Weighted Ensemble Classification of Multi-label Data Streams

Lulu Wang[1], Hong Shen[2,3](✉), and Hui Tian[4]

[1] School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
llwang_14@bjtu.edu.cn

[2] School of Data and Computer Science, Sun Yat-Sen University, Guangzhou, China
hongsh01@gmail.com

[3] School of Computer Science, University of Adelaide, Adelaide, Australia

[4] School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing, China
htian@bjtu.edu.cn

Abstract. Many real world applications involve classification of multi-label data streams. However, most existing classification models mostly focused on classifying single-label data streams. Learning in multi-label data stream scenarios is more challenging, as the classification systems should be able to consider several properties, such as large data volumes, label correlations and concept drifts. In this paper, we propose an efficient and effective ensemble model for multi-label stream classification based on ML-KNN (Multi-Label KNN) [31] and propose a balance AdjustWeight function to combine the predictions which can efficiently process high-speed multi-label stream data with concept drifts. The empirical results indicate that our approach achieves a high accuracy and low storage cost, and outperforms the existing methods ML-KNN and SMART [14].

Keywords: Multi-label · Data stream · Classification

1 Introduction

Due to the recent advances in computer networks and data storage, many data are produced and accumulated at an ever increasing rate in the form of stream. Such as online shopping information, logistics information, online news, stock market data, emails, credit card transactions, *etc.* These data are real-time, continuous and orderly arrival, and need to be analyzed promptly and effectively. For example, in online mail systems, incoming emails need to be classified into different categories, like spams, business emails, personal emails, important emails, *etc.* This classification task, each stream example is associated with a single class label l from a set of labels L ($|L| > 1$), is called single-label data stream classification, and has been extensively studied [3,4,11,20,33] in the literature.

DOI: 10.1007/978-3-319-57529-2_43

In many emerging applications, each stream record may carry multiple class labels. A good example is news reports in the online news systems, most news reports carry multiple news topics (e.g. entertainment, financial and politics), then this is called **M**ulti-**L**abel data **S**tream **Classification** (**MLSC**) [21]. Formally, the multi-label stream classification problem is to training a model to attach a label subset $Y \subseteq L$ to each instance in a high-speed data stream. Although, multi-label classification has been studied in traditional database mining scenarios, multi-label data stream classification is a relatively new concept and has not been fully addressed yet.

An intuitive approach to solving the multi-label classification problem is to transform it into one or more single-label classification problems. In this fashion, traditional single-label classifiers can be employed to make single-label classifications. Finally, the multi-label predictions can be produced by combining these multi-label predictions. On the other hand, an alternative category is to adopt the existing single-label classifiers directly to multi-label classification [18,23].

The multi-label data stream environment has different challenges from the traditional batch learning setting. As the instances in a multi-label data stream contain multi-labels (multi-concepts), dealing with the concept drift is the most important challenge to a classifier. Another challenge with regard to MLSC is that, it is possible that an arriving example will belong to a set of labels, some of which, will not have been previously observed because of the dynamic nature of the set of labels [21]. Besides, the learner must be able to handle the stream using limited memory in real time, because stream data flood in continuously at a high speed, which makes it impossible to be stored in the memory and processed offline [14].

To address the above issues, in this paper we propose an efficient and effective ML-KNN-based ensemble model for multi-label stream classification with a balance AdjustWeight function, called **S**treaming **W**eighted **ML-KNN**-based **E**nsemble **C**lassifier (SWMEC). More specifically, we first propose an ML-KNN based algorithm to build the basic classifier C_i. The ensemble classifiers $C =< C_1, C_2, \cdots, C_L >$ will be build at the beginning of the stream with randomly selected L test data chunks. This only needs to compute and save a small amount of information of the cluster center points. Thus the building process is highly efficient while consuming constant memory space. In addiction, each classifier C_i has its own weight w_i, and $C =< (C_1, w_1), (C_2, w_2), \cdots, (C_L, w_L) >$. As data flow in, the weights will be adjusted and the classifier C will be updated according to the weights. Thus the proposed SWMEC approach can work adaptively to evolving data and deal with concept drifts, and can efficiently classify the incoming data in real-time.

The main contributions of this paper are as follows. (1) an adaption of the existing multi-label methods to evolving data streams. (2) an effective weighting adjustment strategy for ensemble classifiers. (3) experimental results validating the performance of our method and benchmarks in predictive performance and space complexity.

The remainder of this paper is organized as follows. In Sect. 2, we discuss relevant work in multi-label classification and stream classification. Section 3 presents the n preliminaries about the problem. Section 4, describes the proposed framework in details, and the experimental results are presented in Sect. 5. Finally, Sect. 6 concludes the paper.

2 Related Work

Our work is related to multi-label classification and stream classification techniques. We will briefly review the existing work on both of them.

Multi-label classification is the problem to deal with such instances that may belong to multiple different classes simultaneously and focuses on offline settings [16,19]. Multi-label classification methods can be grouped into two categories, namely, *problem transformation* and *algorithm adaptation* [21]. Problem transformation methods transform the multi-label problem into multiple single-label problems. Problem transformation methods transform the multi-label problem into multiple single-label problems including Label Power-set (LP), Binary Relevance (BR) and Ranking by Pairwise Comparison (RPC [12]). Algorithm adaptation methods extend the traditional learning techniques to multi-label context, such as decision trees [5], neural networks [30], maximal margin methods [10], maximum entropy methods [25], and ensemble methods [25], etc. One well-known such approach is ML-KNN [31], which is derived from the popular lazy learning algorithm kNN. It's the most relevant approach to our model and will be introduced in the next section.

Many studies have also been done on single-label stream classification. There are two sets of solutions: single-model based and ensemble based. Single-model based approaches [1,2,6,9,26,27,32,33] use new data to incrementally update their model so that the model can scale to a large data volume. Ensemble based approaches [13,22,28], on the other hand, partition the data stream into equal sized chunks, and train multiple base models on different chunks of data. Then all the models are combined for prediction. The ensemble based approaches are easier to scale and parallelize, tend to achieve better accuracy and can also avoid over fitting than single classifier methods.

Recently, there are also some studies focusing on multi-label stream classification [14,15,17,18,21,29]. Kong et al. [14] builds an ensemble of classifiers on successive data chunks. It proposes a random-tree based algorithm to improve it's efficiency. Work also has been done on adopting the ensemble based strategy in handling multi-label streams [15,29].

We follow a similar strategy to design our classification model with ML-KNN-based ensemble methods. Our model builds streaming classifiers by extending MLkNN, which is just designed for multi-label static data classification.

3 Preliminaries

We first introduce the notations that will be used throughout this paper, and then briefly describe the techniques ML-KNN [31] to make this paper self-contained.

Consider a multi-label stream S with a label set $\mathcal{L} = \{1, 2, \cdots, q\}$, $S \subseteq \mathbb{R}^d$. Stream S consists of infinite data chunks, $\{D_1, \cdots D_n, \cdots\}$, where labeled chunk D_i is denoted by D_L and unlabeled chunk denoted by D_U. Each instance $x \in S$ has a label subset $Y = \{y_1, y_2, \cdots, y_q\} \in \{-1, 1\}^q$, where $Y[j] = \begin{cases} 1 & \text{if } y_j \in Y \\ -1 & \text{if } y_j \notin Y \end{cases}$, (x_i, Y_i) is an instance in the multi-label data stream. The task of multi-label stream classification based on ensemble solution is to train a classification model on the historical examples $F(\cdot)$, $F(\cdot) = g(f_1(\cdot), f_2(\cdot), \cdots, f_L(\cdot))$, where $f_i(\cdot)$ is the sub-classifier, $g(\cdot)$ is the combination function that combines the outputs of all $f(\cdot)$, L is the number of classifiers. Then it uses $F(\cdot)$ to predict a label set Y_i to the incoming data x_i.

Table 1. Summary of major mathematical notations

Notations	Mathematical meanings
S	Multi-label data stream with d-dimensional space $\mathbb{R}^d$
D	data chunk with size N
D_L	Labeled data chunk
D_U	Unlabeled data chunk
x	d-dimensional feature vector $(x_1, x_2, \cdots x_d)^\top (x \in S)$
$\mathcal{L}$	label space with q possible class labels $\{1, 2, \cdots, q\}$
Y	label subset associated with x $(Y \subseteq \mathcal{L})$

3.1 ML-KNN [31]

We briefly describe our model's basic approach ML-KNN, which is derived from the traditional k-Nearest Neighbor (kNN) algorithm and classify the traditional static multi-label data in a lazy learning way. There are three main steps in this approach. For convenience, several notations are summarized in Table 1. In addition, given a training set $T = \{(x_1, Y_1), (x_2, Y_2), \cdots, (x_n, Y_n)\} \left(x_i \in \mathbb{R}^d, Y_i \in \mathcal{L}\right)$, t is the test instance, s is the smoothing parameter with a default value 1.

Step 1: Computing the prior probabilities $P\left(H_b^l\right)$ according to Eqs. 1 and 2. Where $l \in Y$ is the l-th label, and $b \in$ {0,1}, H_1^l represents the event the instance has label l. Conversely, H_0^l means the instance does not have label l.

$$P\left(H_1^l\right) = \left(s + \sum_{i=1}^{n} L_t(l)\right) / (s \times 2 + n) \tag{1}$$

$$P\left(H_0^l\right) = 1 - P\left(H_1^l\right) \tag{2}$$

Step 2: Computing the posterior probabilities $P\left(E_j^l \mid H_b^l\right)$ according to Eqs. 3 and 4. Where $E_j^l\,(j \in \{0, 1, \cdots, k\})$ denotes the event that, among the k nearest neighbors of t, there are exactly j instances which have label l. $c[j]$ counts the number of training instances with label l whose k nearest neighbors contain exactly j instances with label l. Correspondingly, $c^{'}[j]$ counts the number of training instances without label l whose k nearest neighbors contain exactly j instances with label l.

$$P\left(E_m^l \mid H_1^l\right) = \left(s + c\left[j\right]\right) / \left(s \times (k+1) + \sum_{p=0}^{k} c\left[p\right]\right) \tag{3}$$

$$P\left(E_m^l \mid H_0^l\right) = \left(s + c^{'}\left[j\right]\right) / \left(s \times (k+1) + \sum_{p=0}^{k} c^{'}\left[p\right]\right) \tag{4}$$

Step 3: Computing the output y_t (label subset) and r_t of t, where r_t is a real-valued vector calculated to rank labels in $\mathcal{L}$, according to Eqs. 5 and 6.

$$\vec{y_i}\left(l\right) = \arg\max_{b \in \{0,1\}} P\left(H_b^l\right) P\left(E_{C_t(t)}^l \mid H_b^l\right) \tag{5}$$

$$\begin{aligned} \vec{r_i}\left(l\right) &= P\left(H_1^l \mid E_{C_t(l)}^l\right) \\ &= \left(P\left(H_1^l\right) P\left(E_{C_t(l)}^l \mid H_1^l\right)\right) / P\left(E_{C_t(l)}^l\right) \end{aligned} \tag{6}$$

The detailed architecture of ML-KNN was given in [31].

4 Weighted Ensemble Classification

In this section we first give the main idea of our weighted ensemble classification approach, then we introduce the process of the ensemble classifiers' training and updating and the adjustment of their weights, finally we give the description of our algorithm.

4.1 Basic Idea

The data stream S is divided into a fixed number of chunks and each classification model in the ensemble is trained from a different chunk. Each classifier in the ensemble has it's own weight. The new arriving unlabeled data chunk is classified by the ensemble while the corresponding weight will be changed. According to the weights, the latest classified data chunk will be decided if to be trained to generate a new model and replace one of the existing models in the ensemble. In this way the ensemble can be maintained at a fixed size and kept up-to-date. The problems of data stream's infinite length and concept-drift can correspondingly be well addressed.

4.2 Classifier Training and Updating

The ML-KNN based multi-label ensemble classifier is built at the beginning of the data stream and timely updated over the data stream as follow: when a training data chunk in the data stream is arriving at time t, we build h clusters with the labeled data points by the application of XMeans technique. After the building of these clusters, we save each cluster's centroid $O = \{o_1, o_2, \cdots, o_h\}, o_i \in \mathbb{R}^d, i \in \{1, 2, \cdots, h\}$, and compute each centroid's label subset in the same way as ML-KNN. After that, all centroids and their label subset $C =< (o_1, y_1, r_1), (o_2, y_2, r_2), \cdots, (o_h, y_h, r_h) >, y_j \in Y$, y_i,r_j are respectively the label subset and the a real-valued vector calculated by ML-KNN, will be saved as a summary. At the same time, the current summary's arriving time t will also be recorded. Each summary's weight w then can be calculated according to o, $\overrightarrow{r}$ and t, and set to be 1 at beginning. After the process of L chunks, the ensemble classifier $C =< (C_1, w_1), (C_2, w_2), \cdots, (C_L, w_L) >$ (each model C_i is a collection of h summaries and the number of h is unfixed, w_i is the weight of model C_i) will be built. When an unlabeled data chunk D_U is arriving, for each instance $x_i \in D_U$, we find the $m_i = ((o_j, y_j), w_i) \in C_i, j \in [1, 2, \cdots h]$ whose centroid o_j is nearest from x_i. The corresponding weight w_i will then be determined by the distance between o_j and x_i, $\overrightarrow{r_i}$ and t_i. Then the x_i will be labeled according to the summaries $\{m_1, m_2, \cdots, m_L\}$. Each unlabeled data chunk will be classified as above. After a chunk D_U has been handled by the ensemble C, If the lowest $w_{lowest} \in w$ falls below a threshold value ϵ, the corresponding model C_j will be replaced by the new model that trained by D_U. This ensures that the ensemble will be updated with the passing of data chunk and the number of models in the ensemble remains constant.

4.3 Ensemble Weighting

In this paper, inspired by [8], we propose a combination function $g(\cdot)$ including three components ($\overrightarrow{\alpha_i}$, β_i and γ_i) to combine classifiers in the ensemble. They are: (1) label confidence, which is a vector measuring the confidence in the subclassifier outputting each label; (2) time difference; (3) distance difference, both (2) and (3) describe how confident a sub-classifier is when making a classification decision.

We estimate $\overrightarrow{\alpha_i}$ from the ensemble model C by Eq. 7:

$$\overrightarrow{\alpha_i} = \frac{\overrightarrow{r_i}}{\sum_{i=1}^{L} \overrightarrow{r_i}} \tag{7}$$

where β_i, γ_i describe how confident a sub-classifier is when making a classification decision. β_i is estimated by the time difference between arriving new data chunk D_U and the sub-classifier C_i. $\triangle t_i = t_{D_U} - t_{C_i}$, where t_{D_U},t_{C_i} are the arriving times of data chunk D_U and D_i respectively. The longer the chunks are apart, the lower the β is.

$$\beta_i = \frac{e^{-\triangle t_i}}{\sum_{i=1}^{L} e^{-\triangle t_i}}; 0 < \beta_i < 1, \sum_{i=1}^{L} \beta_i = 1 \tag{8}$$

γ_i is estimated by the distance difference between x_i which is the instance from the arriving new data chunk D_U and it's nearest center o_j in sub-classifier C_i.

$$\gamma_i = \frac{e^{-d_i}}{\sum_{i=1}^{L} e^{-d_i}}; 0 < \gamma_i < 1; \sum_{i=1}^{L} \gamma_i = 1 \tag{9}$$

where $d_i = distance\left(x_i, o_j\right)$, the closer the distance between x_i and o_j the more confident the sub-classifier.

Finally, we combine $\vec{\alpha_i}$, β_i and γ_i to decide the weight $w_i = \vec{\alpha_i} \times \beta_i \times \gamma_i$. Consequently:

$$g\left(\cdot\right) = \sum_{i=1}^{L} \left(f_i\left(x_i\right) \times \vec{\alpha_i} \times \beta_i \times \gamma_i\right) \tag{10}$$

The output $Y = \{y_1, y_2, \cdots, y_q\} \in \{-1, 1\}^q$,

$$Y\left[j\right] = \begin{cases} 1 & \text{if } g\left(\cdot\right)\left[j\right] = \sum_i^L \left(f_i\left(\cdot\right)\left[j\right] \times \alpha_i \times \beta_i \times \gamma_i\right) > 0 \\ -1 & \text{if } g\left(\cdot\right)\left[j\right] = \sum_i^L \left(f_i\left(\cdot\right)\left[j\right] \times \alpha_i \times \beta_i \times \gamma_i\right) < 0 \end{cases}.$$

4.4 The Classification Algorithm

The pseudo code of our method for classifying multi-label data streams using the ML-KNN-based ensemble method is given in Algorithm 1.

Algorithm 1. KNN-based ensemble classification for multi-label data streams

```
Input:   Data Stream $S =< D_1, D_2, \cdots, D_n, \cdots >$;
         Initial ensemble classifiers:
         $C =< (C_1, w_1), (C_2, w_2), \cdots, (C_L, w_L) >$;
         Empty $buf$ ;
         Latest chunk of unlabeled instances $D_U$;
         Latest $r$ labeled data chunks $D_r$;
Output:  Classification Result.
 (1)  while true do
 (2)      for all $x_i \in D_U$ do
 (3)          Classification$(C, x_i) = D_L \rightarrow buf$
 (4)      end for
 (5)      Evaluation&Adjust$(C, D_U)$
 (6)      if the lowest $w_{lowest} \in (w_1, w_2, \cdots, w_L) < threshold\ \ \epsilon$ then
 (7)          $buf$ replaces the corresponding data chunk in $D_r$
 (8)          Update$(C, D_r)$
 (9)      end if
(10)      new chunk of unlabeled data $\rightarrow D_U$
(11)  end while
```

5 Experiments

In this section, we show the results of several experiments performed to evaluate the effectiveness of our proposed SWMEC for classification in real-world multi-label data streams TMC2007 [24] (see Table 2 in detail). All the experiments are conducted on a PC with Intel(R) Core(TM) i3-3220 3.30 GHz CPU and 4 GB RAM. We have implemented SWMEC in python2.7. Most importantly, in order to get more precise classification results, we firstly preprocess the data set. We remove the infrequent words that occur in less than 11% of the documents as [14] did.

Table 2. Summary of dataset TMC2007

Data set	Properties					
	N	d	q	Avg$\|Y\|$	IDens	Domain
TMC2007	28,596	204	22	2.158	0.098	Text

A. Classification quality comparison with ML-KNN
We compare the performance of our approach SWMEC against ML-KNN [31] as the base-line of our model. In order to apply this one of the state-of-the-art methods in offline context to the stream classification, we train a ML-KNN classifier on the latest chunk of data, and use it to classify the next chunk of data, which is similar to sliding window approaches. Since the data chunk size is the most important factors in the classification and training process, here we change the chunk sizes (sliding window sizes) to test the performance of basic ML-KNN. The Parameters are set as follows: SWMEC: L (the size of the ensemble classifier model) = 4, k (the number of nearest neighbors in ML-KNN) = 4, ϵ (the threshold about weight's adjustment) = 0.001. ML-KNN ($w = 100$): w (the size of the window) = 100. ML-KNN ($w = 200$): w (the size of the window) = 200. ML-KNN ($w = 400$): w (the size of the window) = 400.

Multi-label classification problems has many different metrics for evaluation. Such as Hamming-loss, F-measure, Log-Loss, Ranking-Loss [17]. Here we adopt two metrics to evaluate multi-label classification performance in a data stream. First, Micro F1: considers both micro average of Precision and Recall with equal importance, evaluates a classifier's label set prediction performance. The higher the value, the better the performance.

$$MicroF1\left(f_i, D_L\right) = \frac{2 \times \sum_{i=1}^{|D_L|} \left|f_i\left(x\right) \cap Y_i\right|}{\sum_{i=1}^{|D_L|} \left|f_i\left(x\right)\right| + \sum_{i=1}^{|D_L|} \left|Y_i\right|}$$

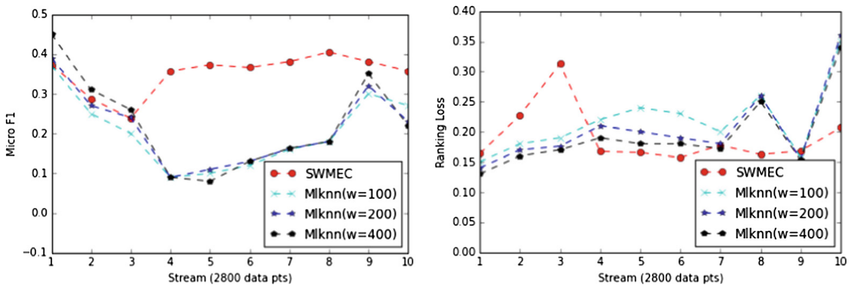
where $|D|$ is the number of instances in a multi-label data stream D, which contains (x_i, Y_i), where $Y_i \subseteq L(i = 1, \cdots, |D|)$, $f(x_i) \subseteq L$ denotes a multi-label classifier's predicted label set for x_i and $f(x_i, k)$ denotes the classifier's probability outputs for x_i on the k-th label (l_k).

Second, Ranking Loss [7]: compute the average number of label pairs that are incorrectly ordered given Y_i weighted by the size of the label set and the number of labels not in the label set. Evaluates the performance of classifier's probability outputs or real-value outputs $f(x_i, k)$. The best performance is achieved with a ranking loss of zero.

$$RankLoss(f, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{1}{|Y_i||\bar{Y_i}|} loss\left(f, x_i, Y_i\right)$$

$$loss(f, x_i, Y_i) = \sum_{k \in Y_i} \sum_{k' \in \bar{Y_i}} I\left(f\left(x_i, k\right) \leq f\left(x_i, k'\right)\right)$$

where the $\bar{Y_i}$ denotes the complementary set of Y_i in $\mathcal{L}$.



(a) Micro F1↑ (b) Ranking Loss↓

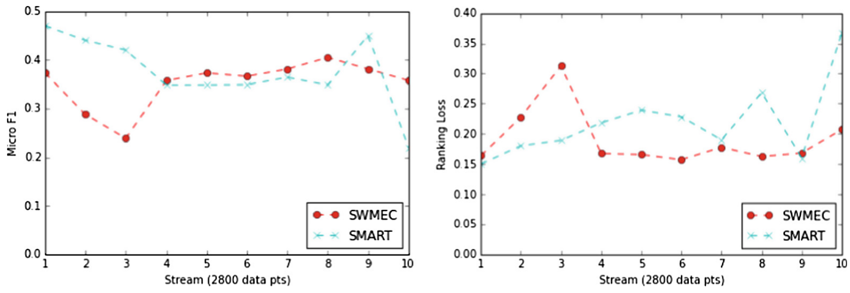
Fig. 1. SWMEC against ML-KNN on TMC2007 dataset.

Figure 1(a) shows the Micro F1 for the four algorithm throughout the stream in TMC2007 data-set. We report the average performance on every $|D|/10$ instances. For example, at X axis = 5, the Y values show the average Micro F1 of four classification models from the $|D| * 4/10$th instance of the stream to the $|D| * 5/10$th instance. At this point, the Micro F1 of SWMEC is 0.3729, the Y values in ML-KNN ($w = 100$), ML-KNN ($w = 200$) and ML-KNN ($w = 400$) are all below SWMEC.

We also calculate the Ranking loss of four classification models. Figure 1(b) show the Ranking loss of the four algorithms throughout the stream in TMC2007 data-set. For example, at X axis = 6, the ranking loss of SWMEC is 0.1571, the Y values in ML-KNN ($w = 100$), ML-KNN ($w = 200$) and ML-KNN ($w = 400$) are all higher than SWEMC.

B. Classification quality comparison with SMART [14]
We also compare the performance of our approach SWMEC against SMART [14], which also adopt the strategy of ensemble and gives us a great inspiration in this paper. As Fig. 2(a) and (b) shows, Our method SWMEC is slightly better than SMART in Micro F1 and Ranking Loss. Especially noteworthy is that SMART has a much bigger space overhead than SWMEC, because SMART

needs to maintain several tree structures while SWMEC only needs to store small amount of central points. In the future, we would like to combine both of these two methods' advantages to address the multi-label classification problem in data streams.



(a) Micro F1↑ (b) Ranking Loss↓

Fig. 2. SWMEC against SMART on TMC2007 dataset.

6 Conclusion

This paper presents an efficient algorithm for multi-label data stream classification based on ML-KNN. As the properties of data stream and multiple labels assigned to each instances. It becomes more challenging than the traditional static multi-label data classification problems and single-label data stream classification problems. To address these challenges, we propose an ensemble multilabel data stream classification approach, manly Streaming Weighting ML-KNN based Ensemble Classifier (SWMEC), to efficiently update the model with the multi-label data stream flows. Then our model can effectively and efficiently predict multiple labels for future data points. The experimental results on the real world validate that our multi-label data stream classification approach is very effective and efficient for multi-label stream classification.

Acknowledgment. This work is supported by Research Initiative Grant of Sun Yat-Sen University under Project 985 and Australian Research Council Discovery Projects funding DP150104871. The corresponding author is Hong Shen.

References

1. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: On demand classification of data streams. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 503–508. ACM (2004)
2. Aggarwal, C.C., Yu, P.S.: Locust: an online analytical processing framework for high dimensional classification of data streams. In: 2008 IEEE 24th International Conference on Data Engineering, pp. 426–435. IEEE (2008)

3. Boutell, M.R., Luo, J., Shen, X., Brown, C.M.: Learning multi-label scene classification. Patt. Recogn. **37**(9), 1757–1771 (2004)
4. Brinker, K., Fürnkranz, J., Hüllermeier, E.: A unified model for multilabel classification and ranking. In: Proceedings of the 2006 Conference on ECAI 2006: 17th European Conference on Artificial Intelligence, August 29–September 1, 2006, Riva del Garda, Italy, pp. 489–493. IOS Press (2006)
5. De Comité, F., Gilleron, R., Tommasi, M.: Learning multi-label alternating decision trees from texts and data. In: Perner, P., Rosenfeld, A. (eds.) MLDM 2003. LNCS, vol. 2734, pp. 35–49. Springer, Heidelberg (2003). doi:10.1007/3-540-45065-3_4
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. Elisseeff, A., Weston, J.: A kernel method for multi-labelled classification. In: Advances in Neural Information Processing Systems, pp. 681–687 (2001)
8. Fung, G.P.C., Yu, J.X., Wang, H., Cheung, D.W., Liu, H.: A balanced ensemble approach to weighting classifiers for text classification. In: Sixth International Conference on Data Mining (ICDM 2006), pp. 869–873. IEEE (2006)
9. Gama, J., Rocha, R., Medas, P.: Accurate decision trees for mining high-speed data streams. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 523–528. ACM (2003)
10. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 22–30. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24775-3_5
11. Gopal, S., Yang, Y.: Multilabel classification with meta-level features. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 315–322. ACM (2010)
12. Hüllermeier, E., Fürnkranz, J., Cheng, W., Brinker, K.: Label ranking by learning pairwise preferences. Artif. Intell. **172**(16), 1897–1916 (2008)
13. Kolter, J.Z., Maloof, M.A.: Using additive expert ensembles to cope with concept drift. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 449–456. ACM (2005)
14. Kong, X., Yu, P.S.: An ensemble-based approach to fast classification of multi-label data streams. In: 2011 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), pp. 95–104. IEEE (2011)
15. 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 (LNAI), vol. 5828, pp. 308–321. Springer, Heidelberg (2009). doi:10.1007/978-3-642-05224-8_24
16. Read, J.: Scalable multi-label classification (2010)
17. 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)
18. Read, J., Bifet, A., Holmes, G., Pfahringer, B.: Efficient multi-label classification for evolving data streams (2010)
19. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Mach. Learn. **85**(3), 333–359 (2011)
20. Sanden, C., Zhang, J.Z.: Enhancing multi-label music genre classification through ensemble techniques. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 705–714. ACM (2011)

21. Spyromitros-Xioufis, E.: Dealing with concept drift and class imbalance in multi-label stream classification. Ph.D. thesis, Department of Computer Science, Aristotle University of Thessaloniki (2011)
22. Street, W.N., Kim, Y.: A streaming ensemble algorithm (sea) for large-scale classification. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 377–382. ACM (2001)
23. Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. Department of Informatics, Aristotle University of Thessaloniki, Greece (2006)
24. Tsoumakas, G., Spyromitros-Xioufis, E., Vilcek, J., Vlahavas, I.: Mulan: a java library for multi-label learning. J. Mach. Learn. Res. **12**, 2411–2414 (2011)
25. Tsoumakas, G., Vlahavas, I.: Random *k*-labelsets: an ensemble method for multilabel classification. In: Kok, J.N., Koronacki, J., Mantaras, R.L., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 406–417. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74958-5_38
26. Wang, H., Wu, J., Zhu, X., Zhang, C.: Time-variant graph classification (2016). arXiv preprint: arXiv:1609.04350
27. Wang, H., Zhang, P., Zhu, X., Tsang, I., Chen, L., Zhang, C., Wu, X.: Incremental subgraph feature selection for graph classification. IEEE Trans. Knowl. Data Eng. **29**, 128–142 (2016)
28. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 226–235. ACM (2003)
29. Wang, P., Zhang, P., Guo, L.: Mining multi-label data streams using ensemble-based active learning. In: SDM, pp. 1131–1140. SIAM (2012)
30. Zhang, M.-L., Zhou, Z.-H.: Multilabel neural networks with applications to functional genomics and text categorization. IEEE Trans. Knowl. Data Eng. **18**(10), 1338–1351 (2006)
31. Zhang, M.-L., Zhou, Z.-H.: Ml-knn: a lazy learning approach to multi-label learning. Patt. Recogn. **40**(7), 2038–2048 (2007)
32. Zhang, P., Gao, B.J., Zhu, X., Guo, L.: Enabling fast lazy learning for data streams. In: 2011 IEEE 11th International Conference on Data Mining, pp. 932–941. IEEE (2011)
33. Zhu, X., Zhang, P., Lin, X., Shi, Y.: Active learning from data streams. In: Seventh IEEE International Conference on Data Mining (ICDM 2007), pp. 757–762. IEEE (2007)