	Weighted voting	0.580	0.453	0.276	0.235	0.134	0.186
Individual	PP	0.136	0.048	0.057	0.065	0.007	0.065
	EKM	0.529	0.392	0.144	0.163	0.030	0.102
	CGN	0.123	0.046	0.045	0.062	0.008	0.075
	TEP	0.083	0.026	0.018	0.044	0.004	0.026
	BPS	0.411	0.264	0.052	0.089	0.023	0.104

Table 4. Phenotype prediction

		GENERAL		MEDIUM-SPECIF		SPECIFIC	
		AUPRC	RC@0.7PR	AUPRC	RC@0.7PR	AUPRC	RC@0.7PR
Late fusion	Early fusion	0.836	0.781	0.610	0.465	0.439	0.309
	One vote	0.732	0.622	0.452	0.227	0.373	0.227
	Two votes	0.796	0.710	0.562	0.413	0.434	0.312
	Three votes	0.820	0.774	0.571	0.430	0.425	0.333
	Consensus	0.799	0.726	0.551	0.383	0.406	0.265
	Weighted voting	0.820	0.762	0.584	0.436	0.430	0.296
Hybrid fusion	One vote	0.732	0.623	0.452	0.227	0.373	0.227
	Two votes	0.802	0.711	0.581	0.439	0.440	0.312
	Three votes	0.831	0.787	0.595	0.449	0.439	0.319
	Consensus	0.808	0.736	0.566	0.399	0.416	0.279
	Weighted voting	0.827	0.772	0.593	0.445	0.446	0.317
Individual	TM	0.791	0.715	0.575	0.428	0.388	0.262
	AAC	0.679	0.540	0.339	0.153	0.270	0.126
	PC	0.611	0.408	0.253	0.119	0.132	0.058
	CGN	0.759	0.692	0.499	0.361	0.429	0.336
	PP	0.781	0.696	0.504	0.347	0.414	0.342
	TEP	0.752	0.684	0.418	0.240	0.259	0.177

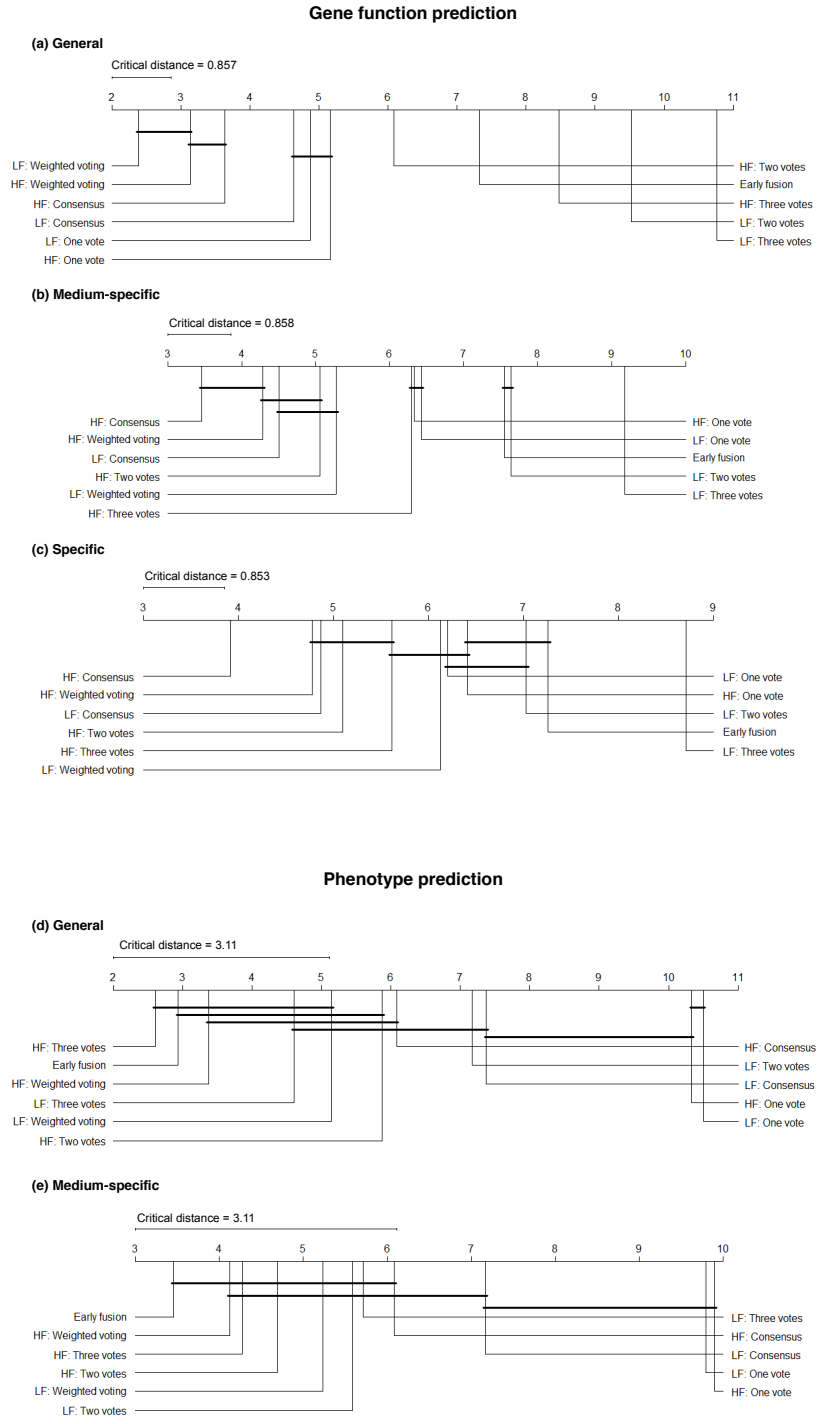


Fig. 2. Average ranks diagrams comparing the performance of fusion approaches over labels belonging to different levels of generality. LF stands for late fusion and HF for hybrid fusion. The numbers on the axis represent ranks and the best ranking approaches are at the leftmost side of the diagram. The approaches that do not differ by less than the critical distance for a significance level of 0.05 are connected with a line.

of the used diversity measure. Assuming independence between individual classifier outputs, higher values of diversity can be expected for ensembles of medium performing classifiers or ensembles with very diverse range of performances. If we assume that we have homogeneous ensemble of classifiers with high recall values, we can expect low diversity measures. The same is true if all classifiers have low recall values since in that case low diversity is driven by high values of N^{00} in Eq.(5). Main correlation trend between diversity and performance for the GFP problem seems to be driven by ensembles of low performing - low diversity classifiers on one hand and mixtures of ensembles with low and high performing classifiers (range of diversity 0.3-0.6).

The same mixture of cases seems to be driving the correlation behind performance measures and diversity in case of the PP data, with one notable difference, the cases with low diversity and very high score. These cases together with the rest of general labels exhibit no correlation between diversity and performance scores.

5.2 Relationship between diversity and generality of labels

In principle, when dividing the label space, we assumed that more general labels will be easier to learn and this holds on average for both problems. However, for GFP problem in particular, diversity seems to play important role for general labels as well. At the first glance there is a counter-intuitive result for the GFP problem: the correlation of diversity and performance is somewhat larger for general categories than for the specific ones. However, there is a considerable number of specific categories that are practically not learnable (R@P=0) at all, which reduces correlation over learnable specific categories. Low learnability seems also to be the explanation for low scores of significant part of general and medium specific GFP labels, which shows that available GFP feature sets are still not fit enough for proper learning of significant part of Gene Ontology graph.

To the contrast, for the PP problem there seems to be no significant correlation between diversity and performance for general category of labels. However, if we compare IC intervals of GFP and PP general category of labels, it can be seen that PP general labels are much more frequent, and therefore also more easily learnable. Overall fitness of feature sets for general PP labels is confirmed through low diversity and very high performance scores for general PP labels. Figure 4 shows increase of the performance scores of fusion approaches over the baseline three votes approach, regarded in our experiment as a proxy for majority voting. Majority voting is the approach that exploits consensus rather than the complementarity of classifiers. On the other hand, weighted voting approach exploits both complementarity and consensus of classifiers. Notable difference between two problems is that improvement for GFP problem is on average significant over all label generality levels, while for the PP problem difference between fusion approaches is practically negligible. Also, for the medium and specific labels of the GFP problem, there is a consistent difference between early, late and hybrid fusion with early fusion as the weakest and hybrid fusion as the

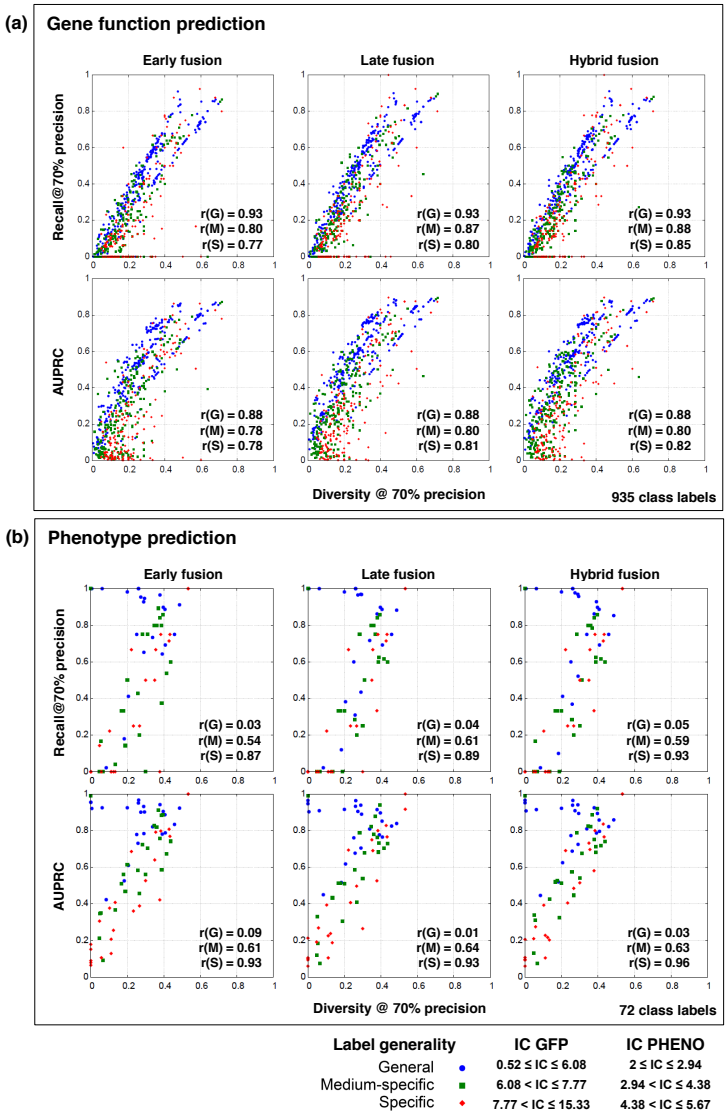


Fig. 3. Diversity of individual classifiers is correlated with the predictive performance of fusion approaches. Late and hybrid fusion are both based on weighted voting approach. r stands for Pearson correlation coefficient, G for general labels, M for medium specific and S for specific labels. IC stands for information content.

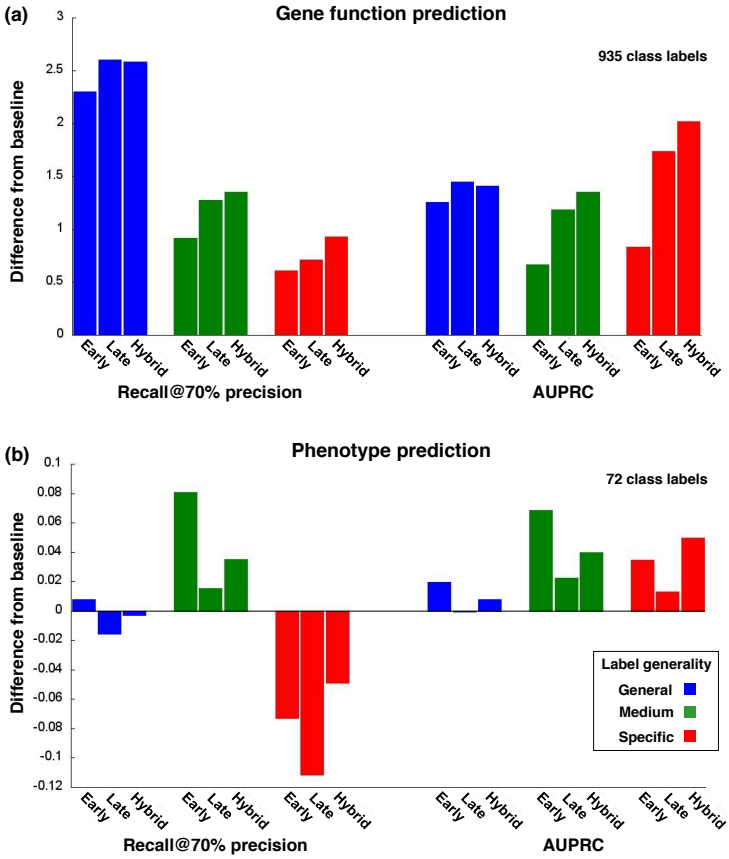


Fig. 4. Differences between fusion approaches and three votes baseline. Late and hybrid fusion results are those for Weighted voting approach.

strongest performer. The logical conclusion is that fusion approaches exploiting complementarity are the best choice for difficult annotation problems, for which individual classifiers perform well only locally. Such cases are characterized by strong correlation between diversity and performance of fused ensembles.

6 Conclusions

This work revisits the problem of classifier fusion with intention to provide new insight into relationship of classifier diversity and performance of classifier fusion approaches. We have used simple fusion approaches that foster complementarity between classifiers, and two performance measures well suited for discovery setting of biological annotation. Although we have performed the analysis on just two datasets, we believe that their complexity and characteristics provide

enough grounds to conclude that diversity of ensembles is indeed strongly related to improved performance of fusion approaches exploiting complementarity. This is the case when individual classifiers exhibit strong performances only locally (i.e. only for a subset of labels). This conclusion slightly departs from the previous findings [5]. In our future efforts we plan to investigate relationship between diversity and performance of fused ensembles in more detail, over more datasets, using larger number of classifiers, and more complex fusion approaches.

Acknowledgements. This work was supported by the European Commission through the projects MAESTRA [ICT-2013-612944], InnoMol [FP7-REGPOT-2012-2013-1-316289] and project Machine Learning Algorithms for Insightful Analysis of Complex Data Structures of the Croatian Science Foundation [Pr. no. 9623].

References

1. Kuncheva, L.: Combining Pattern Classifiers: Methods and Algorithms.Wiley Interscience, 2004
2. Tulyakov, S., Jaeger, S.,Govindaraju, V., Doermann, D.: Review of Classifier Combination Methods, Studies in Computational Intelligence, Vol 90, pp. 361-386, 2008
3. Wozniak, M.,Graña, M.,Corchado, E.: A survey of multiple classifier systems as hybrid systems. Information Fusion 16, pp. 3-17, 2014
4. Tulyakov, S., Jaeger, S.,Govindaraju, V., Doermann, D. : Review of Classifier Combination Methods. Chapter Machine Learning in Document Analysis and Recognition Vol 90, Studies in Computational Intelligence, pp. 361-386, 2008
5. Kuncheva, L.,Whitaker, C.: Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy Machine Learning, Vol 51, pp. 181-207, 2003
6. Madani, O., Georg, M., Ross, D. A.: On using nearly-independent feature families for high precision and confidence. Machine Learning , V 92 (2), pp. 457-477, 2013
7. Piovesan, D., Giollo, M., Leonardi, E., Ferrari, C., Tosatto, S. C.: INGA: protein function prediction combining interaction networks, domain assignments and sequence similarity. Nucleic acids research, 43(W1), 2015
8. Zhou, Z. H.: Ensemble methods: foundations and algorithms. CRC press, 2012
9. Vidulin,V., Šmuc, T., Supek, F.: Extensive complementarity between gene function prediction methods. Bioinformatics, 2016, doi:10.1093/bioinformatics/btw532
10. Powell, S., Forslund, K., Szklarczyk, D., Trachana, K., Roth, A., Huerta-Cepas, J., ... , Bork, P.: eggNOG v4.0: nested orthology inference across 3686 organisms. Nucleic acids research, 2013
11. Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., ..., Sherlock, G.: Gene Ontology: tool for the unification of biology. Nature genetics, 25(1), pp. 25-29, 2000
12. Brbic, M., Piskorec, M., Vidulin, V., Krisko, A., Supek, F.: The landscape of microbial phenotypic traits and their genetic basis, Submitted, 2016
13. Ofer, D., Linial, M.: ProFET: Feature engineering captures high-level protein functions. Bioinformatics, 2015

14. Chaffron, S., Rehrauer, H., Pernthaler, J. Mering, C. von. A global network of coexisting microbes from environmental and whole-genome sequence data. Genome Research 20, pp. 947âĂŞ959, 2010
15. Supek, F. Vlahovicek, K: Comparison of codon usage measures and their applicability in prediction of microbial gene expressivity. BMC Bioinformatics 6 (182), 2005
16. Breiman, L.: Random forests.Machine Learning, 45 (1), pp. 5-32, 2001
17. Kocev, D., Vens, C., Struyf, J., Dzeroski, S.: Tree ensembles for predicting structured outputs. Pattern Recognition, 46(3), pp. 817-833, 2013
18. Vens, C., Struyf, J. Dzeroski, S., and Blockeel, Hendrik: Decision trees for hierarchical multi-label classification. Machine Learning, 73(2), pp. 185-214, 2008
19. Blockeel, H.: Top-down induction of first order logical decision trees. Ph.D. thesis, Katholieke Universiteit Leuven, Leuven, Belgium, 1998
20. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update; SIGKDD Explorations, 11 (1), 2009
21. Skalak, D.: The sources of increased accuracy for two proposed boosting algorithms. In Proc. American Association for Artificial Intelligence, AAAI-96, Integrating Multiple Learned Models Workshop, 1996
22. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, Morgan Kaufmann, San Francisco, CA, pp. 448-453, 1995
23. Škunca, N., Bošnjak, M., Kriško, A., Panov, P., Džeroski, S., Šmuc, T., Supek, F. (2013). Phyletic profiling with cliques of orthologs is enhanced by signatures of paralogy relationships. PLoS Comput Biol, 9(1), e1002852
24. Schietgat, L., Vens, C., Struyf, J., Blockeel, H., Kocev, D., Džeroski, S. (2010) Predicting gene function using hierarchical multi-label decision tree ensembles, BMC Bioinformatics, 11(2), 1-14
25. Demšar, J. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research, 7:1âĂŞ30, 2006

Mining Imbalanced Medical Streams for Automated Fetal State Assesment

Bartosz Krawczyk[1,2] and Michał Woźniak[2]

[1] Department of Computer Science,
Virginia Commonwealth University,
401 West Main Street, 23284 Richmond, VA, USA.
e-mail: bkrawczyk@vcu.edu
[2] Department of Systems and Computer Networks,
Wrocław University of Science and Technology,
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland.
e-mail: michal.wozniak@pwr.edu.pl

Abstract. Mining data streams is one of the most vital fields in the current era of big data. Continuously arriving data may pose various problems, connected to their volume, variety or velocity. This topic has been rarely discussed in the context of medical data mining. In this paper we address the problem of online fetal state monitoring, where new biomedical signals arrive over time from a hospital ward and medical devices. To cope with the changing nature of the data and patient-specific characteristics. we propose to use online version of Extreme Learning Machine that is enhanced by an efficient drift detector and method to alleviate the bias towards the majority class. We investigate three approaches based on undersampling, oversampling and cost-sensitive adaptation. Additionally, to allow for a rapid updating of the proposed classifier we show how to implement online Extreme Learning Machines with the usage of GPU. The proposed approach allows for a highly efficient mining of high-speed, drifting and imbalanced real-time fetal state monitoring with significant acceleration offered by GPU processing.

Keywords: machine learning, data stream mining, medical data streams, imbalanced classification, online learning, fetal state monitoring

1 Introduction

Contemporary computer systems store and process enormous amounts of data. Current predictions point out that the volume of stored information will be doubling every two years. E-mails, social webs, on-line shopping etc. produce ever-growing data, which may carry valuable hidden information. Therefore, three main issues in big data analytics must be addressed (known as 3Vs: *Volume, Velocity, Variety*): how to efficiently transfer such volumes [7], how to store them and how to extract meaningful knowledge from them [9, 11]. In this work we are mainly focusing on the Velocity, because designing big data analytical

tools must take into consideration that most of data in modern systems arrives continuously [3] as so-called data stream [4]. Furthermore, the nature and characteristics of data, i.e., statistical dependencies characterizing an analyzed data stream, can change. A special analytical model which can cope with and adapt to such non-stationary characteristics is required in such cases [6]. This problem can be connected with the skewed class distributions, leading to scenario when not only properties of object changes but also ratios between class distributions.

Among a plethora of solutions dedicated to data stream mining with concept drift we may distinguish the following approaches: (i) based on external module for detecting shift and drifts in the stream [2], (ii) based on adaptive classifiers using windowing technique [10], (iii) based on online classifiers that are able to process stream sample-by-sample, and (iv) using classifier ensembles.

In this work we propose a real-time decision support system for medical data stream analysis coming from online fetal state monitoring. This is a vital problem in contemporary clinical care, where we monitor the biosignals emitted by fetus and try to assess if there is any incorrectness in them. Such an information is used for guided pregnancy in cases of any difficulties or problems with the fetus. However, the abnormal cases appear much less frequently than normal ones, thus leading to the problem of mining imbalanced data stream.

As our data will arrive continuously in a form of medical data stream and are subject to shifts and drifts over time, we require an adaptive classification model. We propose to use a modification online Extreme Learning Machines that is able to handle imbalanced and drifting data streams. Extreme Learning Machines are selected due to their low computational complexity which makes them suitable for mining high-speed data. However, to achieve even more rapid training and update of our classifier we propose a GPU-based implementation of the online classifier that is able to significantly speed-up the entire process. This makes our classifier suitable for big data stream mining. In order to capture the changes in arriving data we employ an efficient ADWIN drift detector that decides whether to update the current model or discard it and train new one from scraps. To alleviate the problem of skewed distributions we propose three approaches utilizing either data-level or cost-sensitive techniques.

2 Fetal state monitoring

Fetal heart activity is the prominent source of information about fetal well being during delivery. Cardiotocography is a technique of recording the fetal heart rate and uterine contractions, which enables obstetricians to detect fetus with deteriorating status. In case of high-risk patients the fetus is monitored continuously thorough the mothers' stay in the hospital ward. Usually a number of patients is being monitored at the same time, thus leading to need for analyzing information arriving from multiple sources ar the same time. There is a variation in the outputted biosignlas by fetuses that do not point to dangerous state but is caused by natural differences contributed to individual characteristics of living beings. .

Any change in monitored signals must be detected in real-time and handled as soon as possible. Such abnormalities may be a direct threat to the health and life of fetus, as well as to mother's well-being. Therefore, a real-time data mining with high-speed response is crucial for such a system. This is why we propose to develop a fully automatic online decision support system that will be implemented on GPU for fast and precise diagnosis in real-time.

In this paper we use CTU-UHB open database of fetal state signals [8]. It contains 552 recorded biomedical fetus signals taken from real hospital cases. The inlying difficulty is the skewness of data . We need to handle a two-class imbalanced problem, consisting of 506 normal readings and 46 abnormal ones. For the classification purposes we use 33 features extracted from the signals, provided by the authors of the database [8]. Up to our best knowledge this is the first study to consider this dataset as a data stream mining problem.

Hypoxia, with prevalence lying in the region of 0.6% to 3.5%, is considered to be the third most common cause of newborn death.

Due to high number of factors to be taken under consideration decision support systems using machine learning are in high demand.

3 GPU-accelerated extreme learning machines for imbalanced drifting streams

In this section details of the proposed GPU-based Extreme Learning classifiers for imbalanced and drifting data streams will be given.

3.1 Extreme learning machines

Extreme Learning Machines (ELMs) [5] are random-based single-layer feedforward neural networks trained in a randomized manner in order to reduce their computational complexity. In last two decades there was a number of significant developments in the field of neural network training algorithms. However, most of these approaches suffered from the extended computational time required for effective execution and a large number of parameters to be set. ELMs are among emerging trends in neural network learning that aim at alleviating the training complexities with the usage of randomly drawn weights for neurons in the hidden layer. One must note here that despite the emerging popularity of ELMs-based approaches this concept can be actually traced further down in the literature to the proposals of Random Vector Functional Link.

Let us now present the concept of ELMs. Let us assume to have n labeled objects in a d-dimensional feature space and a set of M class labels. A single-layer feedforward neural network with N hidden neurons can be described by the following equation:

$$\mathbf{y} = \sum_{i=1}^{N} \mathbf{B}_i f(\mathbf{w}_i \cdot \mathbf{x} + b_i), \tag{1}$$

where $f()$ represents the activation function, $\mathbf{x}$ is the considered object, $\mathbf{w}_i$ stand for input weights associated with i-th hidden neuron, b_i is the bias of i-th hidden neuron and $\mathbf{B}_i$ are weights assigned to output neurons.

When considering all of n training points we use this equation in a matrix form:

$$\mathbf{Y} = \mathbf{HB}, \tag{2}$$

where $\mathbf{H}$ is the matrix storing outputs of hidden layer for each input object:

$$\mathbf{H} = \begin{pmatrix} f(\mathbf{w}_1 \cdot \mathbf{x_1} + b_1) & f(\mathbf{w}_2 \cdot \mathbf{x_1} + b_2) & \cdots & f(\mathbf{w}_N \cdot \mathbf{x_1} + b_N) \\ f(\mathbf{w}_1 \cdot \mathbf{x_2} + b_1) & f(\mathbf{w}_2 \cdot \mathbf{x_2} + b_2) & \cdots & f(\mathbf{w}_N \cdot \mathbf{x_2} + b_N) \\ \vdots & \vdots & \ddots & \vdots \\ f(\mathbf{w}_1 \cdot \mathbf{x_n} + b_1) & f(\mathbf{w}_2 \cdot \mathbf{x_n} + b_2) & \cdots & f(\mathbf{w}_N \cdot \mathbf{x_n} + b_N) \end{pmatrix}, \tag{3}$$

and $\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2, \cdots \mathbf{B}_N)^T$ and $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \cdots \mathbf{y}_n)$. To calculate weights assigned to outputs $\mathbf{B}$ we need to compute the Moore-Penrose generalized inverse of the matrix $\mathbf{H}$ that will be denoted as $\mathbf{H}^{-1}$.

One must note that due to the random nature of ELMs we may face a high variance in their behavior. To counter this drawback one may use regularization, which has been reported as having crucial effect on the quality of ELMs.

In order to regularize ELMs we use an orthogonal projection to get the Moore-Penrose pseudoinverse of $\mathbf{H}$:

$$\mathbf{H}^{-1^*} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T \tag{4}$$

where $\mathbf{H}^T$ is a transposed matrix $\mathbf{H}$. This allows us to add a ridge parameter $\frac{1}{\lambda}$ to the diagonal of $(\mathbf{H}^T\mathbf{H})$, which is known as the ridge-regression regularization approach. Applying it leads to obtaining a more stable solution.

After regularization one can calculate the matrix of output weights in step 3 of ELMs training according to:

$$\mathbf{B} = \left(\frac{\mathbf{I}}{\lambda} + \mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{Y}, \tag{5}$$

where $\mathbf{I}$ is an identity matrix of equal size to $\mathbf{H}$.

The ELM algorithm is summarized in a pseudo-code form in Algorithm 1.

Algorithm 1 Extreme Learning Machine with regularization

1: Randomly generate the bias matrix $\mathbf{b} = (\mathbf{b}_1, \mathbf{b}_2, \cdots \mathbf{b}_N)^T$
2: Randomly generate the weight matrix $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, \cdots \mathbf{w}_N)^T$
3: Calculate $\mathbf{H}$ using Eq. 3
4: Calculate Moore-Penrose pseudoinverse of $\mathbf{H}$ using Eq. 4
5: Calculate the matrix of output weights $\mathbf{B}$ using Eq. 5

3.2 Online extreme learning machines

Due to their low computational complexity, good generalization abilities and short response time ELMs have captured the attention of data stream community, proving themselves to be suitable for managing streaming and evolving objects. We will present the background of online ELMs (oELMs) that are able to efficiently update their structure in either batch or sample-by-sample manner [5].

This algorithm is based on Recursive Least Squares approach for sequentially updating $\mathbf{H}$. The oELMs proceeds in two steps. Firstly an initial ELM model is being trained. This requires a labeled training set representative to the analyzed stream. The size of this sample depends on the application of availability of labeled examples. however, it was proven that ELMs can initialize themselves even with small-sample training sets. This step uses standard ELM training procedure. In the second step ELM enters the online phase that will allow it to update its structure using incoming examples or data chunks. In accordance with incremental learning principles each example will be processed only once and after being processed by oELM it can be discarded. This reduces both computational complexity and storage requirements, two important factors in stream mining.

The oELM algorithm is summarized in a pseudo-code form in Algorithm 2.

Algorithm 2 Online Extreme Learning Machine

$\mathcal{TR}_0 \leftarrow$ starting labeled training set

procedure INITIALIZE($\mathcal{TR}_0$)
  Randomly generate the bias matrix $\mathbf{b} = (\mathbf{b}_1, \mathbf{b}_2, \cdots \mathbf{b}_N)^T$
  Randomly generate the weight matrix $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, \cdots \mathbf{w}_N)^T$
  Calculate starting $\mathbf{H}_0$ using Eq. 3
  $\mathbf{M}_0 = (\mathbf{H}_0^T \mathbf{H}_0)^{-1}$
  Calculate the starting matrix of output weights $\mathbf{B}_0 = \mathbf{M}_0 \mathbf{H}_0^{-1} \mathbf{Y}_0$
  $k \leftarrow 0$
end procedure

procedure UPDATE
  while stream = TRUE **do**
    $k \leftarrow k + 1$
    Get new object $\mathbf{x}_k$
    $\mathbf{h}_k = [f(\mathbf{w}_1 \cdot \mathbf{x_k} + b_1), \cdots, f(\mathbf{w}_N \cdot \mathbf{x_k} + b_N)]^T$
    $\mathbf{M}_k = \mathbf{M}_{k-1} \frac{\mathbf{M}_{k-1} \mathbf{h}_k \mathbf{h}_k^T \mathbf{M}_{k-1}}{1 + \mathbf{h}_k^T \mathbf{M}_{k-1} h_k}$
    $\mathbf{B}_k = \mathbf{B}_{k-1} + \mathbf{M}_k \mathbf{h}_k \left(\mathbf{y}_k^T - \mathbf{h}_k^T \mathbf{B}_{k-1} \right)$
  end while
end procedure

3.3 Schemes for handling imbalanced and drifting streams

In this paper we discuss two difficulties embedded in the nature of data stream mining: non-stationary nature of data and skewed class distributions. Each of these factors individually poses a significant challenge to the pattern classification systems. But when combined they may lead to a highly challenging task. Discussed oELMs are suitable for processing high-speed data streams, but display no robustness to either class imbalance or concept drift. Therefore, to obtain an efficient classification system we must augment them with procedures that will allow to compensate for changing and skewed environment. Let us discuss in details the proposed components of our framework:

Concept drift. In order to detect changes in the stream we either require a self-adaptive learner or an external module known as drift detector. In this work we will use the efficient Adaptive Data Stream Sliding Window (ADWIN) drift detector [2]. It maintains a variable-length window of arriving examples, where the window size is directly related to the hypothesis that were no changes in the average values stored within the window. The only parameter used by ADWIN is a confidence bound, which points to how confident the user needs to be in the output of the algorithm. This is a standard parameter used in all methods dedicated to random processes. Additional advantage of ADWIN is its reduced computational complexity. It does not store the entire window of objects, but instead compresses it using the exponential histogram technique. For a window of length L ADWIN requires has only $O(\log W)$ time and memory complexities per analyzed object.

Imbalanced data. In order to handle the skewed distribution between classes we propose to investigate three approaches based on data modification and classification error:

- Undersampling (oELM$_{under}$). In this strategy each obtained chunk of data is balanced by a random undersampling of the majority class. This solution has a low computational cost and reduced memory cost by shrinking the size of processed chunk. However, due to its random nature we can discard useful objects from the majority class.
- Oversampling (oELM$_{over}$). In this strategy each obtained chunk of data is balanced by a random oversampling of the minority class. This approach preserves all objects from both classes, artificially boosting the quantity of underrepresented group. This way we do not have a risk of removing valuable samples. However, this also leads to an increased memory consumption and processing costs as we increase the size of the processed chunk.
- Cost-sensitive (oELM$_{cost}$). In this approach we use the moving threshold paradigm of neural networks. In binary imbalanced problem the continuous outputs of two neurons in the final layer of oELM for the object x can be denoted as $y_{maj}(x)$ and $y_{min}(x)$, where $y_{maj}(x) + y_{min}(x) = 1$ and both outputs are bounded within [0,1]. We may apply the cost-sensitive modification of minority class output to counter the bias towards the majority class. New output $y^*_{min}(x)$ is computed as $y^*_{min}(x) = \eta y_{min}\text{Cost}[min, maj]$,

where Cost$[min, maj]$ is the cost penalty and η is a normalization parameter such that the new output is still bounded within [0,1]. This way we boost the recognition of the minority class without any data modifications or imposing additional computational costs. Output is adjusted during the classification phase. In order to calculate the Cost$[min, maj]$ and adapt it to changes in data we propose for it be equal to the imbalance ratio (IR) from the previous chunk.

Please note that we do not used more sophisticated preprocessing techniques due to their increased computational complexities. We aim at real-time classification of incoming chunks and therefore at reducing as much as possible the processing time required for it.

The proposed method is summarized in a pseudo-code form in Algorithm 3.

Algorithm 3 Online Extreme Learning Machine for imbalanced and drifting data streams

$\mathcal{DS}_0 \leftarrow$ starting labeled training set
$i \leftarrow 0$
Train initial oELM according to Algorithm 2.INITIALIZE and selected imbalance handling method

while stream = TRUE **do**
    $i \leftarrow i + 1$
    Get new data chunk $\mathcal{DS}_i$
    if cost-sensitive = TRUE **then**
        Classify $\mathcal{DS}_i$ using oELM with y^*_{min} and y_{maj}
    else
        Classify $\mathcal{DS}_i$ using oELM with y_{min} and y_{maj}
    end if
    Obtain true labels for $\mathcal{DS}_i$
    if undersampling = TRUE $\vee$ oversampling = TRUE **then**
        $\mathcal{DS}_i \leftarrow$ undersampling($\mathcal{DS}_i$) $\vee$ oversampling($\mathcal{DS}_i$)
    end if
    if ADWIN detects significant error change in oELM **then**
        Rebuild classifier according to Algorithm 2.INITIALIZE using only $\mathcal{DS}_i$
    else
        Update oELM with $\mathcal{DS}_i$ according to Algorithm 2.UPDATE
    end if
    if cost-sensitive = TRUE **then**
        Update Cost$[min, maj]$ with the new IR
    end if
end while

3.4 GPU-based oELM acceleration

Fast data processing is crucial when dealing with real-time data streams. Therefore, oELMs are an attractive proposition due to their significantly reduced computational complexity in comparison to standard neural networks or many reference classifiers. However, in order to process massive and changing streaming data we require rapid machine learning methods taking advantage of modern computing environments. Therefore, we decided to focus our attention on GPU-based implementation of extreme learning classifiers. There exists an efficient R package allowing for transferring the training procedure onto GPU using NVIDIA CUDA Basic Linear Algebra Subroutines (cuBLAS) library [1]. This allows for significant speed up of the most computationally costly operations such as calculating the matrix storing outputs of hidden layer $\mathbf{H}$ and its Moore-Penrose pseudoinverse (steps 3 and 4 of Algorithm 1). This however was not studied in case of real-time classification and continuously updated classifiers.

We propose to run the oELM using GPU processing applied to updating $\mathbf{M}_k$ and $\mathbf{B_k}$ (steps 17 and 18 of Algorithm 2) using cuBLAS library. This allows us to achieve a significant speed-up when processing continuously arriving chunks of data. Additionally, in case of ADWIN detecting a concept drift the new classifier being trained also uses GPU for calculating $\mathbf{H}_0$, $\mathbf{M}_0$ and $\mathbf{B}_0$ (steps 6, 7 and 8 of Algorithm 2), thus allowing a rapid adaptation to shifts and drifts in evolving data streams.

4 Experimental study

The aim of this experimental study is to verify the quality of the proposed modifications of oELMs for handling drifting and imbalanced fetal state data stream. Additionally, we wanted to establish the influence of GPU-based processing on the computational time required for the proposed models to adapt to changes in biomedical streaming data.

The used oELM has optimized number of hidden neurons $\in$ [10;100] using the initial training set and sigmoid activation function is being applied.

Classifiers are trained on given chunk and tested on incoming one. Then we fuse incoming chunk with the existing one and repeat the procedure for new data. The data block size used for creating data chunks was $d = 5$ to simulate constant monitoring of 5 high-risk patients.

For evaluating classifiers in imbalanced non-stationary scenario we use the prequential AUC [3].

The experiments were performed on a machine equipped with an Intel Core i7-4700MQ Haswell @ 2.40 GHz processor and 16.00 GB of RAM with installed NVidia GTX295 GPU. All experiments were conducted in R[4] environment, with the usage of RMOA[5] package.

[3] http://www.cs.put.poznan.pl/dbrzezinski/software.php
[4] http://www.r-project.org/
[5] https://github.com/jwijffels/RMOA

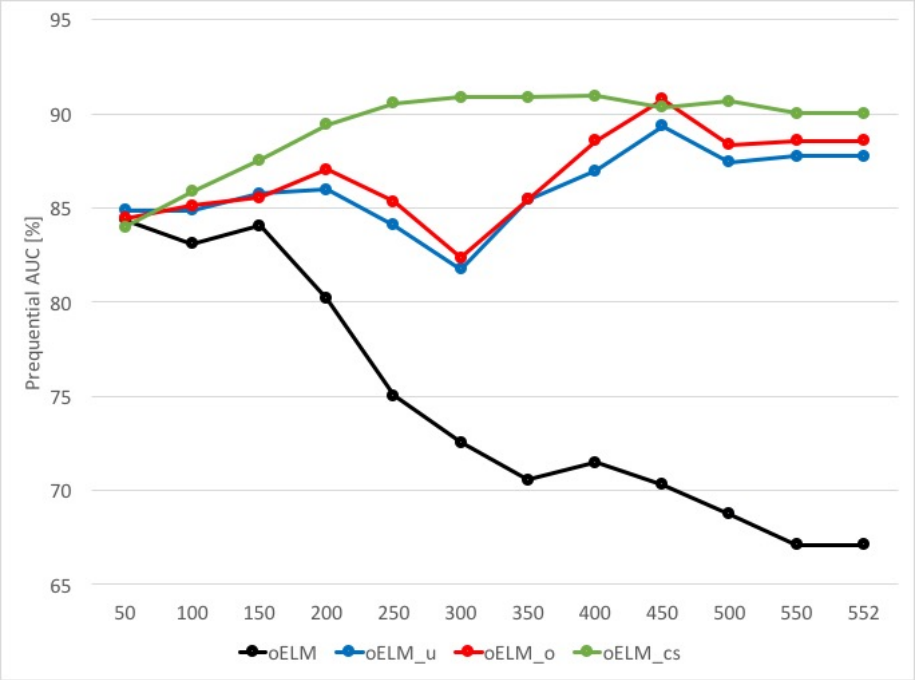


Fig. 1. Prequential AUC for examined classifiers used for mining imbalanced medical data stream for fetal state monitoring.

Obtained results for the CTU-UHB dataset are given in Figure 1, while the memory consumption and update times on CPU and GPU are reported respectively in Tables 1 and 2.

Table 1. Averaged memory consumption [MB] displayed by the examined classifiers.

oELM	oELM$_{under}$	oELM$_{over}$	oELM$_{cost}$
0.78	0.34	1.42	0.90

Table 2. Comparison of averaged updating times [s.] per chunk displayed by the examined classifiers computed wit the usage of CPU and GPU.

CPU				GPU			
oELM	oELM$_{under}$	oELM$_{over}$	oELM$_{cost}$	oELM	oELM$_{under}$	oELM$_{over}$	oELM$_{cost}$
1.76	3.05	4.18	2.01	0.24	0.65	0.71	0.28

When analyzing performance of classifiers for online fetal state monitoring one must take into consideration several criteria in order to gain a broader view on the problem. When we concentrate only on the classification efficacy expressed in terms of AUC we can see that standard oELM with ADWIN detector cannot handle imbalanced and drifting streams properly. Out of three investigated approaches the combination of oELM and oversampling returns superior classification quality on skewed streams. Undersampling does not deliver as good results which can be explained by high imbalance ratio in most of the problems. Here undersampling may remove useful objects that represent current state of the concept and may point out to the potential drift occurring. Thus it is possible to discard an useful information which influences the observed classification process. Cost-sensitive oELM in most cases achieves AUC performance very similar to the oversampling model.

However, when we take into account memory consumption and time complexity required to update a model for each chunk our conclusions must change. Oversampling-based approach is characterized by highest memory and time requirements due to its need to generate, store and process chunks of increased size (which is especially vivid for cases with high IR). On the other hand undersampling-based approach is characterized by the lowest requirements due to significant reduction of the data chunk size being analyzed. Cost-sensitive approach work on unaltered set of objects, applying only modification to its output. Therefore, its requirements are identical as in case of standard oELM.

This allows us to conclude that the cost-sensitive oELM is the most suitable choice for processing drifting and imbalanced medical data streams. It offers classification accuracy comparable to oversampling solution while maintaining balanced time and memory requirements.

When comparing processing times we can see that simple delegation of matrix-based operation to GPU allows us to achieve roughly 10 times computing acceleration regardless of the method used. This proves that using GPU-based oELMs allow for real-time mining of high-speed data streams.

Combination of oELM improved with ADWIN detector, cost-sensitive output modification and realized on GPU offers an adaptive, skew-insensitive and rapid classifier for online fetal state monitoring from imbalanced medical data stream.

5 Conclusions

In this paper an issue of designing an online decision support system for mining medical data streams was discussed. We concentrated on the problem of monitoring fetal state from drifting and skewed stream of instances in real-time. Such a support tool is of high usefulness to hospital wards, where difficult pregnancies should be monitored continuously during day and night.

We have used a novel approach for handling imbalanced and drifting data streams using online Extreme Learning Machines enhanced with drift detector and data-level or cost-sensitive modifications. We showed that such additions allows oELM to efficiently adapt to non-stationary properties of incoming objects,

while alleviating the influence of skewed distributions on its performance. Additionally, an efficient GPU-based implementation was proposed for rapid training and updating of online classifiers. This allowed for achieving 10 times speed-up when compared to standard CPU-based implementation. This issue was crucial in term of mining high-speed data streams.

Acknowledgment

The work was supported by the statutory funds of the *Department of Systems and Computer Networks*, *Wroclaw University of Science and Technology* and by *The Polish National Science Centre* under the grant agreement no. *DEC-2013/09/B/ST6/02264*. All computational experiments were carried out using computer equipment sponsored by EC under FP7, *Coordination and Support Action*, *Grant Agreement Number* 316097, ENGINE – *European Research Centre of Network Intelligence for Innovation Enhancement* (http://engine.pwr.wroc.pl/).

References

1. M. Alia-Martinez, J. Antonanzas, Fernando Antoñanzas-Torres, Alpha V. Pernía-Espinoza, and Ruben Urraca-Valle. A straightforward implementation of a gpu-accelerated ELM in R with NVIDIA graphic cards. In *Hybrid Artificial Intelligent Systems - 10th International Conference, HAIS 2015, Bilbao, Spain, June 22-24, 2015, Proceedings*, pages 656–667, 2015.
2. Albert Bifet and Ricard Gavalda‘. Learning from time-changing data with adaptive windowing. In *Proceedings of the Seventh SIAM International Conference on Data Mining, April 26-28, 2007, Minneapolis, Minnesota, USA*. SIAM, 2007.
3. B. Cyganek and S. Gruszczyński. Hybrid computer vision system for drivers' eye recognition and fatigue monitoring. *Neurocomputing*, 126:78–94, 2014.
4. J. Gama. A survey on learning from data streams: current and future trends. *Progress in AI*, 1(1):45–55, 2012.
5. Guang-Bin Huang, Dianhui Wang, and Yuan Lan. Extreme learning machines: a survey. *Int. J. Machine Learning & Cybernetics*, 2(2):107–122, 2011.
6. D. Jankowski and K. Jackowski. Evolutionary algorithm for decision tree induction. In *Computer Information Systems and Industrial Management - 13th IFIP TC8 International Conference, CISIM 2014, Ho Chi Minh City, Vietnam, November 5-7, 2014. Proceedings*, pages 23–32, 2014.
7. Michal Przewozniczek, Róza Goscien, Krzysztof Walkowiak, and Miroslaw Klinkowski. Towards solving practical problems of large solution space using a novel pattern searching hybrid evolutionary algorithm - an elastic optical network optimization case study. *Expert Syst. Appl.*, 42(21):7781–7796, 2015.
8. Jirí Spilka, Václav Chudácek, Michal Koucký, Lenka Lhotská, Michal Huptych, Petr Janku, George K. Georgoulas, and Chrysostomos D. Stylios. Using nonlinear features for fetal heart rate classification. *Biomed. Signal Proc. and Control*, 7(4):350–357, 2012.
9. I. Triguero, D. Peralta, J. Bacardit, S. García, and F. Herrera. MRPR: A mapreduce solution for prototype reduction in big data classification. *Neurocomputing*, 150:331–345, 2015.

10. M. Woźniak. A hybrid decision tree training method using data streams. *Knowl. Inf. Syst.*, 29(2):335–347, 2011.
11. Michal Wozniak, Andrzej Kasprzak, and Piotr Cal. Application of combined classifiers to data stream classification. In *Proceedings of the 10th International Conference on Flexible Query Answering Systems FQAS 2013*, LNCS, page in press, Berlin, Heidelberg, 2013. Springer-Verlag.

Imbalance medical data classification using Exposer Classifier Ensemble

Pawel Ksieniewicz and Michal Wozniak

Department of Systems and Computer Networks, Faculty of Electronics
Wrocław University of Science and Technology, Wrocław, Poland.
{pawel.ksieniewicz,michal.wozniak}@pwr.edu.pl

Abstract. This work proposes an usage of novel transformation into feature space that follows a photographic intuition: that we can build from pairs of features in original space some kind of photographic plate where the sample data are projected to create a picture of the data distribution in the feature subspace defined by the feature pair. These photographic plates may be used as individuals of a classifier ensemble. The approach allows a natural definition of a confidence weight affecting each individual classifier out for the construction of a combination rule used by the ensemble The approach is insensitive to sample size, robust to dimension increase, and should be able to work well with imbalanced medical data.

Keywords: machine learning, representation learning, imbalance data, medical data

1 Introduction

Recently, the *representation learning* is the fucus of intense research of machine learning community . The underlying idea is that the key for successful discrimination of difficult datasets is a good feature extraction [1]. A transformation of the data space into another space where classification is easy. A kind of this transformation can be performed by an *exposer* — a data structure, being neither histogram nor scatter-plot.

Histogram can be defined as a graphical representation of the distribution of numerical data. It is an estimate of the probability distribution of a continuous variable (quantitative variable). It was first introduced by Karl Pearson [4].

Construction of a histogram consist of the division[1] of the range of values into a series of intervals, called bins and counting how many elements will fall in each of them. The bins *must be adjacent* and usually are equal-sized.

Scatter-plot is a plot used to show distribution of two features for a set of data, displayed as a cloud of points, representing objects from dataset, each corresponding with the value of chosen feature determining the position on the axes [3]. Example *scatter plots* for *iris* dataset are presented in Figure 1.

[1] Unfortunately not the Joy Division.

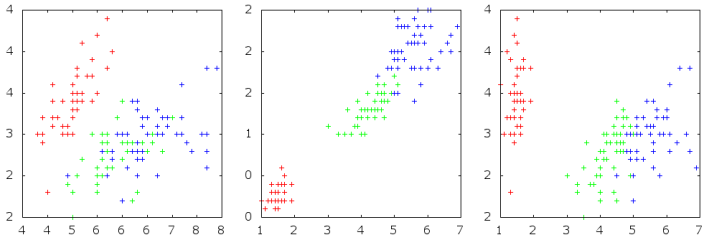


Fig. 1. Example scatter plots for *iris* dataset (1:2, 3:4, 2:3)

2 Exposer Classifier Ensemble

Exposer is drawing from both *histogram* and *scatter plot*, according to three rules:

1. like in *histogram* the range of values is divided into a series of intervals like in a *scatter plot*,
2. the combination of features is analyzed,
3. the rule of bin adjacency is broken here.

Example *exposers*, generated for the same variables as the illustrated *scatter plots* are presented in Figure 2.

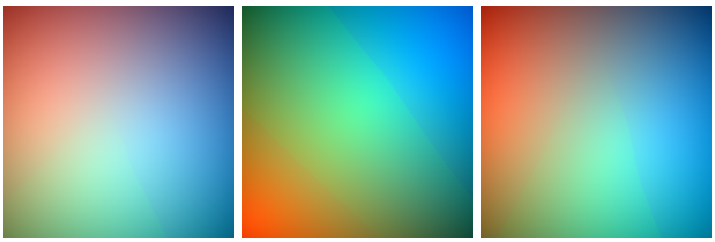


Fig. 2. Example exposers plots for *iris* dataset (1:2, 3:4, 2:3)

The construction of *exposer* is inspired by the process of plate light exposure of chemical photography. Hence, its control parameters are plate grain (*exposer* counterpart of *histograms* bin) and a light dispersion factor (called later a *radius*, as a relative width of a bin according to a range of values). Instead of exposing the photographic plate coated with light-sensitive chemicals to the light source, a numerical representation matrix is exposed to the beams projected from the data samples. Procedure *takes a photography* of a pair of features of the data samples, where intensity of a *light* in every point is a density aggregation of the data samples falling in its neighborhood.

The main difference from classic photography is a redefinition of the concept of color. For *exposers* it consists not of classical three RGB spectral channels [5], but of one dimension per class of the dataset. Hence, the representation matrix has as many layers as classes. The representation matrix exposure process sensitizes each layer projecting only objects from the class corresponding to the layer. There may be assumed, that *exposing* procedure generates some kind of multispectral imaging [2] from the data.

Algorithm 1 shows pseudocode of procedure used to create an exposer.

Algorithm 1 Expose algorithm

```
1: procedure EXPOSE(objects,radius,grain,features)
2:     for class in classes do
3:         objects' ← objects labeled as class
4:         for x ← 1 to grain do
5:             for y ← 1 to grain do
6:                 base ← (x, y)/grain
7:                 brightness ← 0
8:                 for object in objects' do
9:                     base' ← object(features)
10:                    distance ← √(Σ (base' − base)²)
11:                    if distance < radius then
12:                        brightness ← brightness(radius − distance)
13:                    end if
14:                end for
15:                E[x, y, class] ← brightness
16:            end for
17:        end for
18:    end for
19: end procedure
```

We have set C, which contains two-element combinations of d features.

$$\begin{aligned} C &= \{\gamma_1, \ldots, \gamma_{|C|}\} \\ \gamma_z &= [x^{(u)}, x^{(v)}], \quad u, v \in \{1, \ldots, d\}, \quad u \neq v \\ |C| &= \binom{d}{2} \end{aligned}$$

Representation of *exposer* E is an multispectral image of spatial dimensions $G \times G$ and a spectral dimension M, where G is a *grain* parameter and M is a number of classes in dataset.

Exposer E is a multispectral image of spatial dimensions $G \times G$ and a spectral dimension M, where G is a *grain* parameter and M is a number of classes in dataset.

$$\begin{aligned} E &\in G \times G \times M \\ E &= \{E_1, \ldots, E_M\} \end{aligned}$$

Every point described by g_u and g_v gives *spectral signature pix*.

$$pix(g_u, g_v) = [pix_1(g_u, g_v), \ldots, pix_M(g_u, g_v)]$$

Which is a discretization using g quants in every spatial dimension.

$$pix_i(g_u, g_v) = \sum_{k=1}^{n} \Big[d\Big(cent([g_u, g_v]), x_k\Big) < r \ \wedge \ i_k = i \Big] \cdot \Big(r - d(cent([g_u, g_v]), x_k) \Big)$$

Proper color-visualization would be possible only with 3-class problem. To universalize it, an HSV interpretation was introduced. Figure 3 shows its output.

$$HSV(pix) = \left[\frac{argmax(pix) \times 360^\circ}{M}, \ max(pix) - min(pix), \ max(pix) \right]$$

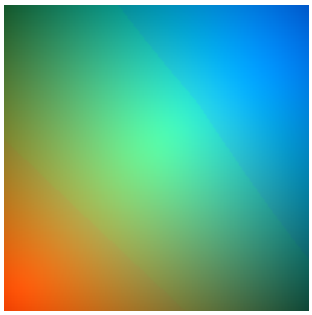


Fig. 3. Exposer

Exposer Classifier Using the *exposers* as a classification tool is quite easy process. There is an *exposer*, exposed on a *learning set* and a testing sample x_k from a *testing set* to classify.

$$x_k = \{x_k^{(1)}, x_k^{(2)}, \ldots, x_k^{(d)}\}, \qquad x_k \in \mathcal{TS}$$

Localization of pixel according to sample.

$$fin(x) = [g_u, g_v]$$

The prediction (Ψ) is the class index with the maximum value from a spectral signature *pix* corresponding to the testing sample.

$$\Psi(x_k) = \underset{i \in \mathcal{M}}{argmax} \left(pix_i\Big(fin(x_k)\Big) \right)$$

Exposer Classifier Ensemble To create the classification committee, as an ensemble of exposers, the following procedure is used.

We have ensemble of *exposers* Π.

$$\begin{aligned} &\Gamma \subset C, \qquad |\Gamma| = N, \qquad N \leq \binom{d}{k} \\ &\Pi = \{\Psi^{(1)}, \ldots, \Psi^{(N)}\} \end{aligned} \tag{1}$$

And a weight θ_i, which is an average *saturation* for points according to *class* for *values* higher than given *threshold*.

$$\theta_i = \frac{\sum_{g_u=1}^{G} \sum_{g_v=1}^{G} \Big[argmax(pix) = i \ \wedge \ pix_i > threshold \Big] \ S(pix)}{\sum_{g_u=1}^{G} \sum_{g_v=1}^{G} \Big[argmax(pix) = i \Big]} \tag{2}$$

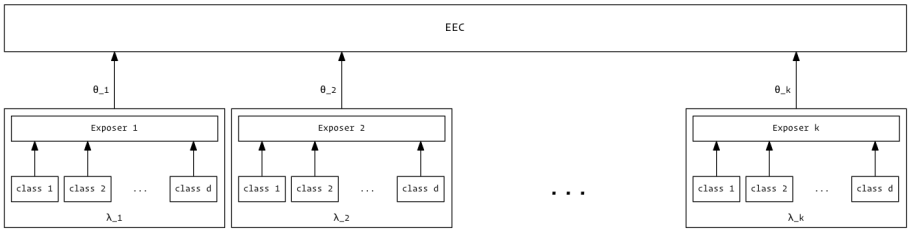


Fig. 4. ECE classification model diagram.

Classification model The classification model is a three-level construction, presented in Figure 4, which consists of:

- a set of monochrome layers,
- a member classifier, characterized by a combination of features, denoted by γ_i, and a weight θ_i used to combine its output with the remaining classifiers into *exposer*,
- ECE ensemble.

3 Experimental evaluation

3.1 Datasets

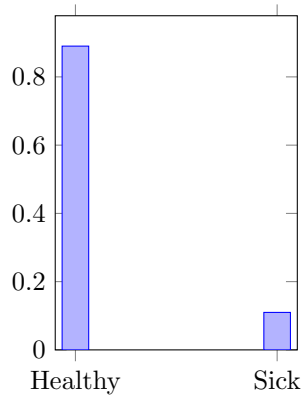
In medical data sets, data are predominately composed of *normal* samples with only a small percentage of *abnormal* ones, leading to the so-called class imbalance problems. In class imbalance problems, inputting all the data into the classifier to build up the learning model will usually lead a learning bias to the majority class. To deal with this a strategy which over-samples the minority class and

under-samples the majority one to balance the data sets is often used. Often there also occurs a problem with *missing values*.

For experiments, two imbalanced datasets was chosen.

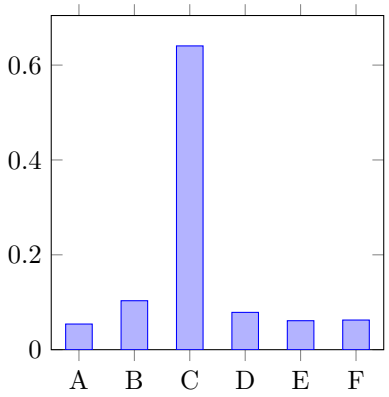
Yeasts3

- 1484 samples
- 2 classes
- 8 features
- imbalanced
- binary



Hypertension

- 1424 samples
- 6 classes
- 19 features
- imbalanced
- multilabel
- missing values



3.2 Experimental environment

Experiments was conducted under our original machine learning framework KSSKML[2] and its extension ECE[3], available as a module at `pip` repository, actually in version 0.6.3.

The code of following evaluation is available at public repository [4] and includes four experiments:

ECE	Regular ECE for both *Yeast3* and *Hypertension*.
ECE_B	Binarized ECE for *Hypertension*.
ECE_S	Sick-only ECE for *Hypertension*.
ECE_2	Two-staged ECE for *Hypertension*: *First* $\boldsymbol{ECE_B}$ *for healthy or sick, later* $\boldsymbol{ECE_S}$ *to distinguish kind of sickness.*

For all experiments 5v2 cross validation was used, and as comparison, they were conducted also using k-NN algorithm.

3.3 Results

[2] https://github.com/w4k2/KSSKML
[3] https://github.com/w4k2/ece
[4] http://github.com/xehivs/hypertension

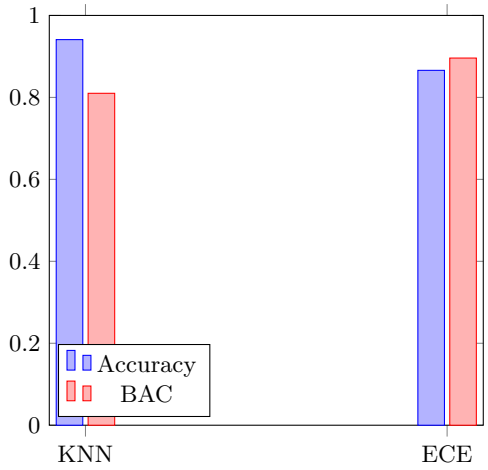


Fig. 5. Yeasts3 results

Yeast3 results As we can see in Figure 3.3 and Table 1, ECE algorithm works better than k-NN. We can observe regression of accuracy, but for imbalanced medical data, the balanced accuracy is more accurate measure.

Hypertension results As we can see in Figure 3.3 and Table 1, for more complex – multi-class problem – the classical approach (ECE) gives us smaller increase of classification effectiveness. The best solutions seems to be initial binarization of problem (ECEB) and dealing separately with discrimination on healthy–sick level and differentiating between classes of illness.

Table 1. Accuracy and balanced accuracy according to experiment

Dataset	Algorithm	Accuracy	BAC
Yeasts3	KNN	.941	.810
	ECE	.866	.896
Hypertension	KNN	.878	.569
	ECE	.790	.608
	ECE_B	.670	.673
	ECE_S	.778	.592
	ECE_2	.799	.626

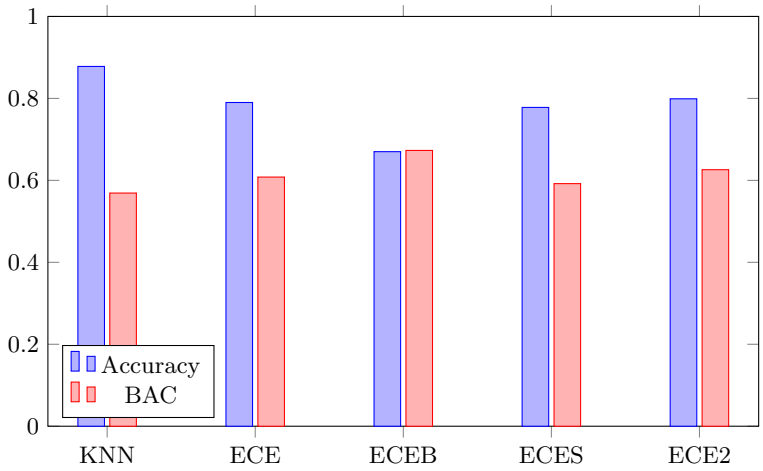


Fig. 6. Hypertension results

4 Conclusions

In this paper, we present how a classifier, formulated as an ensemble of multi-spectral images generated from paired features of data, deals with imbalance medical data.

We showed, that this approach leads to create classifier that are competitive to existing ones and able to outperform them. It could be used for real-life applications.

For future works we plan to test also an heuristic method of feature combination limitation, to improve performance of classifier and to adapt the proposed model to deal with the fast changing streaming data.

Acknowledgment

The work was supported by the statutory funds of the *Department of Systems and Computer Networks*, *Wroclaw University of Science and Technology* and by *The Polish National Science Centre* under the grant agreement no. *DEC-2013/09/B/ST6/02264*. All computational experiments were carried out using computer equipment sponsored by EC under FP7, *Coordination and Support Action*, *Grant Agreement Number* 316097, ENGINE – *European Research Centre of Network Intelligence for Innovation Enhancement* (http://engine.pwr.wroc.pl/).

References

1. Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2012.

2. G C Giakos. Multifusion, Multispectral, Optical Polarimetric Imaging Sensing Principles. *IEEE Transactions on Instrumentation and Measurement*, 55(5):1628–1633, 2006.
3. M Jessica. Seeing Through Statistics 3rd Edition, Thomson Brooks/Cole, 2005. Technical report, ISBN 0-534-39402-7, 2005.
4. K Pearson. Contributions to the Mathematical Theory of Evolution. II. Skew Variation in Homogeneous Material. *Philosophical Transactions of the Royal Society of London Series A*, 186:343–414, 1895.
5. G Svaetichin. Spectral response curves from single cones. *Acta physiologica Scandinavica. Supplementum*, 39(134):17–46, 1956.

Author Index