Multi-label Classification via Multi-target Regression on Data Streams

Aljaž Osojnik[1,2(✉)], Panče Panov[1], and Sašo Džeroski[1,2,3]

[1] Jožef Stefan Institute, Jamova cesta 39, Ljubljana, Slovenia
{aljaz.osojnik,pance.panov,saso.dzeroski}@ijs.si
[2] Jožef Stefan IPS, Jamova cesta 39, Ljubljana, Slovenia
[3] CIPKeBiP, Jamova Cesta 39, Ljubljana, Slovenia

Abstract. Multi-label classification is becoming more and more critical in data mining applications. Many efficient methods exist in the classical batch setting, however, in the streaming setting, comparatively few methods exist. In this paper, we propose a new methodology for multi-label classification via multi-target regression in a streaming setting and develop a streaming multi-target regressor iSOUP-Tree, which uses this approach. We experimentally evaluated two variants of the iSOUP-Tree algorithm, and determined that the use of regression trees is advisable over the use model trees. Furthermore, we compared our results to the state-of-the-art and found that the iSOUP-Tree method is comparable to the other streaming multi-label learners. This is a motivation for the potential use of iSOUP-Tree in an ensemble setting as a base learner.

1 Introduction

In recent years, the task of multi-label classification has been very prominent in the data mining research community [8]. It can be seen as a generalization of the ubiquitous multi-class classification task, where instead of a single label, each example is associated with multiple labels. This is one of the reasons why multi-label classification is the go-to approach when it comes to automatic annotation of media, such as images, texts or videos, with tags or genres.

Most research into multi-label classification has been in the batch context, however, strides have also been made to explore multi-label classification in the streaming setting [4,14,16]. The tendency of big data is clear and present in the research community, as well as in the real world. With an appropriate method, the streaming context allows for real-time analysis of large amounts of data, e.g., emails, blogs, RSS feeds, social networks, etc.

However, due to the nature of the streaming setting, there are several constraints that need to be considered. A data stream is potentially infinite sequence of examples, which needs to be analyzed with finite resources, in particular, in finite time and memory. The largest point of divergence from the batch setting is the fact that the underlying concept we are trying to learn, can change at any point. Therefore, the algorithm design is often divided into two parts: (1) learning the stationary concept, and (2) detecting and adapting to it's changes.

N. Japkowicz and S. Matwin (Eds.): DS 2015, LNAI 9356, pp. 170–185, 2015.
DOI: 10.1007/978-3-319-24282-8_15

In this paper, we focus on a method for multi-label classification in the streaming context that learns the stationary concept.

Many algorithms in the literature take the problem transformation approach to multi-label classification, both in the batch and the streaming setting. They transform the multi-label classification problem into several problems that can be solved with off-the-shelf methods, e.g., transformation into an array of binary classification problems. With this transformation, the label inter-correlations can be lost, and, consequently, the predictive performance can decrease.

In this paper, we take a different transformation approach and transform the multi-label classification problem into a multi-target regression problem. Multi-target regression is a generalization of single-target regression, i.e., it is used to predict multiple continuous variables. Many facets of the multi-label classification are also expressed in multi-target regression, e.g., the correlation between labels/variables, which motivated us to experiment with multi-label classification by using multi-target regression methods.

To address the multi-label classification task, we have developed a straightforward multi-label classification via multi-target regression methodology, and used it in a combination with a streaming multi-target regressor (iSOUP-Tree). The generality of this approach is paramount as it allows us to address multiple types of structured output prediction problems, such as multi-label classification and hierarchical multi-label classification, in the streaming setting. In this paper, we show that this approach is a viable candidate for the multi-label classification task on data streams. Furthermore, we explore the multi-target regressor in detail to determine which internal methodology is most appropriate for the task at hand. Finally, we perform comparisons with state-of-the-art methods for multi-label classification in the streaming setting.

The structure of the paper is as follows. First, we present the background and related work (Sect. 2). Next, we present the task of multi-label classification via multi-target regression on data streams (Sect. 3). Furthermore, we present the research questions and the experimental design (Sect. 4). Finally, we conclude with the discussion of the results (Sect. 5), conclusions, and further work (Sect. 6).

2 Background and Related Work

In this section, we review the state-of-the art in multi-label classification, both in the batch and the streaming context. In addition, we present the background of the multi-target regression task, which we use as a foundation for defining the multi-label classification via multi target regression approach.

2.1 Multi-label Classification Task

Stemming from the usual multi-class classification, where only one of the possible labels needs to be predicted, the task of *multi-label classification* (MLC) requires a model to predict a combination of the possible labels. Formally, this means

that for each data instance $\boldsymbol{x}$ from an input space X a model needs to provide a prediction $\hat{y}$ from an output space Y, which is constructed as a powerset of the labelset $\mathcal{L}$, i.e., $Y = 2^{\mathcal{L}}$. This is in contrast to the multi-class classification task, where the output space is simply the labelset $Y = \mathcal{L}$. We denote the real labels of an instance $\boldsymbol{x}$ by y, and a prediction made by a model for $\boldsymbol{x}$ by $\hat{y}(\boldsymbol{x})$ (or $\hat{y}$).

In the batch setting, the problem transformation approach is commonly used to tackle the task of multi-label classification. Problem transformation methods are usually used as basic methods to compare to, and are used in a combination with off-the-shelf base algorithms. The most common approach, called *binary relevance* (BR), transforms a multi-label task into several binary classification tasks, one for each of the possible labels [17]. Binary relevance models have been often overlooked due to their inability to account for label correlations, though some BR methods are capable of modeling label correlations during classification.

Another common problem transformation approach is the *label combination* or *label powerset* (LC), where each subset of the labelset is considered as an atomic label for a multi-class classification problem [18,26]. If we start with a multi-label classification task with a labelset of $\mathcal{L}$, we transform this into a multi-class classification with a labelset $\mathcal{L}' = 2^{\mathcal{L}}$.

Third most common problem transformation approach is the *pairwise clasification*, where we have a binary model for each possible pair of labels [7]. This method performs well in some contexts, but for larger problems the method becomes intractable because of model complexity.

In addition to problem transformation methods, there are also adaptations of the well known algorithms that handle the task of multi-label classification directly. Examples of such algorithms are the adaptation of the decision tree learning algorithm for MLC [27], support-vector machines for MLC [9], k-nearest neighbours for MLC [28], instance based learning for MLC [5], and others.

2.2 Multi-label Classification on Data Streams

Many of the problem transformation methods for the multi-label classification task have also been used in the streaming context. Unlike the batch context, where a fixed and complete dataset is given as input to a learning algorithm, the streaming context presents several limitations that the stream learning algorithm must take into account. The most relevant are [2]: (1) the examples arrive sequentially; (2) there can potentially be infinitely many examples; (3) the distribution of examples need not be stationary; and (4) after an example is processed it is discarded or archived. The fact that the distribution of examples is not presumed to be stationary means that algorithms should be able to detect and adapt to changes in the distribution (*concept drift*). This sub-problem is called *drift detection*.

The first approach to MLC in data streams was a batch-incremental method that trains stacked BR classifiers [14]. Some methods for multi-class classification, such as Hoeffding Trees (HT) [6], have also been adapted to the multi-label classification task [16]. Hoeffding trees are incremental anytime decision trees for learning from data streams that use the notion that a small sample is usually

sufficient for choosing an optimal splitting attribute, i.e., the use of the Hoeffding bound. Bifet et al. [3] also introduced the Java-based Massive Online Analysis (MOA)[1] framework, which also allows for the analysis of concept drift [2] and has become one of the main frameworks for data stream mining. Read et al. [16] proposed the use of multi-label Hoeffding trees with prunned sets (PS) at the leaves (HT_{PS}) and Bifet et al. [4] proposed the use of ensemble methods in this context (e.g., ADWIN Bagging).

Recently, Spyromitros et al. [24] introduced a parameterized windowing technique for dealing with the concept drift in multi-label data in a data stream context. Next, Shi et al. [21] proposed an efficient and effective method to detect concept drift based on label grouping and entropy for multi-label data. Finally, Shi et al. [22] proposed an efficient class incremental learning algorithm, which dynamically recognizes some new frequent label combinations.

2.3 Multi-target Regression

In the same way as was multi-label classification adapted from regular classification, we can look at the multi-target regression task as an extension of the single-target regression task. Multi-target regression (MTR) is the task of predicting multiple numeric variables simultaneously, or, formally, the task of making a prediction $\hat{y}$ from $\mathbb{R}^n$, where n is the number of targets for a given instance $\boldsymbol{x}$ from an input space X.

As in multi-label classification, there is a common problem transformation method that transforms the multi-target regression problem into multiple single-target regression problems. In this case, we consider each numeric target separately and train a single-target regressor for each of them. However, this *local* approach suffers from similar problems as the problem transformation approaches to multi-label classification, e.g., in this case the models do not consider the inter-correlations of the target variables. The task of simultaneous prediction of all target variables at the same time (the *global* approach) has been considered in the batch setting by Struyf and Džeroski [25]. In addition, Appice and Džeroski [1] proposed an algorithm for stepwise induction of multi-target model trees.

In the streaming context, some work has been done on multi-target regression. Ikonomovska et al. [13] introduced an instance-incremental streaming tree-based single-target regressor (FIMT-DD), which utilized the Hoeffding bound. This work was later extended to the multi-target regression setting [12] (FIMT-MT). There has been theoretical debate whether the use of the Hoeffding bound is appropriate [19], however, a recent study by Ikonomovska et al. [11] has shown that in practice the use of the Hoeffding bound produces good results. However, these algorithms had the drawback of ignoring nominal input attributes. Additionally, Shaker et al. [20] introduced an instance-based system for classification and regression (IBLStreams), which can be used for multi-target regression.

[1] http://moa.cms.waikato.ac.nz/, accessed on 2015/05/25.

3 Multi-label Classification via Multi-target Regression

In this section, we present the task of multi-label classification that is solved by transforming the problem into a multi-target regression setting. First, we present the problem formulation that describes the transformation procedure. Second, we describe the implementation of the proposed approach.

3.1 Problem Formulation

The problem transformation methods (see Sect. 2.1) generally transform a multi-label classification task into one, or several, binary or multi-class classification tasks. In this work, we take a different approach and transform a classification task into a regression task. The simplest example of a transformation of this type is to transform a binary classification task into a regression task. For example, if we have a binary target with labels *yes* and *no*, by transforming to the regression setting, we would consider a numeric target to which we would assign a numeric value of 0 if the binary label is *no* and 1 if the binary label is *yes*.

In the same way, we can approach the multi-class classification task. Specifically, if the multi-class target variable is ordinal, i.e., the class labels have a meaningful ordering, we can assign the numeric values from 0 to n to each of the corresponding n labels. This makes sense, since if the labels are ordered, a missclassification of a label into a "nearby" label is better than into a "distant" label. However, if the variable is not ordinal this makes less sense, as any given label is not in a strict relationship with other labels.

To address the multi-label classification task using regression, we transform it into a multi-target regression task (see Fig. 1). This procedure is done in two steps: first we transform the multi-label classification target variable into several binary classification variables, similar as in the BR method. However, instead of training one classifier for each of the binary variables, we further transform the values of the binary variable into numbers. A numeric target corresponding to a given label has a value 1 if the label is present in a given instance, and a value 0 if the label is not present.

For example, if we have a multi-label task with target labels $\mathcal{L} = \{\text{red}, \text{blue}, \text{green}\}$, we transform it into a multi-target regression task with three numeric target variables y_{red}, y_{blue}, $y_{green} \in \mathbb{R}$. If an instance is labeled with

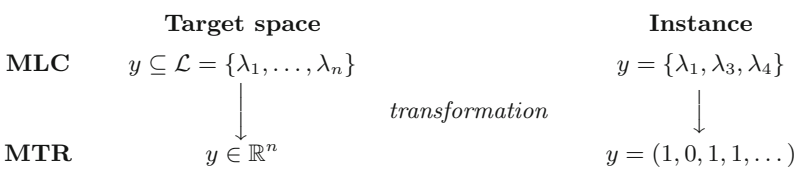


Fig. 1. Transformation from MLC to MTR. Used when the multi-target regressor is learning.

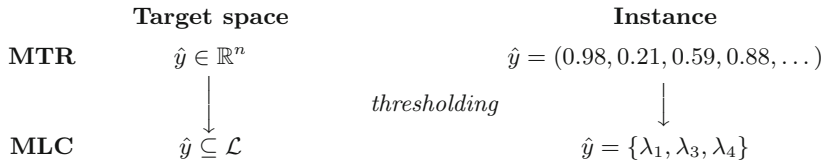


Fig. 2. Transformation from MTR to MLC. Used when transforming a multi-target regression prediction into a mulit-label classification one.

red and green, but not blue, the corresponding numeric targets will have values $y_{red} = 1, y_{blue} = 0$, and $y_{green} = 1$.

Since we are using a regressor, it is possible that a prediction for a given instance will not result in 0 or 1 for each of the targets. For this purpose, we use thresholding to transform back a multi-target regression prediction into a multi-label one (see Fig. 2). Namely, we construct the multi-label prediction in such a way that it contains labels with numeric values over a certain threshold, i.e., in our case, the labels selected are those with a numeric value over 0.5. It is clear, however, that a different choice of threshold leads to different predictions.

In the batch setting, thresholding could be done in the pre- and postprocessing phases, however, in the streaming setting it needs to be done in real time. Specifically, the process of thresholding occurs at two times. The first thresholding occurs when the multi-target regressor has produced a multi-target prediction, which must then be converted into a multi-label prediction. The second thresholding occurs when we are updating the regressor, i.e., when the regressor is learning. Most streaming regressors are heavily dependent on the values of the target variables in the learning process, so the instances must be converted into the numeric representation that the multi-target regressor can utilize.

The problem of thersholding is not only problematic in the MLC via MTR setting, but also when considering the MLC task with other approaches. In general, MLC models produce results which are interpreted as probability estimations for each of the labels, thus the threhsolding problem is a fundamental part of multi-label classifcation.

3.2 Implementation

For the purpose of this work, we have reimplemented the FIMT and FIMT-MT algorithms [12] in the MOA framework to facilitate usability and visibility, as the original implementation was a standalone extension of the C-based VFML library [10] and was not available as part of a larger data stream mining framework. We have also extended the algorithm to consider nominal attributes in the input space when considering splitting decisions. This allows us to use the algorithm on a wider array of datasets, some of which are considered herein.

In this paper, we combined the multi-label classification via multi-target regression methodology with the extended version of FIMT-MT, reimplemented

in MOA. We named this method the incremental Structured OUtput Prediction Tree (iSOUP-Tree), since it is capable of addressing multiple structured output prediction tasks, i.e., multi-label classification and multi-target regression.

Ikonomovska et al. [13] have considered the performance of FIMT-DD when a simple predictive model is placed in each of the leaves, i.e., in this case a single linear unit (a perceptron). Opposed to regular *regression trees* where the prediction in a given leaf for an instance $\boldsymbol{x}$ is made as the average value of the recorded target values, a *model tree* produces the prediction as a linear combination of input attribute values, i.e., $\hat{y}(\boldsymbol{x}) = \frac{1}{|S|}\sum_{y \in S} y$, where S is the set of observed examples in a given leaf, and $\hat{y}(\boldsymbol{x}) = \sum_{i=1}^{m} x_i w_i + b$, where m is the number of input attributes and w_i, b are the perceptron weights, respectively. It was shown that the performance was increased when using model trees, however, this was only experimentally confirmed for regression tasks, where the targets generally exhibit larger variations than in classification tasks.

Specifically, even when considering a classification task through the lens of regression, the actual target variables can only take values of 0 and 1. If we use a linear unit to predict one of the targets, we have no guarantee that the predicted value will land in the [0,1] interval, where as the regression tree will produce an average of zeroes and ones, which will always land in this interval. Additionally, the perceptrons in the leaves are trained in real-time according to the *Widrow-Hoff rule*, which consumes a non-negligible amount of time. This motivated us to consider the use of multi-target regression trees when addressing the task of multi-label classification via multi-target regression. We denote this algorithm variant iSOUP-RT and the model tree variant iSOUP-MT.

4 Experimental Design

In this section, we first present the experimental questions that we want to answer in this paper. Next, we describe the datasets and algorithms used in the experiments. Furthermore, we discuss the evaluation measures used in the experiments. Finally, we conclude with the employed experimental methodology.

Experimental Questions. Our experimental design is constructed in such a way to answer several lines of inquiry. First, we want to explore if the use of model trees improves predictive performance, as it was shown in the regular multi-target regression scenario [13]. Second, we want to compare the introduced methods to other state-of-the-art methods. In this case, we will limit ourselves to comparisons with basic multi-label classification methods. Specifically, this means that we will not be comparing to ensemble or other meta-learning methods, as these methods could potentially utilize the iSOUP-Tree models as base models. Finally, and most crucially, we will consider whether addressing the task of multi-label classification via multi-target regression is a viable approach. For this question, we will use the results from the experiments addressing the previous questions, since they are sufficient.

Table 1. Datasets used in the experiments. N – number of instances, L – number of labels, $\phi_{LC}(D)$ – average number of labels per instance.

Dataset	Enron	IMDB	MediaMill	Ohsumed	Slashdot	TMC
Domain	text	text	video	text	text	text
N	1702	120919	43907	13929	3782	28596
Attribs.	1001 binary	1001 binary	120 numeric	1002 binary	1079 binary	500 binary
L	53	28	101	23	22	22
$\phi_{LC}(D)$	3.4	2.0	4.4	1.7	1.2	2.2

Datasets. In the experiments, we use a subset of datasets listed in [16, Table 3] (see Table 1). The *Enron*[2] dataset [15] is a collection of labelled emails, which, though small by the data stream standards, exhibits some data stream properties, such as time-orderedness and evolution over time. The *IMDB*[3] dataset [16] is constructed from text summaries of movie plots from the Internet Movie DataBase and is labelled with the relevant genres. The *MediaMill* (See footnote 2) dataset [23] consists of video data annotated with various concepts which was used in the TRECVID challenge. The *Ohsumed*[4] dataset [16] was constructed from a collection of peer-reviewed medical articles and labelled with the appropriate disease categories. The *Slashdot* (See footnote 3) dataset [16] was mined from http://slashdot.org web page and consists of article blurbs and is labelled with subject categories. The *TMC* (See footnote 2) dataset was used in the SIAM 2007 Text Mining Competition and consists of human generated aviation safety reports, labelled with the problems being described (we are using the version of the dataset specified in [26]).

Algorithms. To address our experimental questions, we performed experiments using our implementations of algorithms for learning multi-target model trees (**iSOUP-MT**) and multi-target regression trees (**iSOUP-RT**). In addition, to preform comparison with other state-of-the-art algorithms we reuse results of experiments [16], performed under the same experimental settings. These include the following basic algorithms: binary relevance classifier (**BR**), classifier chains (**CC**), multi-label Hoeffding Trees (**HT**) and pruned sets (**PS**).

Evaluation Measures. In the evaluation, we use a set of measures used in recent surveys and experimental comparisons of different multi-label algorithms in the batch setting [8]. These include the following measures: accuracy, Hamming loss, exact match, and ranking loss. Aside from ranking loss, we selected these measures based on the available results for other basic multi-label methods in [16], since we were unable to rerun the experiments with the code made available by the authors. The differences in implementation also disallow for the

[2] http://mulan.sourceforge.net/datasets-mlc.html, accessed on 2015/05/25.
[3] http://meka.sourceforge.net/, accessed on 2015/05/25.
[4] Provided on request by authors of [16].

comparison of running times. However, we will briefly consider the running times of the iSOUP-Tree variants.

In the following definitions, N is the number of examples in the evaluation sample, i.e., the size of one window w, while Q is the number of labels in the provided MLC setting. Accuracy for an example with a prediction set $\hat{y}_i$ and a real labelset y_i is defined as the Jaccard similarity coefficient between them, i.e., $\frac{|\hat{y}_i \cap y_i|}{|\hat{y}_i \cup y_i|}$. The *accuracy* over a sample is the averaged accuracy for each example: $\text{Accuracy} = \frac{1}{N} \sum_{i=1}^{N} \frac{|\hat{y}_i \cap y_i|}{|\hat{y}_i \cup y_i|}$. The higher the accuracy of a model the better its predictive performance.

The *Hamming loss* measures how many times an example-label pair is misclassified. Specifically, each label that is either predicted but not real, or vice versa, carries a penalty to the score. The Hamming loss of a single example is the number of such misclassified labels divided by the number of all labels, i.e., $\frac{1}{Q}|\hat{y}_i \triangle y_i|$ where $\hat{y}_i \triangle y_i$ is the symmetric difference of the $\hat{y}_i$ and y_i sets. The Hamming loss of a sample is the averaged Hamming loss over all examples: $\text{HL} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{Q}|\hat{y}_i \triangle y_i|$. The Hamming loss of a perfect model, which makes completely correct predictions, is 0 and the lower the Hamming loss the better the predictive performance of a model. Note, that the Hamming loss will generally be reported as the *Hamming score*, i.e., $\text{HS} = 1 - \text{HL}$.

The *exact match* measure (also known as subset accuracy or 0/1-loss) is a very strict evaluation measure as it requires the predicted labelset to be identical to the real labelset. Formally, the exact match measure is defined as $\text{EM} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{I}(\hat{y}_i, y_i)$, where $\mathbf{I}(\hat{y}_i, y_i) = 1$, iff $\hat{y}_i$ and y_i are identical. The higher the exact match, the better the predictive performance.

Since thresholding can have a large impact on performance measures and determining the optimal threshold is non-trivial, we are also interested in measures that are independent of the chosen threshold. One such measure is *ranking loss*, defined as $\text{RL} = \frac{1}{N} \sum_{i=1}^{N} \frac{|D_i|}{|y_i||\overline{y}_i|}$, where $\overline{y}_i = \mathcal{L} \setminus y_i$ is the complement of y_i in $\mathcal{L}$, $D_i = \{(\lambda_k, \lambda_l) \,|\, s(\hat{y}_i, k) \leq s(\hat{y}_i, l), (\lambda_k, \lambda_l) \in y_i \times \overline{y}_i\}$ and $s(\hat{y}_i, k)$ is the numeric score (probability) for the label λ_k in the prediction $\hat{y}_i$, before applying the threshold. Essentially, it measures how well the labels are ordered by score, i.e., the loss is low when the labels that aren't present have lower scores than the present labels. Consequently, lower values of ranking loss indicate better performance.

Experimental Setup. Throughout our experiments we use the holdout evaluation approach for data streams. This means that a *holdout set* (or *a window*) of fixed size is constructed once enough examples accumulate, after which the predictions on the holdout set are used to calculate and report the evaluation metrics. Following that, the model is then updated with the collected examples and the process is repeated until all of the examples have been used.

To answer the proposed experimental questions, we constructed the following experimental setup. To compare the predictive performance of iSOUP-MT and iSOUP-RT, we have decided to observe the evolution of the ranking loss over time. Ranking loss was selected as the measure of choice, as it is independent of a

chosen threshold and, as discussed earlier, thresholding is a non-trivial problem to solve in the streaming context. In this case, the desired properties are low ranking loss and/or a strongly declining tendency of the ranking loss, indicating an improvement over time.

For our experiments, we used a window size of $w = \frac{N}{20}$, i.e., each of the streams was divided into 20 windows, and the measures were recorded at each window. This not only allows us to look at the time evolution of the selected measures, but is also identical to the experimental setup from Read et al. [16]. Since we wish to directly compare our results to the results provided therein, we averaged the selected measures over all 20 of the windows.

5 Results and Discussion

In this section, we present the results of the performed experiments that answer our experimental questions. First, we compare the performance of the iSOUP-MT and iSOUP-RT methods on several datasets using a set of evaluation measures. Next, we provide a comparison of our methods with different basic incremental ML methods using results from previous studies. Finally, we provide a discussion of results with a focus on possible improvements to our methodology.

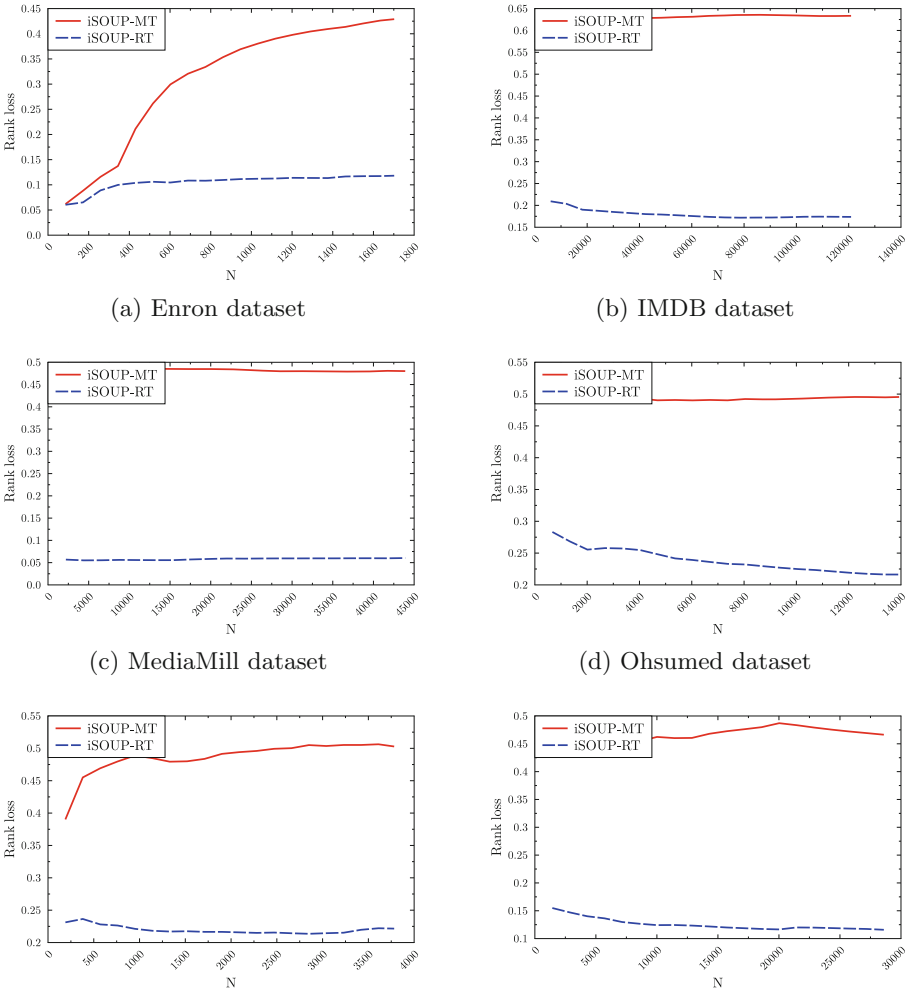
Comparison of iSOUP-MT and iSOUP-RT. In Table 2, we show the comparison of iSOUP-MT and iSOUP-RT on a set of evaluation measures. The results show that with the exception of accuracy on the Slashdot dataset, iSOUP-RT generally achieves better or at least comparable results than iSOUP-MT and clearly uses less time. This indicates that model trees are generally worse than regression trees when using the MLC via MTR methodology. The implementation of iSOUP-MT that uses percetrons in the leaves of the trees should be adapted to capture the dependencies of labels on the input attributes more accurately or a different type of model should be used in their place.

Table 2. Comparison of iSOUP-MT and iSOUP-RT. The best result per dataset is shown in bold. Other than time, the results are an average over 20 windows.

Dataset evaluation measure		Enron	IMDB	MediaMill	Ohsumed	Slashdot	TMC
Exact match	iSOUP-MT	0.165	0.000	0.000	0.000	0.000	0.000
	iSOUP-RT	**0.194**	**0.001**	**0.044**	**0.072**	**0.001**	**0.103**
Hamming score	iSOUP-MT	0.740	0.903	0.560	0.765	0.620	0.516
	iSOUP-RT	**0.945**	**0.929**	**0.966**	**0.979**	**0.947**	**0.912**
Accuracy	iSOUP-MT	0.273	**0.005**	0.047	0.036	**0.065**	0.089
	iSOUP-RT	**0.276**	0.002	**0.346**	**0.114**	0.001	**0.322**
Ranking loss	iSOUP-MT	0.311	0.625	0.483	0.518	0.486	0.465
	iSOUP-RT	**0.105**	**0.180**	**0.058**	**0.250**	**0.220**	**0.126**
Time [s]	iSOUP-MT	15.02	549.97	363.83	96.63	17.60	53.67
	iSOUP-RT	**9.81**	**295.84**	**307.54**	**68.66**	**9.02**	**29.32**

In Fig. 3, we show the ranking loss diagrams, which show the comparison of the iSOUP-MT and iSOUP-RT methods on all 6 datasets used in our experiments. The figures clearly show that the evolution of the ranking loss measure is considerably better for the iSOUP-RT over all datasets. The only dataset where the ranking losses of iSOUP-MT and iSOUP-RT are comparable is the Enron dataset. However, it is a small dataset in terms of data streams, so the windows are small enough that the trees do not have time to significantly grow.

Comparison of Different Incremental Multi-label Methods. In this section, we present the results of the comparison of our methods (iSOUP-MT and



(a) Enron dataset (b) IMDB dataset

(c) MediaMill dataset (d) Ohsumed dataset

(e) Slashdot dataset (f) TMC dataset

Fig. 3. Ranking loss diagrams

Table 3. Exact match measure. The best result per dataset is shown in bold. * marks results reused from [16, Table 6].

Dataset	iSOUP-MT	iSOUP-RT	BR*	CC*	HT*	PS*
Enron	0.168	**0.194**	0.006	0.007	0.058	0.086
IMDB	0.000	0.001	0.031	0.014	**0.108**	0.027
MediaMill	0.000	0.044	0.008	0.000	**0.050**	0.017
Ohsumed	0.000	0.072	0.115	0.054	0.083	**0.212**
Slashdot	0.000	0.001	0.000	0.000	**0.137**	0.113
TMC	0.000	0.076	0.149	0.123	0.087	**0.298**

iSOUP-RT) with other basic incremental multi-label methods. These include: binary relevance classifier (**BR**), classifier chains (**CC**), multi-label Hoeffding Trees (**HT**) and pruned sets (**PS**). Here, we note the results for these methods were reused from Read et al. [16, Tables 5, 6 and 7], because of inability to reproduce the experiments from the software links provided in [16].

In terms of the exact match measure our methods did not often score the best among compared algorithms (see Table 3). In this case, HT performed best on three of the datasets and was followed by PS with best results on two datasets. iSOUP-RT performed best on the Enron dataset. Notably, the results of iSOUP-RT are generally close to those of HT, except for a case where exact match is considerably higher for iSOUP-RT and a case where the opposite holds.

When considering the Hamming loss (presented in Table 4 as the Hamming score), however, iSOUP-RT out matched all other algorithms, except for the TMC dataset. Interestingly, the iSOUP-RT's results here are better aligned with PS's results, and not HT's, as in the case of exact match.

The results for the accuracy measure are less clear (see Table 5). PS performed the best with best results on three of the datasets, iSOUP-RT outperformed other algorithms in two cases and HT performed best on the IMDB dataset.

Table 4. Hamming loss (displayed as $1.0 - loss$). The best result per dataset is shown in bold. * marks results reused from [16, Table 7].

Dataset	iSOUP-MT	iSOUP-RT	BR*	CC*	HT*	PS*
Enron	0.740	**0.945**	0.524	0.503	0.926	0.934
IMDB	0.903	**0.929**	0.884	0.834	0.918	0.875
MediaMill	0.560	**0.966**	0.897	0.634	0.958	0.947
Ohsumed	0.765	**0.979**	0.913	0.866	0.900	0.947
Slashdot	0.620	**0.947**	0.055	0.054	0.915	0.912
TMC	0.516	0.912	0.907	0.871	0.884	**0.935**

Table 5. Accuracy. The best result per dataset is shown in bold. * marks results reused from [16, Table 5].

Dataset	iSOUP-MT	iSOUP-RT	BR*	CC*	HT*	PS*
Enron	0.273	**0.276**	0.144	0.142	0.134	0.241
IMDB	0.005	0.002	0.139	0.170	**0.210**	0.146
MediaMill	0.047	**0.346**	0.119	0.080	0.301	0.297
Ohsumed	0.036	0.114	0.297	0.292	0.125	**0.372**
Slashdot	0.065	0.001	0.054	0.054	0.145	**0.200**
TMC	0.089	0.322	0.415	0.446	0.171	**0.562**

Discussion. The results clearly indicate that the regression tree variant iSOUP-RT is a more appropriate method for the task of MLC via MTR than the model tree variant iSOUP-MT. This indicates that the perceptrons placed in the leaves significantly reduce the method's performance. This may be due to the mechanism of the perceptron, which does not guarantee that the result will land in the [0, 1] interval. Other types of leaf models should be considered and evaluated in the future, similar to [16] where the pruned sets (PS) method was used in the leaves of the Hoeffding trees.

A cursory glance makes it clear that there is a lot of variation in the majority of the results reported in the comparison of different methods. The exact match measure and accuracy fluctuate to a large extent and only the results of Hamming loss are consistent. However, with respect to the Hamming loss, the iSOUP-RT method consistently outperformed other methods, which possibly indicates that the learning mechanism is biased toward optimization of a similar measure.

Given the relatively small selection of evaluation measures and the observed variation among them, it would be prudent to consider other evaluation measures in a more in-depth experimental evaluation. This variation in the results is something that would be out of place in a more classical machine learning setting, however, there are many partially unexplored variables in the MLC context, e.g., drift-detection, thresholding, etc. Looking at the selected datasets also does not give us sufficient data to determine and analyze the effect of data set size on the performance of the various methods. Overall, we have shown that the MLC via MTR methodology is a valid approach for MLC. However, the use of perceptrons as models in the tree leaves is not advisable and other types of models should be considered. We have determined that iSOUP-RT's performance is similar to the other basic incremental multi-label learners. Therefore, iSOUP-RT is a suitable candidate for further experimentation, e.g., as a base model in ensemble methods explored in [16].

6 Conclusion and Future Work

In this paper, we have introduced the multi-label classification via multi-target regression methodology and introduced the iSOUP-Tree algorithm that is used to address the multi-label classification task.

We performed two sets of experiments, the first of which was designed to evaluate whether the use of model trees over regression trees increases the predictive performance as it was shown for the streaming multi-target regression task [13]. We observed the time evolution of the ranking loss, as well as the average ranking loss, exact match, Hamming loss and accuracy measures over the considered datasets. From these experiments, it was made clear that regression trees outperform model trees for the task of MLC via MTR.

The second set of experiments were designed to compare the introduced methods to other multi-label learners. To this end, the experimental design was equal to the one in [16]. While we were not able to establish clear superiority of one method over the other, we were able to determine that the introduced iSOUP-Tree method is a promising candidate for further experimentation, e.g., as a base model in state-of-the-art ensemble or other meta-learning techniques.

Additionally, due to the relatively unexplored nature of the streaming multi-label classification task, we plan to perform a more extensive experimental evaluation on more datasets and with respect to a wider set of evaluation measures. Specifically, we also wish to address the problems of drift detection and thresholding for the iSOUP-Tree method.

We also propose two other avenues of further work, in regards to extending the introduced methodology. The first one focuses on the model and the aim is to extend the iSOUP-Tree method using the option tree paradigm [11], used in the single-target regression setting, to the multi-target regression setting. This approach has been shown to outperform the regression tree methodology. The second extension is specific to the MLC via MTR methodology. In classical (batch) data mining, the task of hierarchical multi-label classification (HMC) is becoming more and more prevalent. In HMC, the labels are ordered in a hierachy and adhere to the hierarchy constraint, i.e., if an example is labeled with a label it also has to be labelled with the label's ancestors. We plan to extend the MLC via MTR methodology to be applicable to HMC tasks in the streaming setting.

Acknowledgements. We would like to acknowledge the support of the EC through the projects: MAESTRA (FP7-ICT-612944) and HBP (FP7-ICT-604102), and the Slovenian Research Agency through a young researcher grant and the program Knowledge Technologies (P2-0103).

References

1. Appice, A., Džeroski, S.: Stepwise Induction of Multi-target Model Trees. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 502–509. Springer, Heidelberg (2007)
2. Bifet, A., Gavaldà, R.: Adaptive Learning from Evolving Data Streams. In: Adams, N.M., Robardet, C., Siebes, A., Boulicaut, J.-F. (eds.) IDA 2009. LNCS, vol. 5772, pp. 249–260. Springer, Heidelberg (2009)
3. Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B.: Moa: massive online analysis. J. Mach. Learn. Res. **11**, 1601–1604 (2010)

4. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., Gavaldà, R.: New ensemble methods for evolving data streams. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 139–148. ACM (2009)
5. Cheng, W., Hüllermeier, E.: Combining instance-based learning and logistic regression for multilabel classification. Mach. Learn. **76**(2–3), 211–225 (2009)
6. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 71–80. ACM (2000)
7. Fürnkranz, J., Hüllermeier, E., Mencía, E.L., Brinker, K.: Multilabel classification via calibrated label ranking. Mach. Learn. **73**(2), 133–153 (2008)
8. Gibaja, E., Ventura, S.: A tutorial on multilabel learning. ACM Comput. Surv. (CSUR) **47**(3), 52 (2015)
9. Gonçalves, E.C., Plastino, A., Freitas, A.A.: A genetic algorithm for optimizing the label ordering in multi-label classifier chains. In: IEEE 25th International Conference on Tools with Artificial Intelligence (ICTAI), 2013, pp. 469–476. IEEE (2013)
10. Hulten, G., Domingos, P.: VFML - a toolkit for mining high-speed time-changing data streams (2003). http://www.cs.washington.edu/dm/vfml/
11. Ikonomovska, E., Gama, J., Džeroski, S.: Online tree-based ensembles and option trees for regression on evolving data streams. Neurocomputing **150**, 458–470 (2015)
12. Ikonomovska, E., Gama, J., Džeroski, S.: Incremental multi-target model trees for data streams. In: Proceedings of the 2011 ACM Symposium on Applied Computing, pp. 988–993. ACM (2011)
13. Ikonomovska, E., Gama, J., Džeroski, S.: Learning model trees from evolving data streams. Data Min. Knowl. Disc. **23**(1), 128–168 (2011)
14. Qu, W., Zhang, Y., Zhu, J., Qiu, Q.: Mining Multi-label Concept-Drifting Data Streams Using Dynamic Classifier Ensemble. In: Zhou, Z.-H., Washio, T. (eds.) ACML 2009. LNCS, vol. 5828, pp. 308–321. Springer, Heidelberg (2009)
15. Read, J.: A pruned problem transformation method for multi-label classification. In: Proceedings of the 2008 New Zealand Computer Science Research Student Conference (NZCSRS 2008), pp. 143–150 (2008)
16. Read, J., Bifet, A., Holmes, G., Pfahringer, B.: Scalable and efficient multi-label classification for evolving data streams. Mach. Learn. **88**(1–2), 243–272 (2012)
17. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Mach. Learn. **85**(3), 333–359 (2011)
18. Read, J., Pfahringer, B., Holmes, G.: Multi-label classification using ensembles of pruned sets. In: Eighth IEEE International Conference on Data Mining, 2008, ICDM 2008, pp. 995–1000. IEEE (2008)
19. Rutkowski, L., Pietruczuk, L., Duda, P., Jaworski, M.: Decision trees for mining data streams based on the McDiarmid's bound. IEEE Trans. Knowl. Data Eng. **25**(6), 1272–1279 (2013)
20. Shaker, A., Hüllermeier, E.: IBLStreams: a system for instance-based classification and regression on data streams. Evolving Syst. **3**(4), 235–249 (2012)
21. Shi, Z., Wen, Y., Feng, C., Zhao, H.: Drift detection for multi-label data streams based on label grouping and entropy. In: 2014 IEEE Data Mining Workshop (ICDMW), pp. 724–731. IEEE (2014)
22. Shi, Z., Xue, Y., Wen, Y., Cai, G.: Efficient class incremental learning for multi-label classification of evolving data streams. In: International Joint Conference on Neural Networks (IJCNN), 2014, pp. 2093–2099. IEEE (2014)

23. Snoek, C.G., Worring, M., Van Gemert, J.C., Geusebroek, J.M., Smeulders, A.W.: The challenge problem for automated detection of 101 semantic concepts in multimedia. In: Proceedings of the 14th Annual ACM International Conference on Multimedia, pp. 421–430. ACM (2006)
24. Spyromitros-Xioufis, E.: Dealing with concept drift and class imbalance in multi-label stream classification. Ph.D. thesis, Aristotle University of Thessaloniki (2011)
25. Struyf, J., Džeroski, S.: Constraint Based Induction of Multi-objective Regression Trees. In: Bonchi, F., Boulicaut, J.-F. (eds.) KDID 2005. LNCS, vol. 3933, pp. 222–233. Springer, Heidelberg (2006)
26. Tsoumakas, G., Vlahavas, I.P.: Random k-Labelsets: An Ensemble Method for Multilabel Classification. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 406–417. Springer, Heidelberg (2007)
27. Vens, C., Struyf, J., Schietgat, L., Džeroski, S., Blockeel, H.: Decision trees for hierarchical multi-label classification. Mach. Learn. **73**(2), 185–214 (2008)
28. Zhang, M.L., Zhou, Z.H.: A k-nearest neighbor based algorithm for multi-label classification. In: 2005 IEEE Granular Computing, vol. 2, pp. 718–721. IEEE (2005)