Contrastive Learning-Enhanced Nearest Neighbor Mechanism for Multi-Label Text Classification

Xi'ao Su, Ran Wang, Xinyu Dai*
National Key Laboratory for Novel Software Technology, Nanjing University, China
Collaborative Innovation Center of Novel Software Technology and Industrialization, China
{nlp_suxa,wangr}@smail.nju.edu.cn, daixinyu@nju.edu.cn

Abstract

Multi-Label Text Classification (MLTC) is a fundamental and challenging task in natural language processing. Previous studies mainly focus on learning text representation and modeling label correlation. However, they neglect the rich knowledge from the existing similar instances when predicting labels of a specific text. To address this oversight, we propose a k nearest neighbor (kNN) mechanism which retrieves several neighbor instances and interpolates the model output with their labels. Moreover, we design a multi-label contrastive learning objective that makes the model aware of the kNN classification process and improves the quality of the retrieved neighbors during inference. Extensive experiments show that our method can bring consistent and considerable performance improvement to multiple MLTC models including the state-of-the-art pretrained and non-pretrained ones.

1 Introduction

Multi-Label Text Classification (MLTC) is a fundamental task in natural language processing, which can be found in many real-world scenarios such as web page tagging (Jain et al., 2016), topic recognition (Yang et al., 2016), sentiment analysis (Wang et al., 2016) and so on. Different from multi-class classification where only one label is identified as positive, MLTC aims to assign multiple labels from a predefined set to each text.

Till now, extensive research has been carried out to solve the MLTC task. Among them, some methods focus on learning enhanced text representation with deep neural networks (Kurata et al., 2016; Liu et al., 2016) or the label-wise attention mechanism (Xiao et al., 2019; Ma et al., 2021). Meanwhile, others try to model the label correlation by the sequential prediction (Nam et al., 2017; Yang et al., 2018), iterative reasoning (Wang et al., 2021), or graph neural networks (Ma et al., 2021).

Text	Labels
The **mutual information** of two random variables is commonly used in learning bayesian nets as well as in other fields ...	**math.ST** **math.IT** **stat.TH** **cs.IT** **cs.AI**
Mutual information is widely used, to measure the stochastic dependence of categorical random variables in order to address questions ...	**math.ST** **math.IT** **stat.TH** **cs.IT** **cs.AI** cs.LG

Table 1: An example of two papers from arXiv.

However, during inference, these methods neglect the rich knowledge which can be directly obtained from the existing training instances. Utilizing this knowledge can assist the model to predict more accurately. For example, Tab. 1 lists two papers from arXiv[1] along with their tags. Both papers research on "Mutual Information" and they have almost the same labels. If we are tagging the second paper, then we can easily get a good reference from the first one. Therefore, when predicting labels for a specific text, the model can get immediate and reliable help from the instances with similar texts.

To this end, for the first time, we solve the MLTC task by the use of k nearest neighbor (kNN) mechanism which can effectively utilize the knowledge from existing multi-label instances. Specifically, it retrieves several neighbor instances based on text representations generated by the MLTC model and interpolates the model prediction with their labels. Moreover, to make the model aware of the kNN process and improve the quality of retrieved neighbors, we propose to train the model with a contrastive learning (CL) objective. Existing super-

* Corresponding author.

[1] https://arxiv.org/

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 672 - 679

vised contrastive learning methods (Gunel et al., 2021; Li et al., 2021) are proposed under the conventional multi-class setting, where two instances are either positive or negative for each other. However, in MLTC, two instances may share some common labels while there may also be some labels that are unique to each instance. How to handle these cases is the key to utilizing contrastive learning in MLTC. We argue that simply treating these instance pairs as positive ones is sub-optimal due to the variable similarities in different instance pairs, which is verified in Section 4.2. To model more fine-grained correlations between multi-label instances, we design a multi-label contrastive learning objective with a dynamic coefficient for each instance pair based on the label similarity. Training with this objective encourages the model to generate closer representations for instance pairs with more shared labels and push away those pairs that have completely different labels. As a result, the kNN mechanism will retrieve instances that contain more relevant labels, thereby further improving the classification performance. It's worth noting that our method is of high versatility and can be directly applied to most existing MLTC models.

In summary, our contributions are as follows:

- We propose a k nearest neighbor mechanism for MLTC that directly utilizes the knowledge from the existing instances during inference.
- We design a multi-label contrastive learning objective which can effectively enhance the kNN mechanism for MLTC.
- Extensive experiments show that our method can consistently and considerably improve the performance of multiple existing MLTC models including the state-of-the-art pretrained and non-pretrained ones.

2 Related Work

Multi-label Text Classification Existing methods for MLTC mainly focus on learning text representation and modeling label correlation. At first, CNN (Kim, 2014; Kurata et al., 2016) and RNN-based (Liu et al., 2016) models were used to capture local and long-distance text dependencies. Besides, Xiao et al. (2019) proposed a label-specific attention network to focus on different tokens when predicting each label. The sequence generation model (Yang et al., 2018) and iterative reasoning mechanism (Wang et al., 2021) were utilized to model the label correlation. Furthermore, Ma et al. (2021) adopted graph neural networks based on label graphs. However, these methods are unable to refer to the existing instances that can guide the model to make better predictions.

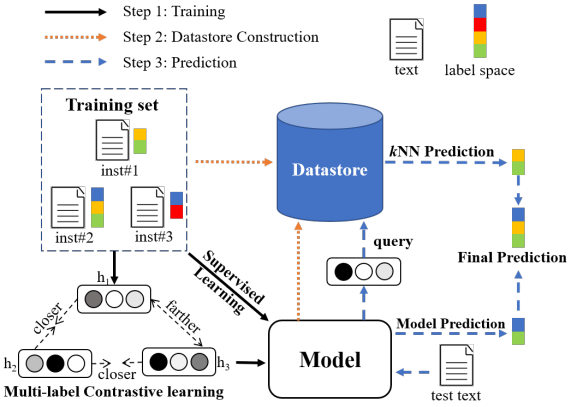


Figure 1: The overview of our proposed method.

Nearest Neighbor Methods in NLP Nearest neighbor methods have achieved great success in many NLP tasks such as language modeling (Khandelwal et al., 2020) and machine translation (Khandelwal et al., 2021; Zheng et al., 2021; Lin et al., 2021; Su et al., 2015). These methods utilize kNN retrieval in the inference stage based on context representation vectors which are generated by a converged model. Zheng et al. (2021) pointed out that simple application of the kNN method tends to introduce noise and we also found this issue in MLTC. Therefore, we design a multi-label contrastive learning objective to improve the quality of the retrieved neighbors. [2]

3 Proposed Method

In this section, we introduce our proposed method in detail. As depicted in Fig. 1, we design a k nearest neighbor mechanism for MLTC (Step 2, 3) and enhance it by training the model with a multi-label contrastive learning objective (Step 1).

3.1 Problem Formulation

Let $D = \{(x_i, y_i)\}_{i=1}^{N}$ be the MLTC training set consisting of N instances. Each x_i is a text and

[2]Contemporary with our work, KNN-BERT (Li et al., 2021) uses kNN and CL to enhance pretrained models' performance on multi-class classification. However, the way it uses kNN and sets positive/negative pairs in CL is inapplicable to multi-label scenarios due to its neglect of multiple non-exclusive labels in each instance, which is addressed by us in Section 3.3.

$y_i \in \{0,1\}^L$ denotes the corresponding multi-hot label vector where L is the total number of labels. The target of MLTC is to learn the mapping from the input text to the relevant labels.

3.2 Nearest Neighbor MLTC

To obtain knowledge from existing instances during inference, we propose a k nearest neighbor mechanism for MLTC including two steps: constructing a datastore of training instances (Step 2) and making the kNN prediction based on it (Step 3).

Datastore Construction Given an instance from the training set $(x_i, y_i) \in D$, the text representation vector $h_i = f(x_i)$ can be generated by an MLTC model. Then the multidimensional datastore D' can be constructed offline by a single forward pass over each training instance: $D' = \{(h_i, y_i)\}_{i=1}^{N}$.

Prediction In the inference stage, given an input text x, the model outputs the prediction vector $\hat{y}_{\text{Mo}} \in \{p|p \in [0,1]\}^L$. The model also outputs the text representation $f(x)$, which is utilized to query the datastore D' according to the euclidean distance to obtain the k nearest neighbors: $\mathcal{N} = \{(h_i, y_i)\}_{i=1}^{k}$. Then the kNN prediction can be made by:

$$\hat{y}_{\text{kNN}} = \sum_{i=1}^{k} \alpha_i y_i,\ \alpha_i = \frac{e^{-d(h_i, f(x))/\tau}}{\sum_j e^{-d(h_j, f(x))/\tau}} \quad (1)$$

where $d(\cdot,\cdot)$ indicates the euclidean distance, τ is the kNN temperature, and α_i denotes the weight of the i-th neighbor. Intuitively, the closer a neighbor is to the test instance, the larger its weight is. The final prediction is calculated as the combination of the base model output and the kNN prediction: $\hat{y} = \lambda \hat{y}_{\text{kNN}} + (1-\lambda)\hat{y}_{\text{Mo}}$ where λ is the proportion parameter.

3.3 Multi-Label Contrastive Learning

In MLTC, a model is usually trained by supervised learning with the binary cross-entropy (BCE) loss which is unaware of the kNN retrieval process. In consequence, retrieved neighbors may not have similar labels to the test instance and provide little help for the prediction. To fill this gap, we propose to train the model with a multi-label contrastive learning objective.

Existing supervised contrastive learning methods tried to narrow distances between instances from the same class and push away those from different classes. However, in MLTC, two instances may share some common labels while there may also be some labels that are unique to each instance. How to handle these cases is the key to utilizing contrastive learning in MLTC. Therefore, to model complex correlations among the multi-label instances, we design a dynamic coefficient based on the label similarity.

Considering a data minibatch of size b, we define a function to output all the other instances for a specific instance i: $g(i) = \{k|k \in \{1,2,\cdots,b\}, k \neq i\}$. The contrastive loss for each instance pair (i,j) can be calculated as:

$$\mathcal{L}_{\text{con}}^{ij} = -\beta_{ij} \log \frac{e^{-d(z_i,z_j)/\tau'}}{\sum_{k\in g(i)} e^{-d(z_i,z_k)/\tau'}} \quad (2)$$

$$C_{ij} = y_i^\top \cdot y_j,\ \beta_{ij} = \frac{C_{ij}}{\sum_{k\in g(i)} C_{ik}} \quad (3)$$

where $d(\cdot,\cdot)$ is the euclidean distance, τ' is the contrastive learning temperature and $z_i = f(x_i)$ denotes the text representation. C_{ij} denotes the label similarity between i,j which is computed by the dot product of their label vectors. The dynamic coefficient β_{ij} is the normalization of C_{ij}.

The contrastive loss for the whole minibatch is the summation over all the instance pairs: $\mathcal{L}_{\text{con}} = \sum_i \sum_{j\in g(i)} \mathcal{L}_{\text{con}}^{ij}$. For a pair of instances (i,j), the greater label similarity C_{ij} will bring larger coefficient β_{ij}, thereby increasing the value of their loss term $\mathcal{L}_{\text{con}}^{ij}$. As a result, their distance $d(z_i,z_j)$ will be optimized to be closer. Meanwhile, if they have no shared labels ($\beta_{ij} = C_{ij} = 0$), then the value of $\mathcal{L}_{\text{con}}^{ij}$ is also zero and their distance $d(z_i,z_j)$ will only appear in the denominators of other terms. Consequently, their distance will have negative gradients and be optimized to become far.

Denoting BCE loss as $\mathcal{L}_{\text{BCE}}$, the overall training loss of our method is: $\mathcal{L} = \mathcal{L}_{\text{BCE}} + \gamma\mathcal{L}_{\text{con}}$. The parameter γ controls the trade-off between losses.

Dataset	I	L	$\overline{\text{L}}$	$\overline{\text{W}}$
AAPD	55,840	54	2.4	163
RCV1-V2	804,414	103	3.2	124

Table 2: Statistics of the datasets. **I** and **L** denote the total number of instances and labels. $\overline{\text{L}}$ and $\overline{\text{W}}$ denote the average number of labels and words per instance.

4 Experiments

In this section, we conduct multiple experiments to evaluate the efficacy of our method. Implementa-

tion details and the overhead of our method can be found in Appendix A and B respectively.

4.1 Settings

Datasets To evaluate our method, we conduct experiments on two benchmark datasets AAPD (Yang et al., 2018) and RCV1-V2 (Lewis et al., 2004). The dataset statistics are listed in Tab. 2.

Evaluation Metrics Following the previous work (Yang et al., 2018), we adopt hamming loss and micro-F1 score as our evaluation metrics.

Baseline We adopt the following models as our baselines and apply our method to all of them:

CNN (Kim, 2014) uses multiple convolutional kernels to extract local text representations.

LDGN (Ma et al., 2021) is the state-of-the-art non-pretrained MLTC model. It is based on the label-wise attention network and a GCN.

BERT (Devlin et al., 2019) is a Transformer-based pretrained language model. Its [CLS] representation is used to do the classification.[3]

Models	AAPD		RCV1-V2	
	HL(-)	F1(+)	HL(-)	F1(+)
CNN	0.02378	69.60	0.00946	83.76
+ours	0.02248	71.69	0.00824	86.14
LDGN	0.02478	70.59	0.00863	86.00
+ours	0.02296	71.38	0.00768	87.29
BERT	0.02257	74.03	0.00766	87.54
+ours	**0.02167**	**75.18**	**0.00715**	**88.36**

Table 3: Performance of all the models. HL and F1 denote the hamming loss and micro-F1 (%). The symbol '+'/'-' indicates that the higher/lower the value is, the better the model performs. Best results are marked bold.

4.2 Results

Main Experiments As shown in Tab. 3, our method can bring consistent and considerable performance improvements to all of the models. For example, our method has improved the micro-F1 of CNN by 2.09% on AAPD and 2.38% on RCV1-V2 respectively. Moreover, both the state-of-the-art LDGN and powerful BERT can still benefit a lot from our method. Specifically, when equipped with our method, the non-pretrained model LDGN obtains competitive performances compared to the pretrained model BERT on the larger RCV1-V2.

[3]We also experimented on RoBERTa but it was outperformed by BERT in our task. Therefore, we choose BERT as the baseline pretrained model in our experiments.

Models	AAPD	RCV1-V2
CNN	69.60	83.76
CNN+kNN	70.19	85.21
CNN+CL	69.43	83.84
CNN+CL+kNN	71.69	86.14
LDGN	70.59	86.00
LDGN+kNN	70.73	86.76
LDGN+CL	70.44	86.51
LDGN+CL+kNN	71.38	87.29
BERT	74.03	87.54
BERT+kNN	74.22	87.84
BERT+CL	73.85	87.74
BERT+CL+kNN	75.18	88.36

Table 4: Micro-F1 (%) of the ablation tests. kNN and CL denote the k nearest neighbor mechanism and contrastive learning objective respectively.

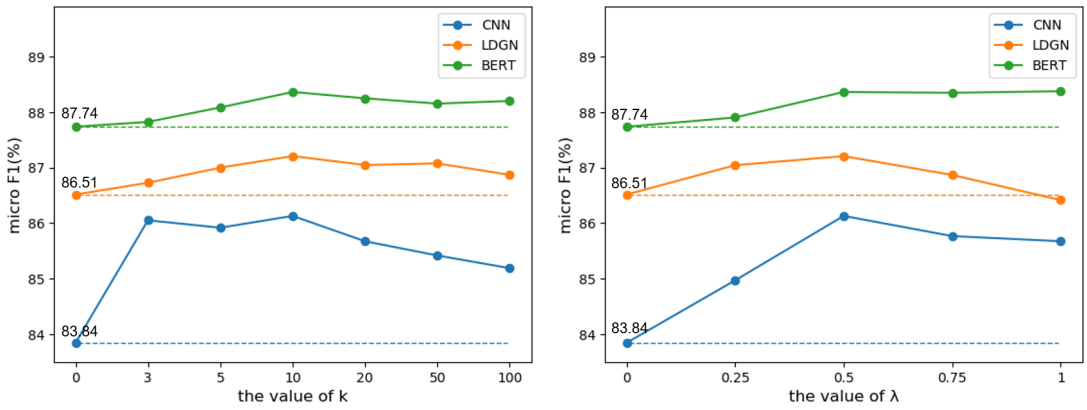
Ablation Test As mentioned above, our method consists of a k nearest neighbor mechanism (denoted as kNN) and a multi-label contrastive learning objective (denoted as CL). We demonstrate the effect of each component via an ablation test.

As shown in Tab. 4, the kNN mechanism can consistently improve the performance of the base models. Moreover, when equipped with our contrastive learning loss, although performances of the base models remain consistent, the improvements brought by the kNN mechanism have increased by a large margin. This verifies that our CL objective does effectively enhance the kNN mechanism.

Models	AAPD		RCV1-V2	
	w/o β	w/ β	w/o β	w/ β
CNN	71.19	71.69	85.27	86.14
LDGN	71.06	71.38	86.78	87.29
BERT	74.66	75.18	88.08	88.36

Table 5: Micro-F1 (%) of our methods with or without the dynamic coefficient β.

Analysis of Dynamic Coefficient In existing CL methods, two instances are either positive or negative for each other. To model more fine-grained similarity between instances, we proposed a dynamic coefficient β for each CL loss term (see Eq. 2,3).



(a) Anaylsis of the number of neighbors (b) Analysis of the proportion of kNN prediction

Figure 2: Hyperparameter analysis of the kNN mechanism on the RCV1-V2 dataset.

To verify the necessity of β, we also apply the simple extension of existing CL methods to MLTC[4]. As shown in Tab. 5, our method outperforms the simple extension method in all cases, which verifies the necessity of considering the fine-grained similarity between multi-label instances.

Analysis of kNN Paramters Here we conduct a parameter analysis of our kNN mechanism on the RCV1-V2 dataset. As shown in Fig. 2(a), for all the models, the performance improves at first and then decreases as the k increases. Moreover, when referring to neighbor instances ($k > 0$), the performance is always better than only using the model output ($k = 0$), which verifies the necessity of utilizing the knowledge from the existing instances. Fig. 2(b) demonstrates the trend of model performance with λ. In general, the trend is similar to that of k which further confirms that only using the model prediction ($\lambda = 0$) is sub-optimal. It's worth noting that on the BERT model, completely using neighbors' prediction ($\lambda = 1$) is highly competitive compared to the uniform combination ($\lambda = 0.5$) which performs the best on the other base models.

Impact of Contrastive Learning To further analyze the impact of our contrastive learning objective, for each test instance, we count the average proportion of shared labels to all labels brought by its nearest neighbors. As shown in Tab. 6, after training the model with contrastive learning, the retrieved instances contain more shared labels with the test instance, which further proves that CL does improve the quality of the retrieved neighbors. An intuitive example can be found in Appendix C.

[4]The extension method can be obtained by setting all the C_{ij} greater than 1 to 1 in Eq. 3. This means if two instances have any shared label, they are considered to be a positive pair.

Models	AAPD		RCV1-V2	
	w/o CL	w/ CL	w/o CL	w/ CL
CNN	64.5	65.5	82.7	84.2
LDGN	63.1	64.2	84.4	84.9
BERT	67.8	68.5	85.5	86.4

Table 6: The average proportion (%) of the shared labels to all labels brought by the nearest neighbors to each test instance with or without our CL objective.

5 Conclusion

In this paper, we proposed a k nearest neighbor mechanism along with a multi-label contrastive learning objective for MLTC. Extensive experiments verified the effectiveness of our method and revealed the source of performance improvements our method brings. For future work, we will explore how to improve the performance of MLTC models directly with contrastive learning.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive comments. This work was supported by NSFC Projects (Nos. 61936012 and 61976114), the National Key RD Program of China (No. 2018YFB1005102).

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of

deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Himanshu Jain, Yashoteja Prabhu, and Manik Varma. 2016. 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, San Francisco, CA, USA, August 13-17, 2016*, pages 935–944. ACM.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Gakuto Kurata, Bing Xiang, and Bowen Zhou. 2016. Improved neural network-based multi-label classification with better initialization leveraging label co-occurrence. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 521–526. The Association for Computational Linguistics.

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.

Linyang Li, Demin Song, Ruotian Ma, Xipeng Qiu, and Xuanjing Huang. 2021. KNN-BERT: fine-tuning pre-trained models with KNN classifier. *CoRR*, abs/2110.02523.

Huan Lin, Liang Yao, Baosong Yang, Dayiheng Liu, Haibo Zhang, Weihua Luo, Degen Huang, and Jinsong Su. 2021. Towards user-driven neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4008–4018. Association for Computational Linguistics.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2873–2879. IJCAI/AAAI Press.

Qianwen Ma, Chunyuan Yuan, Wei Zhou, and Songlin Hu. 2021. Label-specific dual graph neural network for multi-label text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3855–3864. Association for Computational Linguistics.

Jinseok Nam, Eneldo Loza Mencía, Hyunwoo J. Kim, and Johannes Fürnkranz. 2017. Maximizing subset accuracy with recurrent neural networks in multi-label classification. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5413–5423.

Jinsong Su, Deyi Xiong, Biao Zhang, Yang Liu, Junfeng Yao, and Min Zhang. 2015. Bilingual correspondence recursive autoencoder for statistical machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1248–1258. The Association for Computational Linguistics.

Ran Wang, Robert Ridley, Xi'ao Su, Weiguang Qu, and Xinyu Dai. 2021. A novel reasoning mechanism for multi-label text classification. *Inf. Process. Manag.*, 58(2):102441.

Yaqi Wang, Shi Feng, Daling Wang, Ge Yu, and Yifei Zhang. 2016. Multi-label chinese microblog emotion classification via convolutional neural network. In *Web Technologies and Applications - 18th Asia-Pacific Web Conference, APWeb 2016, Suzhou, China, September 23-25, 2016. Proceedings, Part I*, volume 9931 of *Lecture Notes in Computer Science*, pages 567–580. Springer.

Lin Xiao, Xin Huang, Boli Chen, and Liping Jing. 2019. Label-specific document representation for multi-label text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International*

Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 466–475. Association for Computational Linguistics.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 3915–3926. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1480–1489. The Association for Computational Linguistics.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. 2021. Adaptive nearest neighbor machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 368–374. Association for Computational Linguistics.

A Implementation Details

We implement all the methods relying on the PyTorch library[5]. We also use Faiss (Johnson et al., 2021) for fast nearest neighbor search. For CNN and BERT, we directly use the representations from the last hidden layer to construct the datastore. As for the LDGN which generates label-specific text representations, we perform a max-pooling operation on all the l vectors to get the single representation vector.

We train all the models on both datasets up to 30 epochs with an early stop of 3 patience and use the Adam optimizer with a learning rate of 1×10^{-3}. For all the models on AAPD, we use a batch size of 128. On RCV1-V2, we use a batch size of 512 for CNN and LDGN, and 128 for BERT due to its huge memory usage. As for the hyperparameters of our proposed method, $\lambda = 0.5$, $\tau = 1$, $\tau' = 10$ are adopted for all the cases. Besides, we use $k = 5$, $\gamma = 0.1$ for all the models on AAPD and $k = 10$, $\gamma = 0.01$ for those on RCV1-V2.

Models	AAPD	RCV1-V2
CNN	0.09 GB	1.46 GB
LDGN	0.11 GB	1.84 GB
BERT	0.17 GB	2.60 GB

Table 7: Disk usage of each datastore.

Models	AAPD		RCV1-V2	
	w/o kNN	w/ kNN	w/o kNN	w/ kNN
CNN	3.18	3.25	2.89	6.17
LDGN	5.47	7.29	7.61	9.67
BERT	264.89	267.57	265.96	270.73

Table 8: Inference time (ms/text) of different models with or without the kNN prediction. All results are tested with an RTX-2080Ti GPU.

B Space and Time Overhead

In the training stage, the overhead of contrastive learning is negligible compared to supervised learning, so we do not report it here. Most of the overhead lies in the kNN classifier. The disk usage of each datastore is shown in Tab. 7. The inference time per text of different models with or without the kNN prediction on each dataset is listed in

[5]https://pytorch.org/

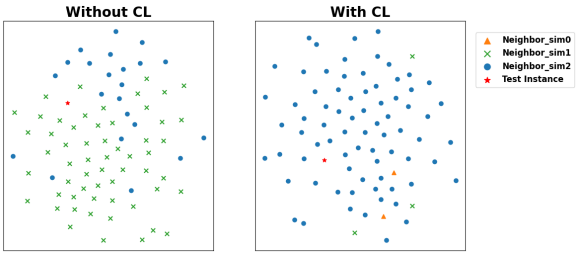


Figure 3: TSNE visualization where the red star stands for the test instance. Neighbors with different similarities to the test instance are plotted with different marks.

Tab. 8. It's worth noting that the extra inference time brought by our method does not exceed 5ms in all cases.

C Case Study: TSNE Visualization

In Fig. 3, we use the TSNE visualization tool to plot the CNN representations of a test instance and its 80 nearest neighbors with or without our CL objective. We use different marks to plot neighbors with different label similarities (C_{ij} in Eq. 3) to the test instance. As demonstrated in the left part, without contrastive learning, most of the nearest neighbors have only the similarity of 1 (green crosses). However, in the right part, with our CL objective, the test instance is surrounded by neighbors which have a high label similarity of 2 (blue circles). This confirms that our CL objective does improve the quality of the retrieved neighbors.

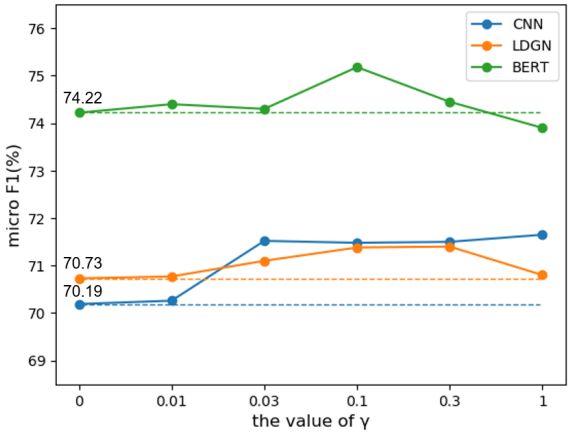


Figure 4: Analysis of the proportion of our contrastive learning objective based on each model.

D Analyzing the Proportion of Contrastive Learning

In this section, we analyze how the proportion of contrastive learning affects the performance of our method. As shown in Fig. 4, when trained with the contrastive learning objective ($\gamma > 0$), the performance of our method is better than that without contrastive learning ($\gamma = 0$) in most cases. However, when training the BERT model, too high proportion of contrastive learning ($\gamma = 1$) even hurts the performance. Besides, different base models have the different γ values for their optimal performance, which indicates that the proportion of contrastive learning to the overall training objective is crucial to the performance and varies with different model structures.