HSCNN: A Hybrid-Siamese Convolutional Neural Network for Extremely Imbalanced Multi-label Text Classification

Wenshuo Yang[1,2], **Jiyi Li**[2]*, **Fumiyo Fukumoto**[2] **and Yanming Ye**[1]
Hangzhou Dianzi University, Hangzhou, China[1]
University of Yamanashi, Kofu, Japan[2]
yangwen7873@gmail.com, {jyli,fukumoto}@yamanashi.ac.jp, yeym@hdu.edu.cn

Abstract

The data imbalance problem is a crucial issue for the multi-label text classification. Some existing works tackle it by proposing imbalanced loss objectives instead of the vanilla cross-entropy loss, but their performances remain limited in the cases of extremely imbalanced data. We propose a hybrid solution which adapts general networks for the head categories, and few-shot techniques for the tail categories. We propose a Hybrid-Siamese Convolutional Neural Network (HSCNN) with additional technical attributes, i.e., a multi-task architecture based on Single and Siamese networks; a category-specific similarity in the Siamese structure; a specific sampling method for training HSCNN. The results using two benchmark datasets and three loss objectives show that our method can improve the performance of Single networks with diverse loss objectives on the tail or entire categories.

1 Introduction

The data imbalance problem is a crucial issue for the multi-label text classification. In many corpora for the classification tasks, the number of instances of a category follows the long tail distribution, where many *tail categories* has only a small number of instances. To handle this problem, some works sample hard examples for training (Shrivastava et al., 2016); some works address the problem by proposing imbalance loss objectives, e.g., weighted cross-entropy loss and Focal loss (Lin et al., 2017), in place of the vanilla cross-entropy loss (Kim, 2014). Although the imbalanced loss objectives are better than the vanilla one, their performances remain limited in the cases of extremely imbalanced data because they are not designed for it, i.e., *tail (head) categories* have extremely small (large) numbers of instances.

On the one hand, the recent few-shot learning techniques (e.g., optimization-based methods (Finn et al., 2017; Munkhdalai and Yu, 2017; Mishra et al., 2018), metric-based methods (Koch et al., 2015; Vinyals et al., 2016; Snell et al., 2017)) have become popular for various NLP tasks (Yu et al., 2018; Han et al., 2018). They have already shown the capability for few-shot classifications. They thus may also perform well for tail category classification (some of the tail categories are few-shot, all have relatively small numbers of instances). On the other hand, for the head categories with many instances, general approaches such as the single CNN model (Kim, 2014; Liu et al., 2017) may be more effective in terms of performance and more efficient in terms of complexity. Therefore, our basic idea for tackling the problem of extremely imbalanced multi-label text classification is a hybrid solution that adapts a general approach (i.e., a Single network) for head categories and a few-shot approach for tail categories, so that we can take the advantages of both of them. For the few-shot approach, we select the Siamese network (Koch et al., 2015) because it is easier to integrate with different Single networks. A naïve solution is training them separately and utilizing their results on head categories and tail categories respectively as the combined classification results.

To make the hybrid solution effective, rather than a naïve combination on the results of two types of networks, we propose a Hybrid-Siamese Convolutional Neural Network (HSCNN) with additional technical properties. First, it is based on a multi-task architecture to deal with forgetting and overfitting problems when training the Siamese network. Second, the single similarity output of the vanilla Siamese structure is limited for estimating the similarities for a large number of categories and multiple categories; we thus propose a category-specific similarity in the Siamese structure. Third, we propose a specific sampling method to train the HSCNN.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 6716–6722,
November 16–20, 2020.

The results using two benchmark datasets and three loss objectives (including one vanilla cross-entropy loss and two imbalanced losses) show that the proposed method can improve the performance of a Single network with diverse loss objectives on the tail categories and the entire categories. The main contributions of this paper can be summarized as follows. (1). We propose a hybrid method based on general and few-shot techniques to mitigate the extremely imbalanced multi-label text classification problem. (2). We propose a novel HSCNN model based on a multi-task architecture, a category-specific similarity, and a specific sampling method. (3). Our approach can be integrated with the imbalanced loss objectives to improve the performance; the Hybrid-Siamese architecture can extend to incorporate with other types of Single networks rather than the Single CNN network.

Loss	Objectives
BCE	$-\sum_c \sum_{v\in\{0,1\}} y_{cv} \log p_{cv}$
WCE	$-\sum_c \alpha_c \sum_{v\in\{0,1\}} y_{cv} \log p_{cv}$
Focal	$-\sum_c \sum_{v\in\{0,1\}} y_{cv}(1-p_{cv})^\gamma \log p_{cv}$

Table 1: Loss Objectives

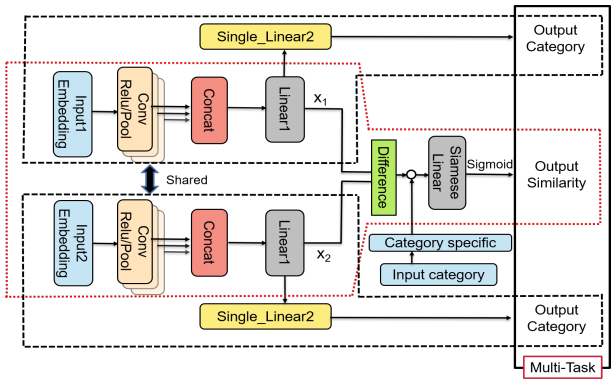


Figure 1: HSCNN Model

2 Our Approach

We denote the data as $\mathcal{D} = \{d_i\}_i$ and an instance as d_i, the category set as $\mathcal{C}$ and a category as c. The number of training instances is $\mathcal{N}$. The number of training instances of a category c is $\mathcal{N}_c$.

2.1 Single and Siamese Architectures

The Single architecture we use for multi-label text classification is similar to the CNN based models in existing works (Kim, 2014; Liu et al., 2017; Shimura et al., 2018). It includes an embedding layer, a convolutional layer and a pooling layer, and two fully connected layers. The black dashed line in Figure 1 marks a Single architecture. Note that this Single CNN network can also be replaced by other types of Single networks such as RNN, HAN (Yang et al., 2016), and so on. We utilize the CNN based one because it is one of the typical models.

We have several alternatives on the loss objectives computed by the predicted categories and true categories. Table 1 lists them. For each instance d_i, $y_{c1} = 1$ if d_i has the category c, $y_{c0} = 1 - y_{c1}$; p_{c1} is the predicted probability that d_i has category c, $p_{c0} = 1 - p_{c1}$. We use both the vanilla Binary Cross Entropy (BCE) and the imbalanced loss objectives including Weighted binary Cross Entropy (WCE) and Focal loss (Lin et al., 2017). We empirically set $\alpha_c = \log\left((\mathcal{N} - \mathcal{N}_c)/\mathcal{N}_c\right)$ for WCE loss and $\gamma = 1$ for Focal loss following the existing works (Li et al., 2020b). We do not use the Dice loss (Li et al., 2020b) because we empirically observe that it does not perform well for the multi-label text classification with a large number of categories, although it is also an imbalanced loss objective.

Siamese network (Koch et al., 2015) is a typical technique of few-shot learning. It contains two duplicated Single networks, and the inputs are two instances. The output is computed by comparing the representations extracted after the first fully connected layer of the Single network (Linear1 in Figure 1) for the two instances. Assuming that the representations of two input instances d_i and d_j are $\mathbf{x}_i$ and $\mathbf{x}_j$, there are two options on the comparison component. One option is leveraging a contrastive loss on the distance of $\mathbf{x}_i$ and $\mathbf{x}_j$. Another option (Koch et al., 2015) is utilizing a fully connected network on the difference of $\mathbf{x}_i$ and $\mathbf{x}_j$ to estimate their similarity and using a cross-entropy loss on the similarity. Because we need to estimate the similarities among a large number of categories, we select the later one to measure the rich information of the similarities. The dashed red line in Figure 1 marks a vanilla Siamese structure.

2.2 Hybrid-Siamese CNN

A naïve hybrid solution is training Single and Siamese networks separately and adapting them for head and tail categories respectively. To make the hybrid solution effective, we propose a Hybrid-Siamese Convolutional Neural Network (HSCNN) model (Figure 1) with three technical attributes.

Multi-task architecture: On the one hand, in the naïve solution, we can first train a Single network, then use it to initialize a Siamese network,

and after that train the Siamese network alone. The Siamese network may forget the knowledge learned by the Single network. On the other hand, When the number of training instances is large, the number of instance pairs is squared and huge. It is infeasible to train the Siamese network with all instances pairs, and we can only sample a subset. The number of training pairs is relatively small, which results in the overfitting of the Siamese network.

To prevent the above problems, we propose a multi-task architecture based on the Single and Siamese networks. As shown in Figure 1 with black solid line, the loss function is as follows, $\mathcal{L} = \lambda_s\mathcal{L}_s + \lambda_{m_1}\mathcal{L}_{m_1} + \lambda_{m_2}\mathcal{L}_{m_2}$. $\mathcal{L}_s$ is the loss of the Siamese network, and $\mathcal{L}_m$ is the loss of a Single network. For similar pairs of input instances, $\mathcal{L}_{m_1}$ and $\mathcal{L}_{m_2}$ are the same. For dissimilar pairs, $\mathcal{L}_{m_1}$ and $\mathcal{L}_{m_2}$ are the losses for each input instance, respectively. The comparison part of the HSCNN and the Single CNN part are trained together in the multi-task architecture. The Single network part can be regarded as a constraint to avoid the forgetting and overfitting of the Siamese network part. Without loss of generality, we set λ_s=λ_{m_1}=λ_{m_2}=1.

Category-specific similarity: The Siamese structure has only a single similarity output, which is limited for estimating the similarity between a large number of categories and multiple categories. For example, if an instance has multiple categories and each category has a representation vector, it is difficult to learn a representation of this instance near the representations of all these categories through a single similarity output. Therefore, we propose a category-specific similarity in the Siamese structure to capture the rich information in the similarities.

As shown in Figure 1, HSCNN has an asymmetric structure. In addition to the inputs of two instances d_i and d_j, there is another input category c which means that d_j has category c. Given the input triplet (d_i,d_j,c), the similarity output can be explained as "whether d_i is similar with d_j on category c". Denoting the one-hot encoding of category c as $\mathbf{q}_c$, a category-specific difference is computed by $\mathbf{h} = |\mathbf{x}_i - \mathbf{x}_j| \circ \mathbf{h}_c$, where $\mathbf{h}_c = \sigma((\mathbf{w}\mathbf{q_c} + \mathbf{b})/\sqrt{|\mathcal{C}|})$. σ is the ReLU activation function and $\circ$ is the elementwise multiplication. A linear layer with the sigmoid function then computes the similarity. This category-specific similarity can also be explained as a Machine Reading Comprehension (MRC) framework (Li et al., 2020a, 2019), which can improve non-MRC tasks' performance by learning additional information from the query. It is also related to the joint embedding on the instance and category (Wang et al., 2018), while ours focuses on the category-specific similarity of instances.

Sampling method: A common sampling method of the training data for a Siamese network is randomly selecting similar ((d_i,c),(d_j,c)) and dissimilar pairs ((d_i,c_i),(d_j,c_j)) with the ratio of 1:1. In this work, we generate one pair by randomly selecting the categories and selecting the instances in the categories. We set a heuristic rule to ensure that each category can be selected as least $\mathcal{T}$(e.g., ten) times. To follow the asymmetric structure of HSCNN, we propose a specific sampling method for training HSCNN. For each similar pair ((d_i,c),(d_j,c)), we generate one triplet (d_i,d_j,c) for training; for each dissimilar pair ((d_i,c_i),(d_j,c_j)), we generate two triplets (d_i,d_j,c_j) and (d_j,d_i,c_i), the ratio of similar and dissimilar pairs is thus 1:2.

We train and utilize HSCNN for classification as follows. **Training:** We first train the Single CNN separately by utilizing the raw training data in the dataset until convergence. After that, we use it to initialize HSCNN and train HSCNN by utilizing the sampled triplets as the training data. **Classification:** When using HSCNN to predict the categories of test instances, we use the Siamese part's output. For a test instance d_i and a category c, we randomly select five instances from the category c of the raw training data. We compute the mean $\bar{\mathbf{x}}^c$ of the representations of these five instances obtained by the representation extraction component. We compare the representation $\mathbf{x}_i$ of d_i with $\bar{\mathbf{x}}^c$ by the comparison component to calculate the similarity. If the similarity is higher than 0.5, we assign the category of c to d_i. Note that the option of first computing the mean representation of the five instances is not mandatory in our proposed approach. Another option for classification is first comparing d_i with each of the five instances and then using majority voting to aggregate the results. We choose the mean representation option because we obtain a little bit better results than another option. **Merge:** We finally merge the classification results of CNN and HSCNN as the results of our hybrid solution on the entire categories. We set a threshold $\mathcal{N}_\phi$ on the number of instances in a category. For tail categories ($\mathcal{N}_c < \mathcal{N}_\phi$, a subset with the instances that contain at least one tail cate-

Dataset	Train	Test	$\|\mathcal{C}\|$	$\bar{\mathcal{N}_c}$	$\mathcal{N}_c^{max}$	$\mathcal{N}_c^{min}$
RCV1	23,149	781,265	103	225	10,787	1
Delicious	12,920	3,185	983	14	3,867	12

Table 2: Data statistics. $|\mathcal{C}|$ is total number of categories, $\bar{\mathcal{N}_c}$ is mean of $\mathcal{N}_c$. $\mathcal{N}_c^{max}$ and $\mathcal{N}_c^{min}$ are maximum and minimum of $\mathcal{N}_c$. (URLs are in Appendix.)

gory), we use HSCNN results; For head categories ($\mathcal{N}_c \geq \mathcal{N}_\phi$), we use CNN results.

3 Experiments

3.1 Experimental Settings

The two benchmark datasets we use are the multi-label text corpus RCV1 (Lewis et al., 2004) and Delicious (Tsoumakas et al., 2008). Table 2 lists the dataset statistics. There are many categories, and the categories are extremely imbalanced on the instance numbers. We randomly split the raw training data into 75% for training and 25% for validation. We use the fastText (Joulin et al., 2017) to generate the initial word embeddings in the models. For training HSCNN, we sample 150,000 triplets for RCV1 and 300,000 triplets for Delicious; Delicious uses more triplets because it has more categories.

We mainly compare our approach with the Single CNN (black dashed line in Figure 1, named as "Sing.") in the cases of using different loss objectives list in Table 1. We also compare with additional baselines in the case of using BCE loss. "Naïve" is a hybrid solution using a vanilla Siamese network and Single network separately. "/CSS" uses HSCNN without the category-specific similarity and specific sampling. "/MT" uses HSCNN without multi-task architecture. "/CSS" and "/MT" are for the ablation test.

The parameters of the CNN architectures for all approaches refer to the ones used in exiting work (Shimura et al., 2018). The detailed parameters are list in the appendix. The evaluation metrics are Micro-F1, Macro-F1, Precision, and nDCG. For computing Precision and nDCG, we need to rank all of the categories for an instance. Here, the probabilities of head categories are obtained from the outputs of Single CNN; the similarities to a tail category is obtained from the outputs of the Siamese part of HSCNN. The probabilities of head categories and the similarities to the tail categories are not directly comparable, but the ranges of them are both in [0,1]. We just roughly rank the categories based on the probability/similarity directly. We evaluate the performance on the tail categories and the entire categories, respectively. We arbitrarily set the threshold $\mathcal{N}_\phi$ of the hybrid solutions as 100. RCV1 has 35 categories with $\mathcal{N}_c < 100$; Delicious has 472 such categories.

3.2 Experiments results

Table 3 lists the main experimental results. The left part of Table 3 shows the performance on the tail categories. First, comparing "Sing." and "Our", our approach can prominently improve the results of Single network in all cases of using vanilla or imbalanced loss objectives.

Second, comparing "Sing." and "Naïve", a "Naïve" hybrid method is even worse than "Sing.". One potential reason is that, in the head categories with a large number $\mathcal{N}$ of training instances, the number of these instances is sufficient to train a Single CNN. However, for the vanilla Siamese network, the number of potential training pairs is $O(\mathcal{N}^2)$, but we can only sample a small subset of them. Our multi-task component solves this problem. Another potential reason is that the vanilla Siamese network only has single similarity output and is limited for estimating the similarity for a large number of categories and multiple categories. For example, two instances that are "partially and almost" similar will have noise on the inconsistent categories if they are labeled as a similar pair. Our category-specific component solves this problem.

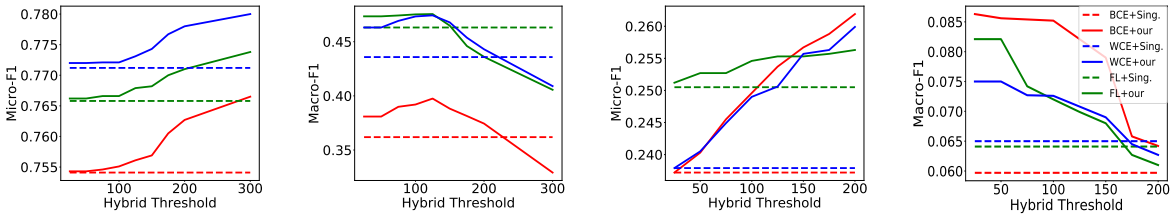
Third, comparing "Naïve" and "/CSS" (or "/MT" and "Our"), the proposed multi-task architecture can improve the performance. Comparing "Naïve" and "/MT" (or "/CSS" and "Our"), the proposed category-specific similarity and specific sampling can improve the performance. Using all additional technical attributes ("Our") can mutually benefit each other and improve the performance a lot.

The right part of Table 3 is the performance on entire categories. The observations on entire categories are consistent with that on tail categories. The improvement of Macro-F1 is more prominent than that of Micro-F1. It is because there are a large number of instances in head categories that influence the average computation of Micro-F1.

We also investigate the influences of different threshold $\mathcal{N}_\phi$ on the performance of entire categories. Figure 2 shows the results. Micro-F1 increases gradually as the threshold increases. Macro-F1 increases when the $\mathcal{N}_\phi$ is not too large and then decreases. It is because the numbers of training triplets are not enough for the categories with many

Data	Me.	**Tail** categories									**Entire** categories								
		BCE					WCE		FL		BCE					WCE		FL	
		Sing.	Naïve	/CSS	/MT	Our	Sing.	Our	Sing.	Our	Sing.	Naïve	/CSS	/MT	Our	Sing.	Our	Sing.	Our
RCV1	Mi.	6.83	5.79	6.10	6.83	**18.86**	18.27	**30.29**	27.83	**29.65**	75.41	75.05	75.13	75.40	**75.51**	77.12	**77.21**	76.58	**76.66**
	Ma.	3.96	3.37	3.47	3.88	**12.80**	12.02	**23.08**	17.54	**21.10**	36.19	35.96	36.00	36.17	**39.20**	43.58	**47.33**	46.31	**47.52**
	P@1	2.17	2.29	2.31	2.69	**3.72**	2.00	**3.76**	2.77	**3.99**	94.75	94.14	94.14	94.70	**94.90**	92.99	**93.59**	94.53	**94.86**
	P@3	1.69	1.35	1.47	1.55	**1.79**	1.39	**1.92**	**1.71**	1.69	77.60	77.53	77.54	77.56	**77.60**	76.55	**76.91**	77.48	**77.57**
	P@5	1.26	1.10	1.13	1.23	**1.26**	1.08	**1.25**	1.19	**1.26**	**54.38**	54.28	54.28	54.34	54.37	53.80	**54.02**	54.28	**54.40**
	G@1	2.71	2.29	2.31	2.69	**3.72**	2.00	**3.76**	2.77	**3.99**	94.75	94.14	94.14	94.70	**94.90**	92.99	**93.59**	94.53	**94.86**
	G@3	1.92	1.66	1.69	1.79	**1.99**	1.53	**2.31**	1.95	**2.36**	81.76	81.73	81.72	81.74	**81.77**	80.57	**80.93**	81.63	**81.76**
	G@5	1.51	1.35	1.37	1.49	**1.55**	1.27	**1.75**	1.53	**1.78**	64.56	65.53	64.54	64.57	**64.60**	63.73	**64.04**	64.49	**64.59**
Deli.	Mi.	2.43	1.86	1.93	2.41	**5.85**	2.97	**5.77**	2.40	**6.24**	23.72	12.56	16.53	23.72	**24.96**	23.79	**24.42**	25.05	**25.53**
	Ma.	1.51	1.47	1.48	1.50	**1.73**	1.86	**1.99**	1.56	**2.24**	5.97	5.76	5.89	5.93	**8.52**	6.50	**7.26**	6.41	**7.20**
	P@1	1.32	1.28	1.28	1.29	**8.45**	1.50	**8.55**	1.41	**8.42**	64.97	47.78	53.83	64.97	**65.04**	65.35	**65.57**	65.23	**65.89**
	P@3	1.13	1.13	1.13	1.13	**6.45**	1.12	**6.55**	1.27	**6.23**	58.96	39.14	48.00	57.99	**59.00**	58.81	**58.94**	58.36	**58.77**
	P@5	0.98	0.92	0.93	0.98	**5.47**	1.03	**5.49**	1.15	**5.31**	54.02	32.81	37.65	53.36	**54.06**	54.05	**54.10**	53.78	**53.99**
	G@1	1.32	1.28	1.28	1.29	**8.45**	1.50	**8.55**	1.41	**8.42**	64.97	47.78	53.83	64.79	**65.04**	65.35	**65.57**	65.23	**65.89**
	G@3	1.18	1.16	1.16	1.17	**6.89**	1.20	**6.99**	1.30	**6.73**	60.32	41.13	49.13	60.01	**60.37**	60.33	**60.47**	59.89	**60.34**
	G@5	1.07	1.03	1.05	1.07	**6.09**	1.12	**6.14**	1.21	**5.96**	56.56	36.24	39.45	55.87	**56.60**	56.63	**56.71**	56.29	**56.59**

Table 3: Results, $\mathcal{N}_\phi = 100$. Deli.: Delicious; Me.: Metric; Mi.: Micro-F1; Ma.: Macro-F1; G@k: nDCG@k.



(a). RCV1, Micro-F1 (b). RCV1, Macro-F1 (c). Delicious, Micro-F1 (d). Delicious, Macro-F1

Figure 2: Results on entire categories: change the threshold $\mathcal{N}_\phi$ for our hybrid solution.

Data	#triplets	Tail categories		Entire categories	
		Micro-F1	Macro-F1	Micro-F1	Macro-F1
RCV1	75,000	17.72	10.76	75.31	33.51
	150,000	18.86	12.80	75.51	39.20
	300,000	18.87	12.93	75.51	40.8
Deli.	150,000	1.26	1.61	23.44	5.45
	300,000	5.85	1.73	24.96	8.52
	450,000	5.83	1.81	24.95	8.57

Table 4: Experimental results with different number of sampled triplets for training HSCNN.

instances as the threshold $\mathcal{N}_\phi$ increases. Because the Macro-F1 scores are only worse than those of the baselines at about $\mathcal{N}_\phi > 225$ for RCV1 and $\mathcal{N}_\phi > 175$ for Delicious, the proposed HSCNN model based on few-shot technique is not only limited to few-shot categories but also performs well for the tail categories. The optimal $\mathcal{N}_\phi$ depends on the distribution of instance numbers of categories in a dataset; selecting a conservative value for $\mathcal{N}_\phi$ such as 50 or 100 is expected to obtain better results than the Single models.

Furthermore, Table 4 lists the results with different numbers of sampled triplets for training HSCNN. First, the number of sampled triplets should not be too small (e.g., RCV1 with 75,000 triplets). Second, the required number of sampled triplets to reach acceptable results depends on the dataset and possibly the number of categories, i.e., the number of categories of Delicious dataset is much larger than that of RCV1 dataset. RCV1 dataset with 150,000 already reaches a relatively high performance; Delicious dataset with 150,000 still has a relatively low performance. Third, a very large number of sampled triplets (e.g., RCV1 with 300,000 triplets) may still improve the performance but cannot improve the performance much more.

4 Conclusion

In this paper, we propose a hybrid solution with a HSCNN model for dealing with extremely imbalanced multi-label text classification. The proposed method can improve the performance of Single networks with diverse loss objectives on the tail categories or entire categories. In future work, we will try other types of Single networks (e.g., (Lai et al., 2015; Yang et al., 2016; Shimura et al., 2019)).

Acknowledgments

This work was partially supported by KDDI Foundation Research Grant Program.

References

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, page 1126–1135. JMLR.org.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5:361–397.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020a. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020b. Dice loss for data-imbalanced NLP tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 465–476, Online. Association for Computational Linguistics.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1340–1350, Florence, Italy. Association for Computational Linguistics.

T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. 2017. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 115–124, New York, NY, USA. Association for Computing Machinery.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2018. A simple neural attentive meta-learner. In *International Conference on Learning Representations*.

Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2554–2563, International Convention Centre, Sydney, Australia. PMLR.

Kazuya Shimura, Jiyi Li, and Fumiyo Fukumoto. 2018. HFT-CNN: Learning hierarchical category structure for multi-label short text categorization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 811–816, Brussels, Belgium. Association for Computational Linguistics.

Kazuya Shimura, Jiyi Li, and Fumiyo Fukumoto. 2019. Text categorization by learning predominant sense of words as auxiliary task. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1109–1119, Florence, Italy. Association for Computational Linguistics.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. 2016. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 761–769.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2008. Effective and efficient multilabel classification in domains with large number of labels. In *Proc. ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, volume 21, pages 53–59. sn.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2321–2331, Melbourne, Australia. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1206–1215, New Orleans, Louisiana. Association for Computational Linguistics.

A Appendices

Additional information on datasets and experimental settings are as follows.

Datasets: Figure 3 and 4 shows the distribution of the instance numbers of the categories in the datasets RCV1[1] and Delicious[2]. In both datasets, the instance numbers of the categories have long-tail distribution.

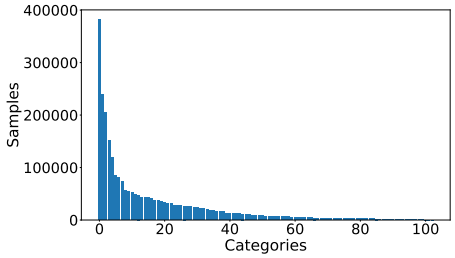


Figure 3: Distribution of instance numbers of categories: RCV1.

Experimental Settings: Table 5 lists the parameters of our model which refer to the common ones used in the existing works. We implement the methods by PyTorch. Dropout1 is after the embedded layer and Dropout2 is after the convolutional layer.

[1] www.ai.mit.edu/projects/jmlr/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm

[2] http://www.uco.es/kdis/mllresources/

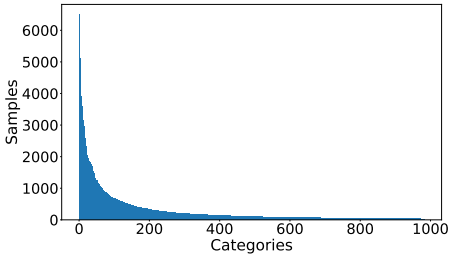


Figure 4: Distribution of instance numbers of categories: Delicious.

Description	Value	Description	Value
Filter size	3,4,5	Feature maps	128
Pooling	Max pooling	Hidden layers	1024
Activation	ReLu	Batch size	100
Word vectors	fastText	Activation function	Relu
Hidden layer	1024	Dropout1	0.25
Dropout2	0.5	Epoch	100*

Table 5: Model settings. *: with early stopping