Naive Bayes Classifier for Dynamic Chaining approach in Multi-label Learning.

PAWEL TRAJDOS
Wroclaw University of Science
and Technology
Department of Systems
and Computer Networks
Wyb. Wyspianskiego 27, 50-370 Wroclaw
POLAND
pawel.trajdos@pwr.edu.pl

MAREK KURZYNSKI
Wroclaw University of Science
and Technology
Department of Systems
and Computer Networks
Wyb. Wyspianskiego 27, 50-370 Wroclaw
POLAND
marek.kurzynski@pwr.edu.pl

Abstract: In this paper, we addressed an issue of building dynamic classifier chain ensembles for multi-label classification. We built a classifier that allows us to change label order of the chain without rebuilding the entire model. Such a model allows anticipating the instance-specific chain order without a significant increase in computational burden. The proposed chain model is built using the Naive Bayes classifier as a base single-label classifier. Additionally, we proposed a simple heuristic that allows the system to find relatively good label order. That is, the heuristic tries to minimise the phenomenon of error propagation in the chain. The experimental results showed that the proposed model based on Naive Bayes classifier the above-mentioned heuristic is an efficient tool for building dynamic chain classifiers.

Key–Words: multi-label, classifier-chains, naive bayes, dynamic chains

1 Introduction

Under well-known single-label classification framework, an object is assigned to only one class which provides a full description of the object. However, many real-world datasets contain objects that are assigned to different categories at the same time. All of these categories constitute a full description of the object. Omitting of one of these concepts induces a loss of information. Classification process in which such kind of data is involved is called multi-label classification [10]. A great example of a multi-label dataset is a gallery of tagged photos. Each photo may be described using such tags as mountains, sea, forest, beach, sunset, etc. Multi-label classification is a relatively new idea that is explored extensively for last two decades. As a consequence, it was employed in a wide range of practical applications including text classification [17], multimedia classification [24] and bioinformatics [33] to name a few.

Multi-label classification algorithms can be broadly partitioned into two main groups i.e. dataset transformation algorithms and algorithm adaptation approaches [10].

Methods belong to the group of algorithm adaptation approaches provides a generalisation of an existing multi-class algorithm. The generalised algorithm is able to solve multi-label classification problem in a direct way. Among the others, the most known approaches from this group are: multi label KNN algorithm [17], the Structured SVM approach [5] or deep-learning-based algorithms [31].

In this paper, we investigate only dataset transformation algorithms that decompose a multi-label problem into a set of single-label classification tasks. To reconstruct a multi-label response, during the inference phase, outputs of the underlying single-label classifiers are combined in order to create a multi-label prediction.

Let's focus on one of the simplest decomposition methods. That is the *binary relevance* (BR) approach that decomposes a multi-label classification task into a set of *one-vs-rest* binary classification problems [1]. This approach assumes that labels are conditionally independent. However the assumption does not hold in most of real-life recognition problems, the BR framework is one of the most widespread multi-label classification methods [30]. This is due to its excellent scalability and acceptable classification quality [19].

To preserve scalability of BR systems, and provide a model of inter-label relations, Read et al. [22, 23] provided us with the *Classifier Chain* model (CC) which establish a linked chain of modified one-vs-rest binary classifiers. The modification consists of an extension of the input space of single-label classifiers

along the chain sequence. To be more strict, for a given label sequence, the feature space of each classifier along the chain is extended with a set of binary variables corresponding to the labels that precede the given one. The model implies that, during the training phase, input space of given classifier is extended using the ground-truth labels extracted from the training set. During the inference step, due to lack of the ground-truth labels, we employ binary labels predicted by preceding classifiers. The inference is done in a greedy way that makes the best decision for each of considered labels. That is, the described approach passes along the chain, information allowing CC to take into account inter-label relations at the cost of allowing the label-prediction-errors to propagate along the chain [23]. This way of performing classification induces a major drawback of the CC system. That is, the performance of a chain classifier strongly depends on chain configuration [25]. To overcome these effects, the authors suggested to generate an *ensemble of chain classifiers* (ECC). The ensemble consists of classifiers trained using different label sequences [22].

The originally proposed ECC ensemble uses randomly generated label orders. This simple, yet effective approach allows improving the classification quality significantly in comparison to single chain classifier. However, the intuition says that there is still room for improvement. Indeed, later research shows that the members of the ensemble may be chosen in such a way that provides further improvement of classification quality [12, 11, 29]. The above-cited research provides methods of building the entire ensemble in a heuristic way. That is, the algorithms generate a large set of random label orders and then chooses the best ensemble using a genetic algorithm.

Another strategy is to generate bootstrap samples of the original training set and then to choose the best classifiers for each of samples. Under this framework, the chain structure is usually optimised using Bayesian Network [36] or Monte Carlo optimisation [20].

The previously cited methods build ensemble structure during the training procedure. Consequently, throughout this paper, this kind of methods will be called static methods. The dynamic chain classifiers, on the other hand, determines the best label order at the prediction phase [4]. The above-mentioned classifier produces a set of randomly generated label sequences and then validates the chain classifiers. During the validation phase, each point from the validation set is assigned with a label order that produces the most accurate output vector for this point. As the experimental research shows, the dynamic methods of building a label order may achieve better classification quality [4].

We observed that during the building of a dynamic chain classifier, multiple chain classifiers must be learned. These classifiers are built using the same training set and differ only in chain order. As a consequence, the computational burden of the algorithm may be reduced if there exists a classifier that is trained once and changing the label sequence is done without rebuilding the model. To address this issue, we built a model based on the Naive Bayes [15] approach that meets the above-mentioned properties.

Additionally, we proposed a dynamic method of determining the chain order based on classification quality for each label separately.

The rest of the paper is organised as follows. Next Section 1 provides a formal description of the multi-label classification problem and describes the developed approach. Section 3 contains a description of the conducted experiments. The results are presented and discussed in Section 4. Finally, Section 5 concludes the paper.

2 Proposed Method

In this section, we introduce a formal notation of multi-label classification problem and provide a description of the proposed method.

2.1 Preliminaries

Under the *multi-label* (ML) formalism a $d-$dimensional object $\vec{x} = [x_1, x_2, \ldots, x_d] \in \mathcal{X}$ is assigned to a set of labels indicated by a binary vector of length L: $\vec{y} = [y_1, y_2, \ldots, y_L] \in \mathcal{Y} = \{0, 1\}^L$, where L denotes the number of labels.

In this paper, we follow a statistical classification framework. As a consequence, it is assumed that object $\vec{x}$ and its set of labels y are realizations of corresponding random vectors $\vec{\mathbf{X}} = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_d]$, $\vec{\mathbf{Y}} = [\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_L]$ and the joint probability distribution $P(\vec{\mathbf{X}}, \vec{\mathbf{Y}})$ on $\mathcal{X} \times \mathcal{Y}$ is known.

Because the above-mentioned assumption is never meet in real world, in this study, we suppose that multi-label classifier H, which maps feature space $\mathcal{X}$ to the set $\mathcal{Y}$, is built in a supervised learning procedure using the training set $\mathcal{T}$ containing N pairs of feature vectors $\vec{x}$ and corresponding class labels $\vec{y}$:

$$\mathcal{T} = \left\{ (\vec{x}^{(1)}, \vec{y}^{(1)}), (\vec{x}^{(2)}, \vec{y}^{(2)}), \ldots, (\vec{x}^{(N)}, \vec{y}^{(N)}) \right\}. \tag{1}$$

2.2 Naive Bayes Classifier for Classifier Chains

In this paper, we consider ML classifiers build according to the chain rule. That is, the classifier H is an

ensemble of L single-label classifiers ψ_i that constitutes a linked chain which is built according to a permutation of label sequence π. As it was mentioned earlier, in this paper we follow the statistical classification framework. Consequently, each classifier $\psi_{\pi(i)}$ along with the chain makes its decision according to the following rule:

$$h_{\pi(i)}(\vec{x}) = \underset{y\in\{0,1\}}{\operatorname{argmax}} P(\mathbf{Y}_{\pi(i)} = y | B_{\pi(i)}(\vec{x})), \quad (2)$$

where $B_{\pi(i)}(\vec{x})$ is a random event defined below:

$$\begin{aligned} B_{\pi(i)}(\vec{x}) = (\vec{\mathbf{X}} = \vec{x}, \mathbf{Y}_{\pi(i-1)} = h_{\pi(i-1)}(\vec{x}) \\ , \mathbf{Y}_{\pi(i-2)} = h_{\pi(i-2)}(\vec{x}), \cdots \\ , \mathbf{Y}_{\pi(1)} = h_{\pi(1)}(\vec{x})). \end{aligned} \quad (3)$$

The probability defined in (2) is then computed using the Bayes rule:

$$\begin{aligned} P(\mathbf{Y}_{\pi(i)} = y | B_{\pi(i)}(\vec{x})) = \frac{P(\mathbf{Y}_{\pi(i)} = y)}{P(B_{\pi(i)}(\vec{x}))} \\ * P(B_{\pi(i)}(\vec{x}) | \mathbf{Y}_{\pi(i)} = y). \end{aligned} \quad (4)$$

The term $P(B_{\pi(i)}(\vec{x}))$ does not depend on event $\mathbf{Y}_{\pi(i)} = y$. Consequently, the decision rule (2) is rewritten:

$$\begin{aligned} h_{\pi(i)}(\vec{x}) = \underset{y\in\{0,1\}}{\operatorname{argmax}} P(\mathbf{Y}_{\pi(i)} = y) \\ * P(B_{\pi(i)}(\vec{x}) | \mathbf{Y}_{\pi(i)} = y) \end{aligned} \quad (5)$$

Now, to improve the readability we simplify the notation:

$$P(B_{\pi(i)}(\vec{x}) | \mathbf{Y}_{\pi(i)} = y) = P(B_{\pi(i)}(\vec{x}) | y). \quad (6)$$

Then, following the Naive Bayes rule, we assume that all random variables that constitute $B_{\pi(i)}(\vec{x})$ are conditionally independent given $\mathbf{Y}_{\pi(i)} = y$. Consequently, $P(B_{\pi(i)}(\vec{x})|y)$ is defined using the following formula:

$$\begin{aligned} P(B_{\pi(i)}(\vec{x}) | y) = \prod_{m=1}^{d} P(\vec{\mathbf{X}}_m = \vec{x}_m | y) \\ * \prod_{l=1}^{l=i-1} P(\mathbf{Y}_{\pi(l)} = h_{\pi(l)}(\vec{x}) | y). \end{aligned} \quad (7)$$

Now, it is easy to see that the term $\prod_{l=1}^{l=i-1} P(\mathbf{Y}_{\pi(l)} = h_{\pi(l)}(\vec{x})|y)$, contrary to $\prod_{m=1}^{d} P(\vec{\mathbf{X}}_m = \vec{x}_m|y)$, depends on the chain structure. Furthermore, all probability distributions used in the above-mentioned terms can be estimated during the training phase when the chain structure is unknown.

The training and inference phases are described in detail using pseudocode shown in Algorithms 1 and 2.

Algorithm 1 Pseudocode of the learning procedure.

```
Input data:
  T - training set;
BEGIN
  Split T into T_A and V so that:
  |T_A| = t|T| and |V| = (1-t)|T|, t ∈ (0,1)
  T_A ∩ V = ∅;
  Using T_A build estimators of
  the following distributions:
    P(Y_π(i) = y) ∀i ∈ {1,2,···,L}, y ∈ {0,1}
    P(X_m | Y_π(i) = y) ∀i ∈ {1,2,···,L}, y ∈ {0,1}, m ∈ {1,2,···,d}
    P(Y_π(l) | Y_π(i) = y) ∀i,l ∈ {1,2,···,L}; i ≠ l
END
```

Algorithm 2 Pseudocode of the inference procedure.

```
Input data:
  x ∈ X -- input instance;
  V -- validation set;
BEGIN
  #Query the BR models
  FOR i ∈ {1,2,···,L}:
    e_i^0 = ∏_{m=1}^{d} P(X_k = x_m | Y_i = 0);
    e_i^1 = ∏_{m=1}^{d} P(X_k = x_m | Y_i = 1);
  END FOR;
  Determine label permutaion π using V and x;
  SET i = 1;
  DO:
    h_π(i)(x) = argmax_{y∈{0,1}} e^y_π(i) P(Y_π(i) = y)
    FOR j ∈ {i+1, i+2, ···, L}:
      d^0_π(j) := e^0_π(j) * P(Y_π(i) = h_π(i)(x) | Y_π(j) = 0)
      d^1_π(j) := e^1_π(j) * P(Y_π(i) = h_π(i)(x) | Y_π(j) = 1)
    END FOR;
    i := i + 1;
  WHILE (i < L);
  RETURN [h_1(x), h_2(x), ···, h_L(x)];
END
```

2.3 Computational complexity

In this section, we assess the increase in computational complexity that the proposed algorithm causes.

First of all, it is easy to see that for both the original and the proposed algorithm the number of estimators that must be built to assess $P(\vec{\mathbf{X}}_m|\mathbf{Y}_{\pi(i)} = y)\forall i \in \{1, 2, \cdots, L\}, y \in \{0, 1\}$ is: $2Ld$.

The number of estimators of $P(\mathbf{Y}_{\pi(i)} = y)\forall i \in \{1, 2, \cdots, L\}, y \in \{0, 1\}$ that must be built is also the same for both classifiers: L.

The key difference is in the number of estimators of $P(\mathbf{Y}_{\pi(l)}|\mathbf{Y}_{\pi(i)} = y)$ that must be built. For the original CC classifier the number of estimators that is built is $L(L-1)$. On the other hand our method builds $2L^2$ estimators.

At the inference phase, the only additional calculations are performed to determine the permutation of labels. Since the validation set is involved in this process, a number of calculations is proportional to $O(|\mathcal{V}|L)$.

2.4 Dynamic Chain order

In this subsection, we define a local measure of classification quality. To do so, we employed a modified version of the well-known F_1 measure.

First of all, we defined a fuzzy neighbourhood in the input space. The neighbourhood of an instance $\vec{x}$ is defined using the following fuzzy set [35]:

$$\mathcal{N}(\vec{x}) = \left\{ \left(\vec{x}^{(n)}, \vec{y}^{(n)}, \mu(\vec{x}, \vec{x}^{(n)}) \right) : \left(\vec{x}^{(n)}, \vec{y}^{(n)} \right) \in \mathcal{V} \right\}, \tag{8}$$

where each tripplet $\left(\vec{x}^{(n)}, \vec{y}^{(n)}, \zeta \right)$ defines fuzzy set with the membership coefficient ζ. The membership function $\mu(\vec{x}, \vec{x}^{(n)})$ is defined using gaussian potential function:

$$\mu(\vec{x}, \vec{x}^{(n)}) = \exp(-\beta \delta(\vec{x}, \vec{x}^{(n)})^2). \tag{9}$$

The distance function $\delta(\vec{x}, \vec{x}^{(n)})$ is simple euclidean distance and the β coefficient is tuned during the experiments.

Then, we define set of points that belongs to given label $\mathcal{V}_l$ and that are classified as given label $\mathcal{D}_l$:

$$\mathcal{V}_l = \left\{ \left(\vec{x}^{(n)}, \vec{y}^{(n)}, 1 \right) : \left(\vec{x}^{(n)}, \vec{y}^{(n)} \right) \in \mathcal{V}, \vec{y}_l^{(n)} = 1 \right\} \tag{10}$$

$$\mathcal{D}_l = \left\{ \left(\vec{x}^{(n)}, \vec{y}^{(n)}, 1 \right) : \left(\vec{x}^{(n)}, \vec{y}^{(n)} \right) \in \mathcal{V}, h_l^{BR}(\vec{x}^{(n)}) = 1 \right\} \tag{11}$$

The above-mentioned classifier responses are related to the binary relevance classifier that can be built without knowing the order of the chain. The classifier is defined using the following classification rule:

$$h_{\pi(i)}^{BR}(\vec{x}) = \underset{y \in \{0,1\}}{\operatorname{argmax}} P(\mathbf{Y}_{\pi(i)} = y) * \prod_{m=1}^{d} P(\vec{\mathbf{X}}_m = \vec{x}_m | \mathbf{Y}_{\pi(i)} = y) \tag{12}$$

Since the neighbourhood of a given instance is defined as a fuzzy set, consistently the above-mentioned sets are also defined as fuzzy. However, the sets are fuzzy singletons. The visualisation of aforementioned sets is provided in Figure 1.

Using the above-mentioned sets we define local True Positive rate, False Positive rate, False Negative rate respectively:

$$\mathrm{TP}_l(\vec{x}) = |\mathcal{V}_l \cap \mathcal{D}_l \cap \mathcal{N}(\vec{x})|, \tag{13}$$

$$\mathrm{FP}_l(\vec{x}) = |(\mathcal{D}_l \setminus \mathcal{V}_l) \cap \mathcal{N}(\vec{x})|, \tag{14}$$

$$\mathrm{FN}_l(\vec{x}) = |(\mathcal{V}_l \setminus \mathcal{D}_l) \cap \mathcal{N}(\vec{x})|, \tag{15}$$

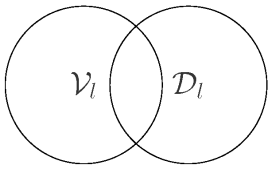

Figure 1: Visualisation of ground truth labels and the decision set of the algorithm.

where $|\cdot|$ is the cardinality of a fuzzy set [7]. Then, we define the local measure of classification quality:

$$F_l(\vec{x}) = \frac{2\mathrm{TP}_l(\vec{x})}{2\mathrm{TP}_l(\vec{x}) + \mathrm{FP}_l(\vec{x}) + \mathrm{FN}_l(\vec{x})} \tag{16}$$

Finally, the label order π is chosen so that the following inequalities are met:

$$F_{\pi(1)}(\vec{x}) \geq F_{\pi(2)}(\vec{x}) \geq \cdots \geq F_{\pi(L)}(\vec{x}). \tag{17}$$

That is labels for whom the classification quality is higher precedes other labels in the chain structure. In other words, this simple heuristic is aimed at dealing with error propagation in the chain structure by employing the most accurate models at the beginning of the chain.

2.5 The Ensemble Classifier

Now, let us define a ML $K - \mathrm{element}$ classifier ensemble: $eH = \{H_1, \ldots, H_K\}$. The ensemble is built using classifier chain algorithms defined in previous sections. Each ensemble classifier is built using a subset of the original dataset. The size of subset is 66% of the original training set.

The BR transformation may produce imbalanced single-label dataset. To prevent the classifier from learning from a highly imbalanced dataset, we applied the random undersampling technique [9]. The majority class is undersampled when imbalance ratio is higher than 20. The goal of undersampling is to keep the imbalance ratio at the level of 20.

The research on the application of Naive Bayes algorithm under the CC framework shows that when the number of features in the input space is significantly higher in comparison to the number of labels the Naive Bayes classifier may not perform well [4]. To prevent the proposed system from being affected by this phenomenon, we applied the feature selection procedure for each single-label separately. That is, the attributes are selected in order to improve the classification quality for given label. The feature selection removes only attributes related to the original input space. Features related to labels are passed through the chain without selection. We employed

the selection procedure based on correlation. In other words, we select attributes that are highly correlated to the predicted label and their inter-correlations are low [14]. Additionally, if the number of selected features is higher than 300, we select 300 random features from the set of previously selected features.

The final prediction vector of the ensemble is obtained via is a simple averaging of response vectors corresponding to base classifiers of the ensemble followed by the thresholding procedure:

$$\widetilde{h}_i(\vec{x}) = \left[\!\!\left[K^{-1} \sum_{k=1}^{K} h_i^k(\vec{x}) > 0.5 \right]\!\!\right], \qquad (18)$$

where $[\![\cdot]\!]$ is the Iverson bracket.

3 Experimental Setup

The conducted experimental study provides an empirical evaluation of the classification quality of the proposed method and compares it to reference methods. Namely, we conducted our experiments using the following algorithms:

1. The proposed approach (Section 2).
2. Static ensemble generated using a genetic algorithm [29]. The enesmble is tuned to optimise the macro-averaged F_1 measure
3. ECC ensemble with randomly generated chain orders [22].
4. OOCC dynamic method proposed by da Silva et al. [4]. The ensemble is tuned to optimise the example based F_1 measure. Additionally, the reference method uses single split into training and validation sets.

In the following sections of this paper, we will refer to the investigated algorithms using the above-said numbers. The reference algorithm also uses Naive Bayes algorithm with data preprocessing procedures described in Section 2.5.

The extraction of training and test datasets was performed using 10 fold cross-validation. For each ensemble, the proportion of the training set $\mathcal{T}_A$ was fixed at $t = 0.6$ of the original training set (see Algorithm 1). For each ensemble, the size of the committee was set to $K = 20$. For the algorithm based on the genetic algorithm, the initial size of the committee was set to $3K$. Each numeric attribute in the training and validation datasets was also standardised. After the standardisation, the mean value of the attribute is 0 and its standard deviation is 1.

The β coefficient was tuned during the training procedure using 3 CV approach. The best value among $\{1, 2, \cdots, 10\}$ is chosen.

Single label classifiers were implemented using WEKA software [13]. Multi-label classifier were implemented using Mulan software [26].

The experiments were conducted using 32 multi-label benchmark sets. The main characteristics of the datasets are summarized in Table 1. We used datasets from the sources abbreviated as follows:A [3], B [21] M–[27]; W–[33]; X–[34]; Z–[37]; T–[28]; O – [18]. Some of the employed sets needed some preprocessing. That is, we used multi-label multi-instance [37] sets (sources Z and W) which were transformed to single-instance multi-label datasets according to the suggestion made by Zhou et al. [37]. Multi-target regression sets (No 9, 31) were binarised using simple thresholding strategy. That is if the response is greater than 0 the resulting label is set relevant. Two of the used datasets are synthetic ones (source T) and they were generated using algorithm described in [28]. To reduce the computational burden, we use only a subset of original Tmc2007 and IMDB sets. Additionally, the number of labels in Stackex datasets is reduced to 15.

The algorithms were compared in terms of 11 different quality criteria coming from three groups [19]: Instance-based (Hamming, Zero-One, F_1, False Discovery Rate, False Negative Rate); Label-based. The last group contains the following measures: Macro Averaged (False Discovery Rate (FDR, 1- Precision), False Negative Rate (FNR, 1-Recall), F_1) and Micro Averaged versions of the above-mentioned criteria.

Statistical evaluation of the results was performed using the Wilcoxon signed-rank test [6, 32] and the family-wise error rates were controlled using the Holm procedure [6, 16]. For all statistical tests, the significance level was set to $\alpha = 0.1$. Additionally, we also applied the Friedman [8] test followed by the Nemenyi post-hoc procedure [6].

4 Results and Discussion

The results of the experimental study are presented in Tables 2 – 1 and Figure 2. Tables 2 and 3 show full results of the experiment. Table 1 provides results of the statistical evaluation of the experiments. Figure 2 visualises the average ranks and provide a view of the Nemenyi post-hoc procedure.

First, let's analyse differences between the proposed heuristic and the simple ECC ensemble. The proposed method is tailored to optimise the macro-averaged F_1 loss so we begin with investigating macro-averaged measures. It is easy to see that both

Table 1: Summarised properties of the datasets employed in the experimental study. Sr denotes the source of dataset, No. is the ordinal number of a set, N is the number of instances, d is the dimensionality of input space, L denotes the number of labels. LC, LD, avIR are label cardinality, label density and average imbalance ratio respectively [19, 2].

No	Name	Sr	N	d	L	LC	LD	avIR
1	Arts1	M	7484	1733	26	1.654	.064	94.74
2	Azotobacter	W	407	20	13	1.469	.113	2.225
3	Birds	M	645	260	19	1.014	.053	5.407
4	Caenorhabditis	W	2512	20	21	2.419	.115	2.347
5	Drosophila	W	2605	20	22	2.656	.121	1.744
6	Emotions	M	593	72	6	1.868	.311	1.478
7	Enron	M	1702	1001	53	3.378	.064	73.95
8	Flags	X	194	43	7	3.392	.485	2.255
9	Flare2	M	1066	27	3	0.209	.070	14.15
10	Genbase	M	662	1186	27	1.252	.046	37.32
11	Geobacter	W	379	20	11	1.264	.115	2.750
12	Haloarcula	W	304	20	13	1.602	.123	2.419
13	Human	X	3106	440	14	1.185	.085	15.29
14	Image	M	2000	294	5	1.236	.247	1.193
15	IMDB	M	3042	1001	28	1.987	.071	24.61
16	LLOG	B	1460	1004	75	1.180	.016	39.27
17	Medical	M	978	1449	45	1.245	.028	89.50
18	MimlImg	Z	2000	135	5	1.236	.247	1.193
19	Ohsumed	O	13929	1002	23	1.663	.072	7.869
20	Plant	X	978	440	12	1.079	.090	6.690
21	Pyrococcus	W	425	20	18	2.136	.119	2.421
22	Reutersk500	B	6000	500	103	1.462	.014	51.98
23	Saccharomyces	W	3509	20	27	2.275	.084	2.077
24	Scene	X	2407	294	6	1.074	.179	1.254
25	SimpleHC	T	3000	30	10	1.900	.190	1.138
26	SimpleHS	T	3000	30	10	2.307	.231	2.622
27	SLASHDOT	B	3782	1079	22	1.181	.054	17.69
28	Stackex_chem	A	6961	540	15	1.010	.067	3.981
29	Stackex_chess	A	1675	585	15	1.137	.076	4.744
30	Tmc2007-500	M	2857	500	22	2.222	.101	17.15
31	water-quality	M	1060	16	14	5.073	.362	1.767
32	yeast	M	2417	103	14	4.237	.303	7.197

Table 2: Full results – micro-averaged criteria. Each row corresponds to the set-specific result under given quality criterion.

	Micro FDR				Micro FNR				Micro F_1			
No.	1	2	3	4	1	2	3	4	1	2	3	4
1	.523	.578	.636	.519	.926	.822	.923	.934	.873	.751	.873	.884
2	.513	.453	.437	.318	.956	.953	.958	.956	.922	.915	.923	.921
3	.469	.603	.541	.413	.636	.674	.633	.648	.573	.648	.596	.563
4	.462	.437	.409	.143	.845	.825	.848	.846	.761	.734	.760	.741
5	.550	.518	.546	.451	.839	.885	.868	.874	.764	.816	.797	.796
6	.392	.381	.389	.377	.279	.268	.277	.288	.340	.330	.338	.336
7	.742	.755	.777	.700	.511	.454	.517	.520	.663	.662	.695	.632
8	.256	.261	.262	.264	.210	.211	.217	.230	.234	.237	.241	.248
9	.628	.547	.640	.602	.673	.664	.691	.678	.659	.616	.674	.649
10	.215	.071	.469	.127	.443	.046	.419	.425	.351	.060	.450	.308
11	.415	.449	.458	.315	.927	.916	.938	.927	.873	.860	.891	.871
12	.303	.436	.355	.341	.874	.865	.867	.882	.790	.785	.785	.803
13	.586	.593	.592	.579	.639	.604	.620	.641	.615	.599	.607	.613
14	.574	.555	.574	.581	.308	.303	.305	.286	.473	.457	.472	.472
15	.692	.759	.724	.646	.926	.888	.907	.909	.881	.849	.863	.857
16	.971	.976	.984	.973	.654	.542	.511	.673	.947	.954	.969	.951
17	.335	.238	.415	.326	.681	.412	.686	.620	.571	.341	.596	.514
18	.482	.479	.489	.480	.455	.472	.458	.445	.469	.476	.474	.463
19	.334	.425	.340	.310	.728	.532	.728	.747	.614	.484	.615	.629
20	.553	.568	.577	.559	.844	.842	.828	.843	.771	.771	.756	.771
21	.448	.479	.530	.306	.958	.931	.933	.956	.923	.883	.883	.917
22	.876	.926	.956	.874	.570	.524	.481	.554	.808	.872	.918	.804
23	.641	.553	.522	.477	.954	.963	.960	.968	.919	.932	.927	.939
24	.402	.393	.403	.409	.198	.211	.195	.194	.315	.314	.315	.318
25	.262	.235	.254	.240	.504	.508	.503	.513	.407	.402	.404	.407
26	.410	.445	.421	.418	.781	.754	.773	.778	.681	.659	.674	.679
27	.235	.335	.239	.208	.841	.620	.834	.856	.738	.516	.728	.757
28	.437	.464	.437	.420	.837	.783	.839	.850	.748	.691	.751	.763
29	.359	.341	.348	.357	.838	.792	.834	.839	.742	.685	.736	.744
30	.389	.406	.390	.361	.316	.306	.312	.325	.355	.360	.354	.344
31	.500	.509	.503	.492	.333	.325	.350	.369	.429	.432	.437	.437
32	.364	.352	.367	.353	.344	.350	.346	.358	.355	.351	.357	.356

methods are comparable in terms of recall but the proposed one is significantly better in terms of precision. It means that the proposed method makes significantly less false positive predictions. Consequently, under the macro-averaged F_1 loss the proposed method outperforms the ECC ensemble. The same pattern is also present in results related to micro-averaged measures. However, the difference for the micro-averaged F_1 measure is not significant. In contrast, under example based measures, except the Hamming loss, there are no significant differences between investigated methods.

The results show that the proposed heuristic provides an effective way of improving classification quality for classifier chains ensemble. Moving the best performing label-specific models at the beginning of the chain reduces the error that propagates along the chain. What is more, the experimental study also showed that the Naive Bayes classifier combined with proper data preprocessing may be effectively employed in classifier chain ensembles.

Now, let's compare the proposed method to the other algorithm based on the dynamic chain approach. When we investigate the example-based criteria it is easy to see that the OOCC algorithm outperforms the proposed one in terms of FDR and Hamming loss. Those results combined with results achieved in terms of macro and micro averaged measures shows that the OOCC mthod seems to be too much conservative. That is, it tends to makes many false negative predictions in comparison to the other methods. The outstanding results for the Hamming loss are a consequence of the imbalanced nature of the multi-label data. That is, the presence of labels is relatively rare and the prediction that contains many false negatives may achieve inadequately hight performance under the Hamming loss [19].

On the other hand, the average ranks clearly show that the method based on genetic algorithm achieves the best results in comparison to the other investigated methods. The main reason is that the GA-based approach optimises the entire ensemble structure, whereas the investigated dynamic chain methods, choose the best label order for single classifier chain. Then the locally chosen chains are combined into an ensemble. It gives us an important clue. That is when we consider an algorithm for dynamic chain order selection, we should think about a single chain and the global structure of the entire ensemble as well.

Table 3: Full results – Example based and macro-averaged criteria.

No.	Hamming				Zero-One				EX FDR				EX FNR				EX F_1				Macro FDR				Macro FNR				Macro F_1			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	.064	.068	.068	.064	.936	.871	.936	.945	.902	.790	.902	.912	.910	.794	.907	.921	.912	.807	.911	.922	.650	.667	.715	.683	.904	.849	.929	.936	.886	.832	.914	.916
2	.112	.112	.111	.111	.553	.558	.555	.546	.540	.538	.536	.526	.527	.530	.527	.525	.536	.537	.534	.528	.905	.917	.922	.883	.948	.945	.952	.952	.941	.939	.943	.937
3	.052	.065	.058	.048	.496	.513	.510	.494	.349	.396	.361	.338	.376	.399	.381	.377	.380	.417	.390	.374	.629	.705	.661	.587	.716	.753	.710	.709	.702	.749	.707	.682
4	.113	.111	.110	.101	.480	.473	.475	.432	.435	.433	.434	.415	.425	.416	.420	.421	.438	.431	.433	.422	.577	.525	.569	.440	.861	.840	.865	.862	.804	.781	.807	.798
5	.125	.122	.124	.118	.473	.483	.481	.472	.439	.456	.456	.437	.421	.447	.441	.437	.440	.456	.455	.444	.645	.664	.696	.596	.843	.888	.871	.878	.794	.843	.829	.824
6	.232	.224	.231	.225	.709	.712	.711	.710	.399	.385	.395	.375	.282	.271	.283	.286	.368	.358	.367	.363	.406	.391	.403	.393	.292	.282	.289	.302	.366	.352	.361	.364
7	.123	.137	.141	.106	.871	.879	.867	.871	.495	.496	.513	.467	.482	.436	.487	.491	.546	.525	.560	.531	.860	.874	.882	.861	.766	.729	.762	.808	.855	.851	.870	.866
8	.234	.238	.241	.246	.789	.803	.814	.799	.264	.269	.272	.274	.218	.219	.226	.236	.252	.256	.260	.267	.296	.302	.298	.310	.302	.304	.306	.326	.318	.322	.323	.339
9	.088	.075	.088	.083	.217	.203	.218	.210	.200	.191	.203	.197	.190	.192	.193	.191	.198	.193	.201	.196	.728	.534	.750	.689	.689	.629	.728	.707	.739	.601	.765	.717
10	.028	.006	.045	.024	.560	.063	.554	.538	.514	.022	.523	.497	.511	.030	.494	.493	.518	.031	.521	.499	.285	.172	.591	.308	.353	.139	.502	.380	.351	.162	.579	.376
11	.113	.113	.112	.111	.565	.573	.573	.562	.546	.554	.546	.538	.536	.545	.544	.539	.544	.549	.552	.545	.885	.875	.902	.877	.926	.922	.936	.926	.917	.909	.928	.912
12	.115	.120	.115	.116	.622	.645	.625	.641	.560	.557	.554	.559	.585	.593	.578	.594	.579	.585	.572	.585	.831	.855	.848	.858	.908	.907	.904	.918	.888	.892	.889	.900
13	.097	.100	.099	.096	.839	.842	.846	.836	.704	.690	.699	.704	.632	.601	.615	.635	.690	.670	.681	.690	.765	.759	.758	.761	.827	.814	.818	.827	.824	.814	.820	.826
14	.307	.290	.307	.316	.874	.844	.872	.881	.542	.517	.541	.544	.266	.259	.263	.242	.466	.447	.464	.462	.491	.483	.495	.496	.292	.288	.288	.272	.454	.440	.453	.452
15	.077	.089	.082	.076	.960	.952	.962	.952	.862	.812	.833	.827	.915	.878	.901	.898	.901	.865	.886	.879	.920	.908	.903	.909	.979	.954	.962	.973	.972	.949	.958	.968
16	.196	.303	.481	.201	.838	.985	.950	.900	.795	.786	.943	.900	.549	.463	.570	.680	.787	.777	.932	.890	.929	.977	.985	.977	.768	.719	.700	.836	.913	.961	.972	.962
17	.023	.017	.026	.022	.777	.534	.796	.722	.657	.424	.687	.593	.686	.429	.698	.620	.683	.438	.705	.619	.508	.475	.591	.479	.544	.500	.616	.521	.533	.496	.608	.511
18	.238	.237	.242	.237	.684	.680	.693	.686	.467	.476	.477	.466	.427	.444	.431	.417	.472	.482	.479	.467	.469	.460	.482	.463	.441	.459	.443	.431	.475	.476	.483	.467
19	.063	.064	.063	.062	.844	.798	.848	.851	.644	.509	.644	.659	.701	.503	.702	.720	.695	.540	.696	.711	.442	.518	.445	.434	.783	.580	.783	.804	.717	.566	.718	.736
20	.094	.095	.096	.094	.875	.872	.870	.877	.848	.847	.833	.848	.840	.836	.825	.838	.847	.845	.834	.847	.816	.803	.824	.821	.905	.901	.896	.901	.884	.877	.878	.880
21	.118	.121	.120	.116	.683	.697	.692	.659	.638	.626	.620	.617	.646	.653	.624	.637	.648	.651	.632	.632	.895	.913	.933	.916	.965	.951	.941	.963	.951	.945	.942	.955
22	.052	.093	.166	.053	.826	.833	.863	.836	.708	.711	.761	.709	.555	.520	.487	.542	.691	.692	.738	.687	.760	.933	.951	.884	.575	.625	.559	.636	.751	.903	.927	.856
23	.087	.085	.085	.084	.482	.476	.486	.476	.463	.457	.468	.457	.549	.464	.474	.467	.471	.465	.475	.466	.772	.807	.792	.824	.955	.963	.959	.968	.930	.948	.947	.957
24	.132	.129	.132	.135	.604	.592	.601	.616	.382	.382	.378	.388	.188	.201	.185	.184	.323	.327	.319	.326	.346	.343	.347	.352	.198	.211	.195	.194	.290	.292	.289	.292
25	.130	.126	.128	.127	.740	.730	.739	.732	.216	.203	.212	.213	.390	.396	.389	.401	.363	.360	.367	.367	.245	.219	.239	.225	.508	.511	.507	.518	.415	.406	.410	.415
26	.216	.220	.216	.217	.952	.942	.952	.951	.553	.512	.546	.547	.782	.753	.773	.778	.724	.693	.716	.719	.781	.710	.724	.785	.894	.876	.887	.893	.894	.878	.885	.895
27	.048	.044	.048	.048	.662	.849	.677	.677	.613	.591	.614	.615	.833	.597	.825	.849	.829	.597	.822	.845	.473	.475	.491	.472	.777	.652	.778	.782	.733	.613	.733	.740
28	.065	.065	.065	.065	.670	.870	.724	.722	.637	.604	.641	.636	.667	.629	.667	.634	.628	.606	.663	.660	.683	.568	.629	.671	.840	.766	.844	.855	.771	.712	.778	.788
29	.070	.068	.070	.070	.724	.701	.724	.722	.637	.604	.641	.636	.667	.630	.667	.666	.662	.628	.663	.660	.683	.568	.629	.671	.898	.837	.890	.897	.853	.783	.840	.850
30	.076	.079	.076	.071	.772	.777	.769	.772	.341	.354	.342	.330	.292	.284	.290	.300	.365	.368	.364	.360	.511	.527	.514	.447	.462	.453	.448	.486	.505	.510	.502	.487
31	.363	.372	.365	.355	.998	1.000	.998	.999	.505	.512	.508	.497	.330	.323	.349	.365	.454	.458	.463	.462	.497	.513	.495	.490	.380	.367	.394	.422	.479	.478	.488	.505
32	.218	.213	.220	.214	.839	.835	.837	.837	.353	.345	.358	.344	.334	.342	.336	.347	.370	.369	.373	.373	.344	.537	.554	.530	.570	.588	.575	.581	.580	.595	.587	.583

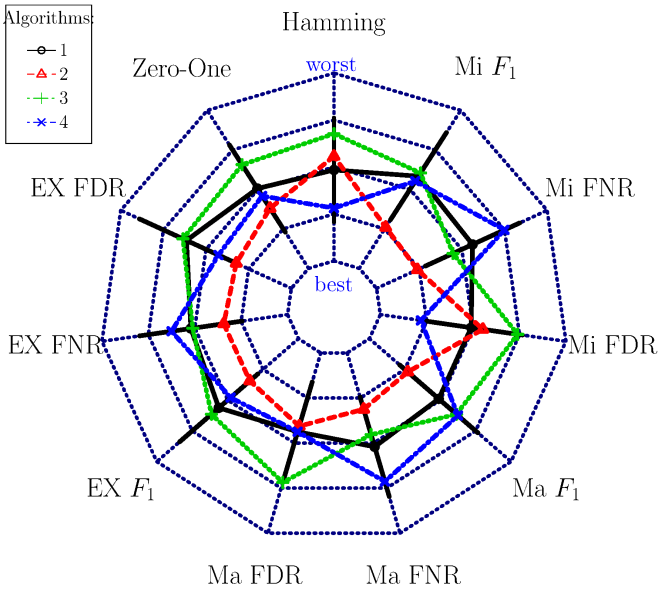

Figure 2: Visualisation of average ranks achieved by algorithms and corresponding critical distances for the Nemenyi post-hoc test. Each axis of the radar plot corresponds to given quality criterion. The closer a point is to the centre of the radar plot, the lower average rank is (lower is better). Black bars parallel to criterion-specific axes denote critical difference for the Nemenyi tests.

5 Conclusions

The main goal of this research was to provide an effective chain classifier that allows changing label order at relatively low computational cost. We achieved it using a classifier based on the Naive Bayes approach. To prove that the proposed method allows handling inter-label relations in an efficient way, we proposed a simple heuristic method that determines label order that should minimise label propagation error. Indeed, the experimental results showed that the proposed method is able to produce a good chain structure at a low computational cost. However, the proposed method of building a dynamic ensemble does not allow to outperform the static system that optimizes the entire ensemble structure. The obtained results are very promising. We believe that there is still a room for improvement. In our opinion, the performance of the system may be improved if we provide better, a better heuristic that optimises the entire ensemble in a dynamic way. The proposed dynamic classifier is a first step in the process of investigating dynamic classifier chain ensembles.

Another way of improving this idea is to build different classifiers that would be able to change the

Table 4: Result of statistical evaluation. Rnk stands for average rank over all datasets, Frd is the p-value obtained using the Friedman test and Wp-i denotes the p-value associated with the Wilcoxon test that compares the i-th algorithm against the others.

Alg. No.	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
	Hamming				Zero-One				EX FDR				EX FNR			
Rnk	2.453	2.672	3.031	**1.844**	2.516	**2.141**	2.969	2.375	2.844	**1.969**	2.906	2.281	2.562	**2.031**	2.531	2.875
Frd	.02113				.19410				.03824				.19410			
Wp-1		0.702	0.025	0.063		0.358	0.199	0.761		0.019	0.295	0.134		0.053	0.700	0.700
Wp-2			0.319	0.319			0.006	0.368			0.005	0.239			0.136	0.015
Wp-3				0.001				0.097				0.040				0.136
	EX F_1				Macro FDR				Macro FNR				Macro F_1			
Rnk	2.688	**2.031**	2.844	2.438	2.312	**2.219**	3.156	2.312	2.562	**1.938**	2.359	3.141	2.469	**1.812**	2.844	2.875
Frd	.19410				.04402				.02113				.02113			
Wp-1		0.066	0.821	0.821		1.000	0.012	1.000		0.035	0.919	0.022		0.156	0.096	0.248
Wp-2			0.017	0.112			0.031	1.000			0.174	0.002			0.003	0.022
Wp-3				0.590				0.012				0.105				0.733
	Micro FDR				Micro FNR				Micro F_1							
Rnk	2.469	2.656	3.219	**1.656**	2.688	**1.719**	2.344	3.250	2.750	**1.781**	2.812	2.656				
Frd	.00027				.00029				.02242							
Wp-1		1.000	0.028	0.000		0.005	0.254	0.239		0.044	0.610	1.000				
Wp-2			1.000	0.008			0.052	0.000			0.002	0.044				
Wp-3				0.000				0.014				1.000				

chain structure without retraining the entire model.

Acknowledgements: This work is financed from Grant For Young Scientists and PhD Students Development, under agreement: 0402/0191/16.

References:

[1] E. Alvares Cherman, J. Metz and M. C. Monard. A Simple Approach to Incorporate Label Dependency in Multi-label Classification. In Advances in Soft Computing. Springer Berlin Heidelberg, 2010. pp. 33–43. doi:10.1007/978-3-642-16773-7_3.

[2] F. Charte, A. Rivera, M. J. del Jesus and F. Herrera. Concurrence among Imbalanced Labels and Its Influence on Multilabel Resampling Algorithms. In Lecture Notes in Computer Science. Springer International Publishing, 2014. pp. 110–121. doi:10.1007/978-3-319-07617-1_10.

[3] F. Charte, A. J. Rivera, M. J. del Jesus and F. Herrera. QUINTA: A question tagging assistant to improve the answering ratio in electronic forums. In IEEE EUROCON 2015 - International Conference on Computer as a Tool (EUROCON). IEEE. doi:10.1109/eurocon.2015.7313677.

[4] P. N. da Silva, E. C. Gonçalves, A. Plastino and A. A. Freitas. Distinct Chains for Different Instances: An Effective Strategy for Multi-label Classifier Chains. In Machine Learning and Knowledge Discovery in Databases. Springer Berlin Heidelberg, 2014. pp. 453–468. doi:10.1007/978-3-662-44851-9_29.

[5] J. Díez, O. Luaces, J. J. del Coz and A. Bahamonde. Optimizing different loss functions in multilabel classifications. *Progress in Artificial Intelligence*, 3(2), (2014), pp. 107–118. doi:10.1007/s13748-014-0060-7.

[6] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7, (2006), pp. 1–30.

[7] M. Dhar. On Cardinality of Fuzzy Sets. *International Journal of Intelligent Systems and Applications*, 5(6), (2013), pp. 47–52. doi:10.5815/ijisa.2013.06.06.

[8] M. Friedman. A Comparison of Alternative Tests of Significance for the Problem of m Rankings. *The Annals of Mathematical Statistics*, 11(1), (1940), pp. 86–92. doi:10.1214/aoms/1177731944.

[9] V. García, J. Sánchez and R. Mollineda. On the effectiveness of preprocessing methods when dealing with different levels of class imbalance. *Knowledge-Based Systems*, 25(1), (2012), pp. 13–21. doi:10.1016/j.knosys.2011.06.013.

[10] E. Gibaja and S. Ventura. Multi-label learning: a review of the state of the art and ongoing research. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(6), (2014), pp. 411–444. doi:10.1002/widm.1139.

[11] E. C. Gonçalves, A. Plastino and A. A. Freitas. Simpler is Better. In Proceedings of the 2015 on Genetic and Evolutionary Computation Conference - GECCO '15. ACM Press. doi: 10.1145/2739480.2754650.

[12] E. C. Goncalves, A. Plastino and A. A. Freitas. A Genetic Algorithm for Optimizing the Label Ordering in Multi-label Classifier Chains. In 2013 IEEE 25th International Conference on Tools with Artificial Intelligence. IEEE. doi: 10.1109/ictai.2013.76.

[13] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten. The WEKA data mining software. *ACM SIGKDD Explorations Newsletter*, 11(1), (2009), p. 10. doi: 10.1145/1656274.1656278.

[14] M. A. Hall. Correlation-based feature selection for machine learning. Ph.D. thesis, The University of Waikato, 1999.

[15] D. J. Hand and K. Yu. Idiot's Bayes: Not So Stupid after All? *International Statistical Review / Revue Internationale de Statistique*, 69(3), (2001), p. 385. doi:10.2307/1403452.

[16] S. Holm. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics*, 6(2), (1979), pp. 65–70. ISSN 03036898. doi:10.2307/4615733.

[17] J.-Y. Jiang, S.-C. Tsai and S.-J. Lee. FSKNN: Multi-label text categorization based on fuzzy similarity and k nearest neighbors. *Expert Systems with Applications*, 39(3), (2012), pp. 2813–2821. doi:10.1016/j.eswa.2011.08.141.

[18] T. Joachims. Text Categorization with Suport Vector Machines: Learning with Many Relevant Features. In Proc. 10th European Conference on Machine Learning. pp. 137–142.

[19] O. Luaces, J. Díez, J. Barranquero, J. J. del Coz and A. Bahamonde. Binary relevance efficacy for multilabel classification. *Progress in Artificial Intelligence*, 1(4), (2012), pp. 303–313. doi: 10.1007/s13748-012-0030-x.

[20] J. Read, L. Martino and D. Luengo. Efficient monte carlo methods for multi-dimensional learning with classifier chains. *Pattern Recognition*, 47(3), (2014), pp. 1535–1546. doi:10.1016/j.patcog.2013.10.006.

[21] J. Read and R. Peter. Meka:http://meka.sourceforge.net/, 2017. URL `http://meka.sourceforge.net/`.

[22] J. Read, B. Pfahringer, G. Holmes and E. Frank. Classifier Chains for Multi-label Classification. In Machine Learning and Knowledge Discovery in Databases. Springer Berlin Heidelberg, 2009. pp. 254–269. doi:10.1007/978-3-642-04174-7_17.

[23] J. Read, B. Pfahringer, G. Holmes and E. Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3), (2011), pp. 333–359. doi:10.1007/s10994-011-5256-5.

[24] C. Sanden and J. Z. Zhang. Enhancing multi-label music genre classification through ensemble techniques. In Proceedings of the 34th international ACM SIGIR conference on Research and development in Information - SIGIR '11. ACM Press. doi:10.1145/2009916.2010011.

[25] R. Senge, J. J. del Coz and E. Hüllermeier. On the Problem of Error Propagation in Classifier Chains for Multi-label Classification. In Studies in Classification, Data Analysis, and Knowledge Organization. Springer International Publishing, 2013. pp. 163–170. doi:10.1007/978-3-319-01595-8_18.

[26] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves and I. Vlahavas. Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, 104(1), (2016), pp. 55–98. doi:10.1007/s10994-016-5546-z.

[27] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves and I. Vlahavas. Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, 104(1), (2016), pp. 55–98. doi:10.1007/s10994-016-5546-z.

[28] J. T. Tomás, N. Spolaôr, E. A. Cherman and M. C. Monard. A Framework to Generate Synthetic Multi-label Datasets. *Electronic Notes in Theoretical Computer Science*, 302, (2014), pp. 155–176. doi:10.1016/j.entcs.2014.01.025.

[29] P. Trajdos and M. Kurzynski. Permutation-Based Diversity Measure for Classifier-Chain Approach. In Advances in Intelligent Systems and Computing. Springer International Publishing, 2017. pp. 412–422. doi:10.1007/978-3-319-59162-9_43.

[30] G. Tsoumakas, I. Katakis and I. Vlahavas. Effective and efficient multilabel classification in domains with large number of labels, 2008. p. 30–44.

[31] Y. Wei, W. Xia, M. Lin, J. Huang, B. Ni, J. Dong, Y. Zhao and S. Yan. HCP: A Flexible CNN Framework for Multi-Label Image Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(9), (2016), pp. 1901–1907. doi:10.1109/tpami.2015.2491929.

[32] F. Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6), (1945), p. 80. doi:10.2307/3001968.

[33] J.-S. Wu, S.-J. Huang and Z.-H. Zhou. Genome-Wide Protein Function Prediction through Multi-Instance Multi-Label Learning. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 11(5), (2014), pp. 891–902. doi:10.1109/tcbb.2014.2323058.

[34] J. Xu. Fast multi-label core vector machine. *Pattern Recognition*, 46(3), (2013), pp. 885–898. doi:10.1016/j.patcog.2012.09.003.

[35] L. Zadeh. Fuzzy sets. *Information and Control*, 8(3), (1965), pp. 338–353. doi:10.1016/s0019-9958(65)90241-x.

[36] P. Zhang, Y. Yang and X. Zhu. Approaching Multi-dimensional Classification by Using Bayesian Network Chain Classifiers. In 2014 Sixth International Conference on Intelligent Human-Machine Systems and Cybernetics. IEEE. doi:10.1109/ihmsc.2014.129.

[37] Z.-H. Zhou, M.-L. Zhang, S.-J. Huang and Y.-F. Li. Multi-instance multi-label learning. *Artificial Intelligence*, 176(1), (2012), pp. 2291–2320. doi:10.1016/j.artint.2011.10.002.