Binary Transformation Method for Multi-Label Stream Classification

Ege Berkay Gulcan
Bilkent University
Ankara, Turkey
berkay.gulcan@bilkent.edu.tr

Isin Su Ecevit
Bilkent University
Ankara, Turkey
su.ecevit@ug.bilkent.edu.tr

Fazli Can
Bilkent University
Ankara, Turkey
canf@cs.bilkent.edu.tr

ABSTRACT

Data streams produce extensive data with high throughput from various domains and require copious amounts of computational resources and energy. Many data streams are generated as multi-labeled and classifying this data is computationally demanding. Some of the most well-known methods for Multi-Label Stream Classification are Problem Transformation schemes; however, previous work on this area does not satisfy the efficiency demands of multi-label data streams. In this study, we propose a novel Problem Transformation method for Multi-Label Stream Classification called Binary Transformation, which utilizes regression algorithms by transforming the labels into a continuous value. We compare our method against three of the leading problem transformation methods using eight datasets. Our results show that Binary Transformation achieves statistically similar effectiveness and provides a much higher level of efficiency.

CCS CONCEPTS

• **Information systems** → **Data stream mining**; • **Computing methodologies** → **Machine learning**.

KEYWORDS

Data stream; classification; multi-label; problem transformation

ACM Reference Format:
Ege Berkay Gulcan, Isin Su Ecevit, and Fazli Can. 2022. Binary Transformation Method for Multi-Label Stream Classification. In *Proceedings of the 31st ACM Int'l Conference on Information and Knowledge Management (CIKM '22), Oct. 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3511808.3557553

1 INTRODUCTION

Data streams carry information that is of extensive size and arrive at extremely high speeds, such as audio and video samples, emails, online articles, and social media updates [9, 14, 21]. Data stream classification, is the affiliation of data instances in data streams with their associated labels. Since data streams provide continuous and potentially infinite data, stream classification algorithms must be able to keep up with the speed, change[5] and storage requirements of the data [9].

A great majority of the data stream classifiers focus on the problem of single-label (multi-class) classification. As far as single-label classification is concerned, each data instance is associated with only one class label from the collection of individual labels [26]. Multi-Label Stream Classification (MLSC) differs from single-label classification in the sense that it handles the problem of one sample being associated with multiple labels at the same time [21], which makes it an important problem domain in information retrieval (IR). MLSC is reported to be used in various fields such as text categorization, image, video, crime and query classification, and diagnosis prediction [15, 25, 29, 32]. For instance, an image of a scenery can include clouds, the sea, and the sun at the same time. MLSC can also be used with different types of data such as tweets for sentiment and stance analysis, or news articles when classifying in terms of truthfulness and objectivity. [4, 11].

Common approaches for MLSC are Problem Transformation (PT), which is converting the problem into a single-label classification problem, Algorithm Adaptation, which is modifying the classification algorithm to handle multiple labels, or Ensemble, which is using a combination of MLSC techniques together. Problem Transformation (PT) techniques include Binary Relevance (BR), Classifier Chains (CC), and Label Powerset (LP). These are some of the most prominent methods for MLSC and are widely used [23]. BR and CC are based on transforming the data into individual labels, whereas LP deals with mapping combinations of labels into new single labels. All of these methods have the benefit of being conceptually simple since they utilize already existing classification algorithms that handle disjoint labels. However, as the size of the label set increases, BR and CC scale linearly, and LP scales exponentially in the solution domain. This not only illustrates that the current PT techniques are inefficient in terms of time and space, but also shows that they require generous amounts of energy and computational resources [22]. The properties make them unsuitable for MLSC in real-life problems.

In this study, we propose a novel PT method for multi-label classification that changes the label vectors into continuous values, which are then used for classification through regression. Our method provides a highly scalable and efficient procedure that is more suitable to process data streams with high throughput.

In this work, we (1) propose a novel and efficient, problem transformation-based multi-label classification method, (2) perform experimental and statistical evaluation of our method by comparing it against three of the most prevalent PT methods using eight datasets with varying IR-related classification domains and properties, (3) compare the efficiency of our method against baseline methodologies in terms of execution time per data item through experimental and statistical testing.

CIKM '22, October 17–21, 2022, Atlanta, GA, USA

ACM ISBN 978-1-4503-9236-5/22/10.
https://doi.org/10.1145/3511808.3557553

2 RELATED WORKS

2.1 Multi-label Stream Classification

In single-label approaches, each data instance d_t in a data stream $\mathcal{D} = d_0, d_1, ..., d_N$ is affiliated with only one class $\lambda \in \{0, 1\}$ from a set of all possible labels $\mathcal{L} = \{\lambda_1, \lambda_2, ..., \lambda_n\}$, where $n > 1$. In binary classification, $n = 2$, and in multi-class classification, $n > 2$ [27].

Let a data instance arriving at time t be $d_t = (x^t, y^t)$, where x^t are the features of the data instance in the input feature space $\mathcal{X} = \mathbb{R}$, $x^t =< x_1^t, x_2^t, ..., x_M^t > \in \mathcal{X}$, and y^t is the ground truth labels of the data instance $y^t =< y_1^t, y_2^t, ..., y_n^t >= \{0, 1\}^n$. Then, in a multi-class environment: $\sum_{i=1}^{n} y_i^t = 1$. In Multi-Label Stream Classification, every data instance d_t is associated with multiple labels at the same time, meaning that: $\sum_{i=1}^{n} y_i^t \geqslant 1$.

A multi-label classifier is concerned with the prediction of a set of relevant labels to associate with a new data instance [6].

2.2 Problem Transformation (PT) Methods

Problem Transformation [3, 24, 25, 35] methods utilize the presence of numerous established single-label learning methods by converting the data from multiple labels into single labels and therefore transform the multi-label problem into separate binary classification problems. The most common transformation methods include Binary Relevance (BR) [25], Classifier Chains (CC) [17], and Label Powerset (LP) [26].

Binary Relevance creates respective classifiers for each label independently in a multi-label setting. This approach makes BR fairly simple to use, and its complexity scales linearly with the number of labels. Furthermore, BR is not limited to a single learning method, since any method for dealing with individual labels can be utilized. One weakness of BR is that the inter-relationship between the labels is overlooked since it handles each label one by one [25, 33].

Classifier Chains has the advantages of BR, and preserves the label correlations, by constructing multiple binary classifiers following the sequence of the labels. Each classifier takes the predictions for the previous labels into account, where the labels can be permuted using a variety of techniques. However, the predictions of CC are highly dependent on the arrangement of labels, therefore accuracy is easily affected by different label sequences. Furthermore, the dependence on the previous classifiers prevents the process from being parallelized, thus decreasing the computational resource utilization [17, 25, 30, 35].

Label Powerset is a simple method that allows the usage of any multi-class learning technique for classification by transforming each combination of labels into a single-class. Unlike BR, LP takes the inter-relationship between labels into account. The drawback of LP is that the number of mapped classes grows exponentially with the number of label combinations, which causes the computation to be costly when dealing with a large number of them. The fact that each combination is converted into a different class can cause some classes to appear significantly less often than others, which may hinder accuracy. Finally, LP cannot predict label combinations it has not seen before [8, 25, 26, 30, 35].

3 OUR APPROACH: BINARY TRANSFORMATION METHOD

In a streaming environment, the high throughput of incoming data requires the classification algorithms to be efficient. However, many of the previously proposed methods fall short in terms of efficiency in time. To solve this problem, we propose the Binary Transformation method, an efficient PT technique for classification that utilizes regression algorithms.

Given a multi-label data stream $\mathcal{D}$, with each data instance being d_t, a classification algorithm predicts the entirety of the output vector $\hat{y}^t$, where $\hat{y}^t =< \hat{y}_1^t, \hat{y}_2^t, ..., \hat{y}_n^t >= \{0, 1\}^n$. However, y^t and $\hat{y}^t$ vectors can also be seen as binary encodings of continuous integers.

During the training process, the BT classifier transforms each ground truth label y^t into an integer. This transformation is done by:

$$C_t = \sum_{i=1}^{n} y_i^t * 2^{i-1} \tag{1}$$

Through continuous transformation of each of the training labels, we reconstruct our data stream into Equ. 2, which we use to train a base regression model R to be able to predict the transformed integer as a continuous value.

$$\mathcal{D}' = d_1', ..., d_N' \quad , \quad d_t' = (x^t, C_t) \tag{2}$$

Following the training of the regression model, given a data instance d_t and its features x^t, the BT classifier predicts by first receiving the regression output:

$$\hat{C}_t = \lfloor R(x^t) \rceil \tag{3}$$

Then, it solves Equ. 4 for each predicted label $\hat{y}_i^t$ where i is the *i-th* label of a prediction $\hat{y}^t$:

$$\hat{C}_t = \hat{y}_1^t \cdot 2^0 + \hat{y}_2^t \cdot 2^1 + ... + \hat{y}_n^t \cdot 2^{n-1} \tag{4}$$

Finally, it obtains the prediction vector:

$$\hat{y}^t =< \hat{y}_1^t, \hat{y}_2^t, ..., \hat{y}_n^t > \tag{5}$$

A well trained regression algorithm (optimized for minimal loss), can make close enough predictions, which results in accurate label vectors after transformation that allows the method to be effective. Furthermore, BT preserves the information about common labels e.g, $< 1, 0, 1 >$ (5) and $< 1, 1, 0 >$ (6) are seen as two independent classes by LP even though they share a common label. However, our approach utilizes this dependency since it directly transforms the label vector into an integer, which produces values in close proximity (5 and 6 in the previous example). This allows the regression model to associate the data features to this produced value range which increases per-label effectiveness.

The regression approach allows BT to be scalable with an increasing number of labels since only a single regression algorithm is employed, thus enabling fast execution. Furthermore, since we only predict one continuous value, BT scales well with the increasing number of label combinations unlike LP. The combinations of these properties makes BT scale in O(1) time in terms of execution time, with regards to number of classes since BT only employs one regressor as the number of class labels increase. Figure 1 exhibits a simple illustration of BT.

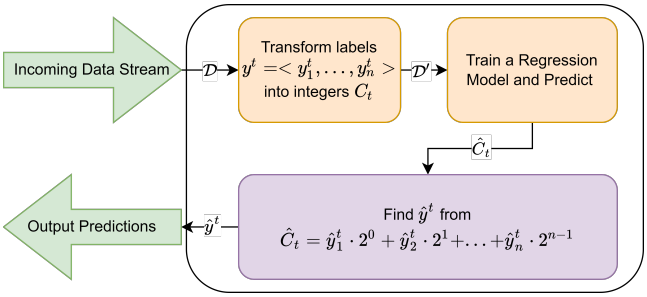

Figure 1: General workflow of the Binary Transformation method.

4 EXPERIMENTAL SETUP AND EVALUATION

We use eight real-world datasets with varying properties and from different IR-related problem domains to show the general effectiveness of BT, in the MEKA [18] format[1]. Their properties are displayed in Table 1. The chosen datasets are transformed into data streams in which the samples arrive in the order they originally occur.

Table 1: The table of multi-label datasets used in the experiments, where N is the number of samples, M is the number of features and n is the number of classes. LC($\mathcal{D}$) is the label cardinality (Average number of true labels for the samples in a data stream $\mathcal{D}$) and LD($\mathcal{D}$) is the label density ($\frac{LC(\mathcal{D})}{n}$).

Dataset	Domain	N	M	n	LC($\mathcal{D}$)	LD($\mathcal{D}$)
20NG [10]	Text	19,300	1,006	20	1.029	0.051
EukaryotePseAAC [31]	Biology	7,766	440	22	1.146	0.052
Imdb [16]	Text	120,900	1,001	28	2.000	0.071
Mediamill [20]	Video	43,910	120	101	4.376	0.043
Reuters-K500 [28]	Text	6,000	500	103	1.462	0.014
Scene [1]	Image	2,407	294	6	1.074	0.179
Slashdot [16]	Text	3,782	1,079	22	1.181	0.054
Yelp [19]	Text	10,810	671	5	1.638	0.328

For our evaluation, we use exact match accuracy, Hamming score, micro-averaged F1 score, and macro-averaged F1 score to measure the effectiveness of the methods [34]. The former two metrics allow us to gauge the complete and partial correctness of the predictions, and the latter two allow us to assess the sample-based and class-based effectiveness. Furthermore, we evaluate the efficiency of our algorithm by measuring the execution time per data item (in centiseconds).

The experiments are performed using the River framework[2] [12] and executed on a machine with Intel Core i5-10400F CPU, 16 GB of RAM and Ubuntu 20.04.1 LTS as the operating system. We compare our algorithm against BR, CC and LP using a Hoeffding Tree (HT) [7] as a base classifier through the interleaved-test-then-train evaluation (prequential evaluation) [2]. Likewise, we use an HT regressor as the base model for our algorithm. The HT classifiers employed are tested using default parameters, where they use adaptive Naive Bayes classifiers on the leaf nodes. The HT regressor for our algorithm similarly utilizes linear regression nodes on the leaves.

Furthermore, although many classifiers are adapted to multi-label learning, we chose to compare BT against other PT techniques to focus the scope of this work since it is also a transformation-based approach.

[1]The datasets can be accessed from: http://www.uco.es/kdis/mllresources/

[2]The source code of BT is avaliable at: https://github.com/egeberkaygulcan/binary-transformation-method

5 RESULTS AND DISCUSSIONS

In the following sections, we present and discuss our experiments on effectiveness and examine the efficiency of the tested methods. In all of the tables presented in this section, the best results are highlighted in bold.

5.1 Effectiveness Analysis

Table 2 displays the effectiveness results of the tested algorithms. The experimental results show that, on average, BT achieves the best results on F1 scores and remains close behind on the other two metrics. From the experiments, we can deduce that the main detriment of BT, seems to be large number of labels in terms of effectiveness. This is due to the rapid increase in the size of the solution set. However, from the macro-averaged F1 scores, we can see that BT performs well based on per-label effectiveness since macro-averaged F1 score is calculated using the individual F1 scores of the classes. This indicates that, although BT performs comparatively worse on streams with a large number of labels per-sample, it still shows adequate effectiveness on partial predictions label-wise.

To further analyze the effectiveness of BT, we perform "Friedman test with Nemenyi post-hoc analysis" [13] where we investigate the statistical significance of our experimental results. Figure 2 illustrates our two-tailed Nemenyi statistical significance tests with critical distance $CD = 1.6583$.

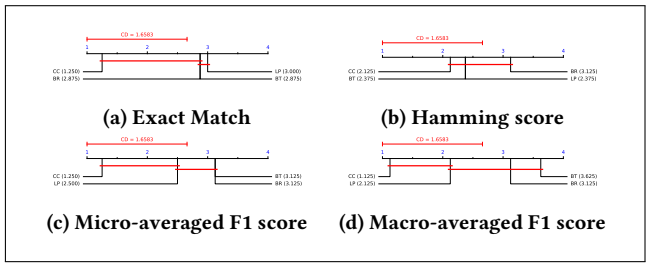

Figure 2: Nemenyi critical distance diagrams for all effectiveness metrics. The number given within parentheses after the method name indicates the rank position of the method.

The statistical analysis shows that overall, BT demonstrates statistically insignificant effectiveness compared to the baselines for all metrics, except CC on F1 scores, where it is better. This means that our method performs statistically similarly to the compared algorithms.

5.2 Efficiency Analysis

Table 3 presents the experiments we conducted where we measure the efficiency of the methods in terms of execution time per data

Table 2: Experimental results on all effectiveness metrics (higher is better). For some experiments, the estimated time for completion was infeasible which is denoted by TLC (Too long to complete).

Subset Accuracy	BT	BR	CC	LP	**Hamming Score**	BT	BR	CC	LP
20NG	0.187	0.248	0.040	**0.357**	20NG	0.917	**0.955**	0.912	0.928
EukaryotePseAAC	**0.776**	0.376	0.072	0.202	EukaryotePseAAC	**0.988**	0.953	0.911	0.925
Imdb	**0.054**	0.007	0.000	TLC	Imdb	0.889	0.925	**0.929**	TLC
Mediamill	0.004	**0.059**	0.054	0.054	Mediamill	0.346	**0.967**	0.958	0.958
Reuters-K500	0.018	0.062	0.000	**0.064**	Reuters-K500	0.820	0.977	**0.986**	**0.986**
Scene	**0.870**	0.301	0.153	0.562	Scene	**0.976**	0.841	0.713	0.861
Slashdot	**0.149**	0.021	0.000	0.141	Slashdot	0.912	**0.947**	0.946	0.915
Yelp	**0.593**	0.275	0.018	0.229	Yelp	**0.889**	0.753	0.502	0.700
Average Rank	2.125	2.125	3.750	**2.000**	Average Rank	2.375	3.125	**2.150**	2.375
Micro-averaged F1 Score					**Macro-averaged F1 Score**				
20NG	0.192	**0.428**	0.049	0.343	20NG	0.192	**0.418**	0.010	0.355
EukaryotePseAAC	**0.886**	0.513	0.083	0.287	EukaryotePseAAC	**0.795**	0.221	0.028	0.106
Imdb	**0.225**	0.041	0.000	TLC	Imdb	**0.129**	0.023	0.000	TLC
Mediamill	0.093	**0.507**	0.429	0.429	Mediamill	0.051	**0.079**	0.019	0.034
Reuters-K500	0.025	**0.059**	0.000	0.045	Reuters-K500	**0.022**	0.002	0.000	0.000
Scene	**0.932**	0.504	0.170	0.639	Scene	**0.938**	0.448	0.050	0.640
Slashdot	**0.179**	0.053	0.000	0.144	Slashdot	**0.086**	0.045	0.000	0.016
Yelp	**0.830**	0.607	0.067	0.514	Yelp	**0.858**	0.526	0.390	0.415
Average Rank	**1.875**	**1.875**	3.750	2.500	Average Rank	**1.375**	1.875	3.875	2.875

item over the datasets. Our results display that BT performs exceedingly better than the baselines. On average, our method exhibits 1,678.88% better performance with a minimum of 81.48% improvement on the Scene dataset which has low number of labels (6). We can see that BT achieves higher improvement margins on datasets with larger numbers of labels or samples since the BR and CC scale linearly with the number of labels. Furthermore, although LP scales similarly to BT in terms of label count, the number of classes LP predicts increases exponentially, since it assigns a new class for each label combination, whereas BT only predicts one continuous value regardless of the number of label combinations.

Table 3: Execution time per data item of the methods in centiseconds (lower is better). Average is calculated by calculating the mean result of the baseline methods. Improvement is the increase in efficiency BT has over the average.

Dataset	BR	BT	CC	LP	Average	Improvement (%)
20NG	12.212	**1.109**	12.813	10.865	11.964	978.97
EukaryotePseAAC	6.322	**3.052**	6.078	7.456	6.619	116.88
Imdb	18.538	**1.160**	18.872	148.883	62.098	5,251.13
Mediamill	9.214	**1.353**	15.903	46.935	24.017	1,675.42
Reuters-K500	17.350	**0.633**	32.867	26.267	25.494	3,925.44
Scene	1.496	**0.748**	1.454	1.122	1.357	81.48
Slashdot	13.485	**1.058**	13.088	9.704	12.092	1,043.33
Yelp	2.590	**0.731**	2.553	4.912	3.352	358.65
Average Rank	2.875	**1.000**	3.000	3.125		

Figure 3 illustrates the two-tailed Nemenyi significance test for our efficiency experiments. It can be seen that BT displays statistically significantly better efficiency in time, while the baselines show similar performance among themselves. Our combined results indicate that BT demonstrates similar effectiveness compared to our baselines in a much smaller time frame. Therefore, in streaming environments, BT is a more suitable PT method for MLSC.

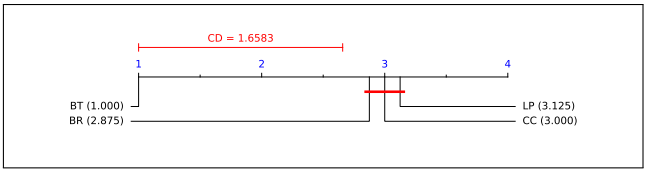

Figure 3: Nemenyi critical distance diagram for execution time per data item.

6 CONCLUSION

In this paper, we propose Binary Transformation method for multi-label stream classification that employs regression algorithms by transforming the labels into a continuous value. Our method allows fast execution while exploiting label dependencies. We perform our evaluation on three of the most prevalent PT methods using eight datasets with varying problem domains such as text and image classification with different properties. Our results show that BT achieves statistically similar effectiveness while providing much higher efficiency in terms of execution time. The results demonstrate that BT is a better PT technique for multi-label data streams than the previous work in the field.

For future work, we plan to study the effectiveness of BT on ensemble methods, such as GOOWE-ML [2], and develop procedures to reduce the range of the solution domain of our algorithm for higher scalability in terms of the number of labels. Furthermore, we intend to compare BT against algorithm adaptation methods for multi-label learning and perform further experiments with different types of regressors.

REFERENCES

[1] Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown. 2004. Learning multi-label scene classification. *Pattern Recognition* 37, 9 (2004), 1757–1771.

[2] Alican Büyükçakir, Hamed Bonab, and Fazli Can. 2018. A novel online stacked ensemble for multi-label stream classification. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1063–1072.

[3] Everton Alvares Cherman, Maria Carolina Monard, and Jean Metz. 2011. Multi-label problem transformation methods: a case study. *CLEI Electronic Journal* 14, 1 (2011), 4–4.

[4] Janaína Ignácio de Morais, Hugo Queiroz Abonizio, Gabriel Marques Tavares, André Azevedo da Fonseca, and Sylvio Barbon Jr. 2019. Deciding among Fake, Satirical, Objective and Legitimate news: A multi-label classification system. In *Proceedings of the XV Brazilian Symposium on Information Systems*. 1–8.

[5] Ege Berkay Gulcan and Fazli Can. 2022. Unsupervised concept drift detection for multi-label data streams. *Artificial Intelligence Review* (2022), 1–34.

[6] Francisco Herrera, Francisco Charte, Antonio J Rivera, and María J Del Jesus. 2016. Multilabel classification. In *Multilabel Classification*. Springer, 17–31.

[7] Geoff Hulten, Laurie Spencer, and Pedro Domingos. 2001. Mining time-changing data streams. In *Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 97–106.

[8] JD Costa Junior, ER Faria, JA Silva, and R Cerri. 2017. Label powerset for multi-label data streams classification with concept drift. In *Proceedings of the 5th Symposium on Knowledge Discovery, Mining and Learning*. Faculdade de Computação-Universidade Federal de Uberlândia, 97–104.

[9] Xiangnan Kong and S Yu Philip. 2011. An ensemble-based approach to fast classification of multi-label data streams. In *7th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*. IEEE, 95–104.

[10] K Lang. 2008. The 20 newsgroup dataset. http://people.csail.mit.edu/jrennie/20Newsgroups/

[11] Saif M Mohammad, Parinaz Sobhani, and Svetlana Kiritchenko. 2017. Stance and sentiment in tweets. *ACM Transactions on Internet Technology (TOIT)* 17, 3 (2017), 1–23.

[12] Jacob Montiel, Max Halford, Saulo Martiello Mastelini, Geoffrey Bolmier, Raphael Sourty, Robin Vaysse, Adil Zouitine, Heitor Murilo Gomes, Jesse Read, Talel Abdessalem, et al. 2021. River: machine learning for streaming data in Python. (2021).

[13] Peter Bjorn Nemenyi. 1963. *Distribution-free Multiple Comparisons*. Princeton University.

[14] Hai-Long Nguyen, Yew-Kwong Woon, and Wee-Keong Ng. 2015. A survey on data stream clustering and classification. *Knowledge and Information Systems* 45, 3 (2015), 535–569.

[15] Zhi Qiao, Zhen Zhang, Xian Wu, Shen Ge, and Wei Fan. 2020. MHM: Multi-modal Clinical Data based Hierarchical Multi-label Diagnosis Prediction. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1841–1844.

[16] Jesse Read. 2010. *Scalable multi-label classification*. Ph. D. Dissertation. University of Waikato.

[17] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. 2011. Classifier chains for multi-label classification. *Machine Learning* 85, 3 (2011), 333–359.

[18] Jesse Read, Peter Reutemann, Bernhard Pfahringer, and Geoff Holmes. 2016. Meka: a multi-label/multi-target extension to WEKA. *The Journal of Machine Learning Research* 17, 1 (2016), 667–671.

[19] Hitesh Sajnani, Vaibhav Saini, Kusum Kumar, Eugenia Gabrielova, Pramit Choudary, and Cristina Lopes. 2012. Classifying yelp reviews into relevant categories. *Mondego Group, Univ. California Press, Berkeley, CA USA, Tech. Rep* (2012).

[20] Cees GM Snoek, Marcel Worring, Jan C Van Gemert, Jan-Mark Geusebroek, and Arnold WM Smeulders. 2006. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the 14th ACM International Conference on Multimedia*. 421–430.

[21] Ricardo Sousa and João Gama. 2018. Multi-label classification from high-speed data streams with adaptive model rules and random rules. *Progress in Artificial Intelligence* 7, 3 (2018), 177–187.

[22] Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. *arXiv preprint arXiv:1906.02243* (2019).

[23] Piotr Szymanski and Tomasz Kajdanowicz. 2019. Scikit-multilearn: a scikit-based Python environment for performing multi-label classification. *The Journal of Machine Learning Research* 20, 1 (2019), 209–230.

[24] Farbound Tai and Hsuan-Tien Lin. 2012. Multilabel classification with principal label space transformation. *Neural Computation* 24, 9 (2012), 2508–2542.

[25] Vaishali S Tidake and Shirish S Sane. 2018. Multi-label Classification: A survey. *International Journal of Engineering and Technology* 7, 4.19 (2018), 1045–1054.

[26] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2009. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*. Springer, 667–685.

[27] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering* 23, 7 (2010), 1079–1089.

[28] Grigorios Tsoumakas and Ioannis Vlahavas. 2007. Random k-labelsets: An ensemble method for multilabel classification. In *European Conference on Machine Learning*. Springer, 406–417.

[29] Pengfei Wang, Yu Fan, Shuzi Niu, Ze Yang, Yongfeng Zhang, and Jiafeng Guo. 2019. Hierarchical matching network for crime classification. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 325–334.

[30] Donna Xu, Yaxin Shi, Ivor W Tsang, Yew-Soon Ong, Chen Gong, and Xiaobo Shen. 2019. Survey on multi-output learning. *IEEE Transactions on Neural Networks and Learning Systems* 31, 7 (2019), 2409–2429.

[31] Jianhua Xu, Jiali Liu, Jing Yin, and Chengyu Sun. 2016. A multi-label feature extraction algorithm via maximizing feature variance and feature-label dependence simultaneously. *Knowledge-Based Systems* 98 (2016), 172–184.

[32] Hang Yu and Lester Litchfield. 2020. Query Classification with Multi-objective Backoff Optimization. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1925–1928.

[33] Min-Ling Zhang, Yu-Kun Li, Xu-Ying Liu, and Xin Geng. 2018. Binary relevance for multi-label learning: an overview. *Frontiers of Computer Science* 12, 2 (2018), 191–202.

[34] Min-Ling Zhang and Zhi-Hua Zhou. 2013. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering* 26, 8 (2013), 1819–1837.

[35] Xiulin Zheng, Peipei Li, Zhe Chu, and Xuegang Hu. 2019. A survey on multi-label data stream classification. *IEEE Access* 8 (2019), 1249–1275.