Extreme Multi-label Classification

Man Jin (mj1637)*, Denglin Jiang (dj1369), Hong Gong (hg1153), Yuwei Wang (yw1854), Yi Xu (yx2090)

Center for Data Science, New York University

*Responsible for submission

1. Introduction

Extreme classification is a multi-label classification problem that annotates a data point with the most relevant subset of labels from an extremely large label set. It has wide applications in diverse areas such as dynamic search advertising, text classification, and recommender systems. The main technical challenges include improving the prediction accuracy and reducing the training time, prediction time and model size. In this project, we perform extreme multi-label classification on EURLex-4K dataset [26], a collection of documents about European Union Law with 3993 categories. We first applied traditional multi-label algorithms as baseline. We further implemented embedding based models Principal Label Space Transformation (PLST) [34] and Sparse Local Embeddings for Extreme Multi-label Classification (SLEEC) [3], and innovatively modify existed algorithms for desired property. We finally focus on one of the leading one-vs-all based extreme classifiers Partitioned Label Trees (Parabel) [30]. We use label ranking average precision (LRAP) as our evaluation metric to assess label ranking performance. We also record training times to evaluate model efficiency. The result shows that the Parabel [30] model achieves the highest LRAP score with the fastest training time among all algorithms we experimented.

2. Related Works

2.1. Traditional Multi-label Classifiers

Traditional algorithms for solving multi-label classification are mainly categorized into Problem Transformation and Algorithm Adaption [39].

Problem Transformation Methods This category of algorithms tackles multi-label learning problem by transforming it into well-established single-label problems (binary-class or multi-class) [39]. Representative transformations include **(1)** transforming to binary classification, such as Binary Relevance [4] and Classifier Chains [32], **(2)** transforming to label ranking, such as Calibrated Label Ranking [10], **(3)** transform to multi-class classification, such as Random k-labelsets [15] (See further details in 3.2.1).

Algorithm Adaption Methods Opposite to Problem Transformation methods, the key idea of Algorithm Adaption methods is to adapt well-established popular machine learning algorithms, usually by changing the cost functions, so that the adapted algorithms can fit the multi-label data [39]. Representative methods include Multilabel-KNN [40], Multilabel-Decision Tree [8], Rank-SVM [9], and Collective Multi-Label Classifier [11].

2.2. Extreme Multi-label Classifiers

However, when it comes to extreme multi-label classification, there emerge problems of data scalability and sparsity [25]. Data scalability problem comes from the rapidly increasing training examples, dimensionality and the number of class labels, whereas sparsity raises challenges to extract correlations among labels.

Over the years, researchers have developed more carefully designed algorithms to deal with such extremeness, most of which are based on trees [1, 17, 20, 29, 31], embeddings [3, 6, 7, 24, 33, 3, 36] and One-vs-All approaches [2, 25, 27, 37, 30]. For this project, we focus on embedding-based and One-vs-All extreme classifiers, and interprete how they successfully deal with scalability and sparsity.

Embedding Based Embedding based extreme multi-label classifiers are based on Label Space Dimension Reduction (LSDR), which assumes that the training label matrix is low-rank, and reduces the effective number of labels by projecting the high dimensional label vectors into a low dimensional linear subspace [3]. LSDR has been an effective and efficient paradigm in extreme multi-label classification [34, 14]. Popular LSDR algorithms include Singular Value Decomposition [12], Locality Preserving Projections [13], Multi-label informed Latent Semantic Indexing [38] etc. Due to different objective functions and optimizing routines, these compression algorithms differ in performances and scalability.

One-vs-All (OvA) OvA method learns weights corresponding to each individual label, and distinguishes the label from the rest. The OvA framework is empirically one of the best solutions for extreme multi-label classification due to its high accuracy. However, the framework greatly suffers from training and prediction inefficiency, since the great amount of labels yields extremely large model size and training complexity. Recent works have been trying to improve efficiency through various methods. Distributed Sparse Machines (DiSMEC) [2] adds a step on top of model training to predict sparsity for each weight parameter and keeps only non-trivial parameters. This solution successfully controls model size without sacrificing prediction accuracy, and its parallel training system reduces the training time. Partitioned Label Trees (Parebel) [30] adopts a tree-structure for model training steps instead of linear loops, which significantly improves the training and predicting efficiency. Scalable Linear Extreme Classifiers (SLICE) [16] solves extreme classification problems with low dimensional dense feature space. It reduces training sample size by negative sampling, approximates a conditional distribution from generative models, and learns linear classifiers for each label using MAP. All these works have been proved successful on the EURLex-4K dataset [26] and has been applied to constructing industry solutions.

3. Problem Definition and Algorithm

3.1. Task

In this project, we performed extreme multi-label classification on the EURLex-4K dataset. We formularized this task as follows[1]. Define $\mathcal{X} = \mathbb{R}^d$ as the d -dimensional input space, $\mathcal{Y} = \{y_1, y_2, \cdots, y_{|L|}\}$ as the output space with $|L|$ possible class labels, and hypothesis space as $\mathcal{H}$. Our task of multi-label learning is to learn a function $h \in \mathcal{H}$, $h : \mathcal{X} \to 2^{\mathcal{Y}}$ from the multi-label training set $\mathcal{D} = (X, Y) = \{(\boldsymbol{x}_i, y_i) \mid 1 \leq i \leq N\}$. For each multi-label example $(\boldsymbol{x}_i, y_i), \boldsymbol{x}_i \in \mathcal{X}$ is a d-dimensional feature vector, and $y_i \subseteq \mathcal{Y}$ is the set of labels associated with x_i. For a test point $x \in \mathcal{X}$, the multi-label classifier $h(\cdot)$ predicts $h(x) \subseteq \mathcal{Y}$ as the set of proper labels for x.

3.2. Algorithms

3.2.1 Problem Transformation Models

Binary Relevance (BR) Binary Relevance [4] decomposes the multi-label problem into $|L|$ independent binary classification problem, where every binary classifier corresponds to one label. However, this approach doesn't take correlations between labels into consideration.

Classifier Chains (CC) Classifier Chains [32] transforms multi-label learning problem into a chain of binary classification problems, where subsequent binary classifiers are built upon the predictions of preceding ones. CC has the advantage of exploiting label correlation, while losing parallelization ability due to its chain property.

[1]The definition is adopted from reference paper [39]

Label Powerset (LP) LP [15, 35] transforms a multi-label problem into a multi-class problem with one multi-class classifier trained on all unique label combinations found in the training data. This approach suffers from inefficiency and incompleteness, as the number of label sets could be exponentially large, and all random label sets comes from the training set and leads to poor generalization performance.

We combine these problem transformation techniques with Logistic Regression, Decision Tree, Random Forest, SVM and XGBoost [5]. Altogether, we have 15 models.

We also apply inherent multiclass classifiers like multi-layer perceptron (MLP), K Neighbors Classifier (KNN), and Random Forest Classifier (RF) to the dataset.

3.2.2 Principal Label Space Transformation (PLST)

Compared with traditional problem transformation methods in Section 3.2.1, PLST is faster and more effective as it captures key correlations among labels and reduces the number of needed classifiers significantly. During training, PLST projects standardized label vectors from high-dimensional label space into lower-dimensional space using Singular Value Decomposition (SVD) [12]. It then trains M linear ridge regressors $r(x)$ on each principal component. During prediction, it utilizes the projection matrix V_M to project predictions back to the original label space.

To capture complicated relations and further improve PLST performance, we **innovatively** replaced the linear Ridge Regression models by non-linear Decision Tree regressors to trade performance with computation efficiency.

Algorithm 1 Modified PLST algorithm

```
Input: Data $[(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)]$, Embedding Dimension $M$
Output: Predictions $[h(x_1), h(x_2), \cdots, h(x_N)]$
/* Training stage */
Let $Z = [z_1 \cdots z_N]^T$ with $z_i = y_i - \overline{y}$
Preform SVD on $Z$ to obtain $Z = A\Sigma B$ with $\sigma_1 \geq \sigma_2 \geq, \cdots, \geq \sigma_N$. Let $V_M$ contain the top rows of A.
for $i = 1$ to $N$ do
    Encode $(x_i, y_i)$ to $(x_i, t_i)$, where $t_i = V_M z_i$
Learn a decision tree regressor $r(x)$ from $\{(x_i, t_i)\}_{i=1}^N$
/* Prediction stage */
for $i = 1$ to $N$ do
    Predict the label-set of an instance $x$ by $h(x_i) = \text{round}({V_M}^T r(x_i) + \bar{y})$
```

3.2.3 Sparse Local Embeddings for Extreme Multi-label Classification (SLEEC)

Unlike PLST which gloabally projects onto a linear low-rank subspace, SLEEC learns an embedding which captures non-linear label relationships by preserving the pair-wise distance between only the closest rather than all label vectors [3]. SLEEC first partitions all the training points X into $Q_1, Q_2, \cdots, Q_C$ clusters by K-means [22]. Then for each cluster, SLEEC constructs a KNearestNeighbor graph Ω to capture pairwise distances. During training, it maps the label vectors y_i to M-dimensional vectors $t_i \in R^M$ and learns a set of regressors $V \in R^{M \times d}$ s.t. $t_i \approx Vx_i, \forall i$. During prediction, for an unseen point x, it first computes an embedding Vx and then performs kNN [21] over the set $[Vx_1, Vx_2, ..., Vx_N]$.

We **innotatively** modified the SLEEC [3] algorithm to improve its performance and efficiency by: **1.** using KD-tree [28] to search for nearest neighbors to fight against curse of dimensionality; **2.** solving matrix completion problem with Alternating Least Square [19] instead of the Singular Value Projection [18] to learn low-rank embedding, which trades performance for computation efficiency; **3.** using Elastic Net [41] instead of L1 norm as regularization term for our multi-linear regressor to gain more robustness while controlling model complexity.

Algorithm 2 Modified SLEEC: Training algorithm

```
Input: Train Data $(X, Y)$, X $\in R^{N \times d}, Y \in R^{N \times L}$,
Embedding Dimension $M$, Number of Neighbors $\bar{n}$, Number of Clusters $C$, Regularization Parameter for Elastic Net Regressor $\lambda$,
Output: $[(Q_1, r_1, Z_1), (Q_2, r_2, Z_2), \cdots, (Q_N, r_N, Z_N)]$
Partition $(X, Y)$ into $Q_1, Q_2, \cdots, Q_C$ using K-means
for each partition $Q_j$ do
    /* Construct Nearest Neighbor Graph */
    Form index set $\Omega$ containing $\bar{n}$ nearest neighbors using KD-tree for each label vector $y_i \in Q_j$
    /* Label Space Dimension Reduction */
    $[UV] \leftarrow$ ALS$(P_\Omega(Y_j Y_j^T), M)$
    $Z_j \leftarrow UV$
    Elastic-net Regressor $r_j \leftarrow$ TRAIN$(X_j, Z_j, \lambda)$
    $Z_j = r_j(X_j)$
```

Algorithm 3 SLEEC: Testing algorithm

```
Input: Test point $x$, Number of Neighbors $\bar{n}$, Number of Desired labels $p$,
Output: Predictions $h(x)$
$Q_c \leftarrow$ partition closest to $x$
$z \leftarrow r_c(x)$ where $r_c$ is the regressor
$N_z \leftarrow \bar{n}$ nearest neighbors of $z \in Z^c$
$P_x \leftarrow$ empirical label distribution for points $\in N_z$
$h(x) \leftarrow$ TOP$_p(P_x)$
```

3.2.4 Partitioned Label Trees (Parabel)

While leading One-vs-All extreme classifiers achieve high prediction accuracies, their training and prediction costs are linear in the number of labels times instances. Parabel [30] is a logarithmic-time One-vs-All algorithm that efficiently learns a balanced label hierarchy using tree structure, where labels in the same leaf are most similar, and negative examples used for training One-vs-All label classifiers are drawn from those examples having labels in the same leaf. It is still one of the fastest algorithms among all leading extreme classifiers.

The algorithm of Parabel can be summarized as follows. It learns a small ensemble of 3 label trees, where each label tree is grown by recursively partitioning the labels into two balanced groups using constrained spherical k=2-means. Nodes become leaves when a minimum leaf size is reached. The leaf nodes contain linear One-vs-All classifiers, one for each label in the leaf, trained on only those examples having at least one leaf node label. At prediction time, a test point will traverse the label tree and reach multiple leaves, as each node learns two linear classifiers indicating whether the test point should be passed down to the left, right, or both children. The One-vs-All classifiers in these leaves are evaluated to determine the probability that the corresponding labels are relevant to the test point, and predictions are made by averaging these probabilities across trees.

4. Experiments

4.1. Data

The EUR-LEX website contains a collection of documents about European Union Law with categorization provided by the EUROVOC descriptors, which is a topic hierarchy with 3993 categories regarding different aspects of European laws. The data [26] we use in this project has already been preprocessed. The input features are TF-IDF representation of the documents with the first 5000 most frequent features after tokenization, stopwords removal, and stemming. The outputs are a set of labels encoded with value between 0 and 3992. There are 15511 examples in training set, and each example contains 5.32 labels on average.

4.2. Methodology

Preprocessed training, validation, and test set are given for this project. For all algorithms mentioned in Section 3.2, we perform grid search on multiple parameters on the training set, and choose the best model based on the prediction accuracy on validation set. For model selection and evaluation, we use Label Ranking Average Precision (LRAP) as the evaluation criteria, which measures the percentage of the true labels among the higher-ranked labels for each of the given samples. We also adopt a popular ranking evaluation metric precision@k, which describes the proportion of recommended items in the top-k set that are relevant. Precision@k focuses more on the accuracy of the top k labels instead of averaged performance. As for the final model submitted on Gradescope, we choose the optimal model with best hyperparameter settings, train the model on both training and validation set together, and make predictions on the test set.

4.3. Results

We list models that give best LRAP scores from section 3.2.1 in Table 1. We compare original PLST results with our modified version in Table 2, and display comparison results between original SLEEC and modified version in Table 3.

Model	LRAP	Training time (s)
Parabel	**0.6337**	**285.55**
MLP	0.6041	4813.29
CC + RF	0.5647	17285.29
BR + RF	0.5593	19268.47
PLST	0.5028	20633.94
KNN	0.3936	466.44 [2]
RF	0.2887	1244.80
LP+RF	-	Out of RAM

Table 1. Best validation model performance

Embedding Dimension	Original LRAP	Modified LRAP	Original Runtime (s)	Modified Runtime (s)
10	0.1512	**0.1586**	655.1667	**472.3658**
50	0.2640	**0.2748**	**467.6908**	2304.2881
100	0.3208	**0.3391**	**874.6648**	5381.154
500	**0.4564**	-	**3995.0264**	Out of time
2000	**0.5028**	-	**20633.9433**	Out of time

Table 2. PLST performance on full data

Model	Orginial SLEEC	Modified SLEEC
LRAP	**0.3529**	0.2584
Precision@1	**0.5050**	0.4200
Precision@3	**0.4227**	0.3467
Precision@5	**0.3576**	0.2752
Training time(s)	3267.44	**2145.55**

Table 3. SLEEC performance on 10% data

4.4. Discussion

In general, our experiment outcomes are in accordance with our expectations for each category of algorithms.

Parabel Parabel has the highest LRAP and least training time, which proves the effectiveness of OvA methodology class as well as the effectiveness of tree structure training. We adopt Parebel as the best solution for our problem for its outstanding performance in both accuracy and efficiency.

MLP After extensive hyperparameter tuning on hidden dimension, we used a MLP with a single 4300-dimensional hidden layer. It yields a decent LRAP score due to its complicated model structure compared to traditional methods, but it does take longer training time due to the wide network. This result is consistent with Leshno and Schocken's theorem that a neural network with one huge hidden layer can uniformly approximate any continuous function on a compact set with a non-polynomial activation function [23].

[2]KNN is a lazy leaner, using training set in test phase rather than model generation.

Problem Transformation Methods Among the 15 combinations of problem transformation techniques and conventional classifiers, transformed Random Forest models stand out. "CC + RF" and "BR + RF" gives decent LRAP scores whereas "LP + RF" runs out of RAM. Compared with BR, CC gives higher score, which is consistent with the summary [39] that chain property would capture label dependence. However, these algorithms takes extremely long time to run, and has problem of scalability.

PLST From 2, higher label space embedding dimension tend to give better overall model performance. To further improve PLST performance, we innovatively replace the linear Ridge Regression models by non-linear Decision Tree regressors. It is able to improve Label Ranking Average Precision score by 0.6% to 1% under different parameter settings of our choice. However, this non-linearity trades performance with computation efficiency, increasing running time by approximately 5 times.

SLEEC Due to lack of necessary RAM and computing resources, we haven't been able to run our python implementation of modified SLEEC on full data. Instead, performance results on 10% data are compared with original SLEEC performance. Our modifications yield higher computation efficiency by shortening running time by $\frac{1}{3}$, but result in worse overall ranking accuracy. This result gives us an intuition that algorithm design is more of an elaborate art than science.

Others Traditional adapted multi-label classifiers like Multi-label Random Forest and Multi-label-KNN yield very low LRAP due to their incapability of handling sparse data in extreme multi-label setting.

5. Conclusions

In this paper, we have reviewed different classes of methods to solve extreme multi-label classification problems. We implemented the representatives of each algorithm groups and conducted experiments on EURLex-4K for comparison of performance. The results stand perfectly with our expectations from methodology analysis. We also proposed minor innovations for two embedding methods, PLST and SLEEC, and proved the improvement of performance. Finally, we concluded that the leading extreme classification model Parabel performs the best on this dataset, as it stands out from other models for both accuracy and efficiency.

Reproductivity

Our codes and implementation could be found at: https://github.com/TRokieG/Extreme_Multilabel_Classification

References

[1] Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web*, pages 13–24, 2013. 1

[2] Rohit Babbar and Bernhard Schölkopf. Dismec: Distributed sparse machines for extreme multi-label classification. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 721–729, 2017. 1, 2

[3] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. Sparse local embeddings for extreme multi-label classification. In *Advances in neural information processing systems*, pages 730–738, 2015. 1, 3

[4] M.R. Boutell, J. Luo, X. Shen, and C.M. Brown. Learning multi-label scene classification. In *Pattern Recognition vol. 37, no. 9*, page 1757–1771, 2004. 1, 2

[5] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016. 2

[6] Yao-Nan Chen and Hsuan-Tien Lin. Feature-aware label space dimension reduction for multi-label classification. In *Advances in Neural Information Processing Systems*, pages 1529–1537, 2012. 1

[7] Moustapha M Cisse, Nicolas Usunier, Thierry Artieres, and Patrick Gallinari. Robust bloom filters for large multilabel classification tasks. In *Advances in Neural Information Processing Systems*, pages 1851–1859, 2013. 1

[8] Amanda Clare and Ross D King. Knowledge discovery in multi-label phenotype data. In *european conference on principles of data mining and knowledge discovery*, pages 42–53. Springer, 2001. 1

[9] André Elisseeff and Jason Weston. A kernel method for multi-labelled classification. In *Advances in neural information processing systems*, pages 681–687, 2002. 1

[10] J. Furnkranz, E. Hullermeier, E. Lozamencla, and K. Brinker. Multilabel classification via calibrated label ranking. In *Machine Learning, vol. 73, no. 2*, page 133–153, 2008. 1

[11] Nadia Ghamrawi and Andrew McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, page 195–200, New York, NY, USA, 2005. Association for Computing Machinery. 1

[12] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. In *Linear Algebra*, pages 134–151. Springer, 1971. 1, 2

[13] Xiaofei He and Partha Niyogi. Locality preserving projections. In *Advances in neural information processing systems*, pages 153–160, 2004. 1

[14] Daniel Hsu, Sham M. Kakade, John Langford, and Tong Zhang. Multi-label prediction via compressed sensing. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, NIPS'09, page 772–780, Red Hook, NY, USA, 2009. Curran Associates Inc. 1

[15] J. Koronacki R. L. de Mantaras S. Matwin D. Mladenic in Lecture Notes in Artificial Intelligence 4701, J. N. Kok and A. Skowron. Random k-labelsets: an ensemble method for multilabel classification. In *Machine Learning, vol. 73, no. 2*, pages 406–417. Berlin: Springer, 2007. 1, 2

[16] Chunduri Jain, Balasubramanian and Varma. Slice: Scalable linear extreme classifiers trained on 100 million labels for related searches. In *WSDM '19: Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 528–536, 2019. 2

[17] Himanshu Jain, Yashoteja Prabhu, and Manik Varma. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 935–944, 2016. 1

[18] Prateek Jain, Raghu Meka, and Inderjit S Dhillon. Guaranteed rank minimization via singular value projection. In *Advances in Neural Information Processing Systems*, pages 937–945, 2010. 3

[19] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674, 2013. 3

[20] Kalina Jasinska, Krzysztof Dembczynski, Róbert Busa-Fekete, Karlson Pfannschmidt, Timo Klerx, and Eyke Hullermeier. Extreme f-measure maximization using sparse probability estimates. In *International Conference on Machine Learning*, pages 1435–1444, 2016. 1

[21] James M Keller, Michael R Gray, and James A Givens. A fuzzy k-nearest neighbor algorithm. *IEEE transactions on systems, man, and cybernetics*, (4):580–585, 1985. 3

[22] K Krishna and M Narasimha Murty. Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(3):433–439, 1999. 3

[23] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867, 1993. 4

[24] Zijia Lin, Guiguang Ding, Mingqing Hu, and Jianmin Wang. Multi-label classification via feature-aware implicit label space encoding. In *International conference on machine learning*, pages 325–333, 2014. 1

[25] Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124, 2017. 1

[26] Eneldo Mencía and Johannes Fürnkranz. Efficient pairwise multilabel classification for large-scale problems in the legal domain. volume 5212, pages 50–65, 09 2008. 1, 2, 3

[27] Alexandru Niculescu-Mizil and Ehsan Abbasnejad. Label filters for large scale multilabel classification. In *Artificial Intelligence and Statistics*, pages 1448–1457, 2017. 1

[28] Rina Panigrahy. An improved algorithm finding nearest neighbor using kd-trees. In *Latin American Symposium on Theoretical Informatics*, pages 387–398. Springer, 2008. 3

[29] Yashoteja Prabhu, Anil Kag, Shilpa Gopinath, Kunal Dahiya, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. Extreme multi-label learning with label features for warm-start tagging, ranking & recommendation. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 441–449, 2018. 1

[30] Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. pages 993–1002, 04 2018. 1, 2, 3

[31] Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *Proceedings of the 2018 World Wide Web Conference*, pages 993–1002, 2018. 1

[32] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. In *in Lecture Notes in Artificial Intelligence 5782 W. Buntine, M. Grobelnik, and J. Shawe-Taylor*, pages 254–269. Berlin: Springer, 2009. 1, 2

[33] Yukihiro Tagami. Annexml: Approximate nearest neighbor search for extreme multi-label classification. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 455–464, 2017. 1

[34] Farbound Tai and Hsuan-Tien Lin. Multilabel classification with principal label space transformation. *Neural computation*, 24(9):2508—2542, September 2012. 1

[35] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):1079–1089, 2010. 2

[36] Chang Xu, Dacheng Tao, and Chao Xu. Robust extreme multi-label learning. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1275–1284, 2016. 1

[37] Ian En-Hsu Yen, Xiangru Huang, Pradeep Ravikumar, Kai Zhong, and Inderjit Dhillon. Pd-sparse: A primal and dual sparse approach to extreme multiclass and multilabel classification. In *International Conference on Machine Learning*, pages 3069–3077, 2016. 1

[38] Kai Yu, Shipeng Yu, and Volker Tresp. Multi-label informed latent semantic indexing. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 258–265, 2005. 1

[39] M. Zhang and Z. Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2014. 1, 2, 4

[40] Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048, 2007. 1

[41] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005. 3