Learning Category Distribution for Text Classification

XIANGYU WANG and CHENGQING ZONG, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China

Label smoothing has a wide range of applications in the machine learning field. Nonetheless, label smoothing only softens the targets by adding a uniform distribution into a one-hot vector, which cannot truthfully reflect the underlying relations among categories. However, learning category relations is of vital importance in many fields such as emotion taxonomy and open set recognition. In this work, we propose a method to obtain the label distribution for each category (category distribution) to reveal category relations. Furthermore, based on the learned category distribution, we calculate new soft targets to improve the performance of model classification. Compared with existing methods, our algorithm can improve neural network models without any side information or additional neural network module by considering category relations. Extensive experiments have been conducted on four original datasets and 10 constructed noisy datasets with three basic neural network models to validate our algorithm. The results demonstrate the effectiveness of our algorithm on the classification task. In addition, three experiments (arrangement, clustering, and similarity) are also conducted to validate the intrinsic quality of the learned category distribution. The results indicate that the learned category distribution can well express underlying relations among categories.

CCS Concepts: • **Computing methodologies** → *Machine learning*; • **Natural language processing** → *Text classification;*

Additional Key Words and Phrases: Category distribution, text classification, neural networks

ACM Reference format:
Xiangyu Wang and Chengqing Zong. 2023. Learning Category Distribution for Text Classification. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 22, 4, Article 122 (April 2023), 13 pages.
https://doi.org/10.1145/3585279

1 INTRODUCTION

The text classification task has been widely studied in **natural language processing (NLP)**. There is a wide range of applications of text classification in our daily life, such as sentiment analysis [39], spam identification [1], and opinion mining [20]. A variety of supervised machine learning algorithms have been introduced in the field of text classification, such as support vector machine [15], k-nearest neighbor [40], and maximum entropy [24]. With the development of deep learning, many datasets as well as models have been proposed to achieve better performance on the text classification task. Graves and Schmidhuber [11] presented **bidirectional long short-term**

Authors' address: X. Wang and C. Zong, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China, 100049; emails: {xiangyu.wang, cqzong}@nlpr.ia.ac.cn.

2375-4699/2023/04-ART122 $15.00
https://doi.org/10.1145/3585279

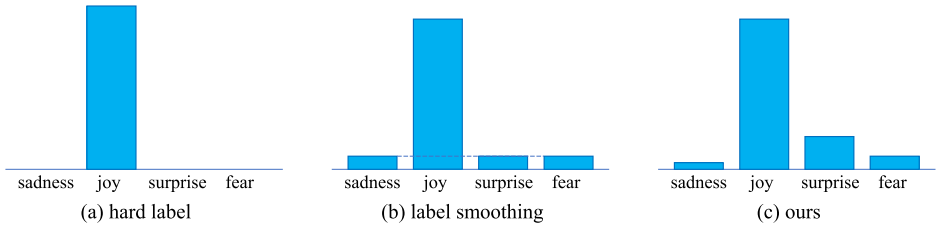


Fig. 1. Illustration of different category targets. The ground-truth emotion category is *joy*. (a) In hard label, the confidence of *joy* is set to be 1. (b) In label smoothing, the confidence of *joy* is set to be slightly less than 1 (such as 0.9), and the other categories share the rest of the confidence evenly. (c) In our algorithm, the confidence of *joy* is set to be slightly less than 1, and the other categories share the rest of the confidence depending on their similarities with *joy*.

memory (BiLSTM) in sequence processing tasks. Kim [17] introduced convolutional neural networks for text classification. More recently, many pre-trained language models such as ELMO [26], BERT [6], and XLNet [41] have shown their contribution to the NLP community.

The algorithms mentioned above focus on the specific structure of the model, and each category is regarded as an independent dimension. In neural network models, each category is represented with one-hot vectors, which are further employed as the target output of the model to minimize cross-entropy loss. The use of one-hot vectors results in two main problems. On the one hand, one-hot representation does not accord with the fact that different categories are not orthogonal to each other. On the other hand, the model trained with one-hot vectors tends to be overconfident [13]. Each instance may be related to multiple labels, especially when the categories are similar to each other. However, each instance is annotated as an independent category and represented with one-hot vectors when training the model. Therefore, the model that represented categories with one-hot vectors tends to be more confident.

Szegedy et al. [32] proposed a technique named label smoothing as shown in Figure 1(b), in which the one-hot labels are replaced with a weighted mixture of a one-hot vector and a uniform vector to calculate the cross-entropy loss. By focusing on the cross-entropy loss function rather than specific model architecture, label smoothing provides another way to improve the accuracy of the modern neural network models in many downstream tasks such as speech recognition [5], machine translation [22], and confidence calibration [23]. Nevertheless, the cosine similarity between different categories is equal to a constant in label smoothing representation. It is still unable to express the realistic category relations.

In real-world data, the relations between different categories are not easy to annotate. And the category relations are ignored in existing classification datasets. However, there are wide applications in many fields to reveal relations among categories. In psychological research, the quantitative analysis of emotion category relations is very helpful for the research of emotion taxonomy [19, 28]. In open set recognition, it benefits the detection of unknown categories to project existing categories into a vector space [9, 29].

In this work, we derive the label distribution for each category (category distribution) from the soft labels output by trained neural network models. Based on the learned category distribution, soft targets of each category are calculated to improve the performance of the model on the classification task. Experimental results demonstrate that our method is especially useful for the datasets with ambiguous labels or heavy noise. In addition, we detect the intrinsic quality of the learned category distribution in expressing underlying relations among categories from two perspectives (arrangement and clustering).

The main contributions of this work can be summarized as follows:

- We propose a novel algorithm to improve the label smoothing technique. Our algorithm does not introduce any additional neural network module. Experiments demonstrate that our algorithm outperforms baseline methods on the classification task.
- We construct 10 noisy datasets to validate the quality of our algorithm on noisy data. Experimental results indicate that our algorithm is especially useful for noisy data.
- We derive category distribution from the soft labels output by neural network models. As the vice product of our algorithm, category distribution is proved to be able to reveal underlying category relations effectively.

2 RELATED WORK

2.1 Text Classification

Text classification is a fundamental task that has been widely studied in a number of diverse domains, such as data mining, sentiment analysis, information retrieval, and medical diagnosis. Traditional text classification algorithms follow a two-step procedure. First, some artificial features are designed and extracted from the initial document [8, 38, 47]. Then, the features are fed into the algorithm to make the final classification decision [16, 25, 48]. With the breakthroughs in deep learning in recent years, many deep learning models have shown their success in text classification. Zhang et al. [46] introduced an empirical exploration on the use of character-level convolutional neural networks for text classification. Yao et al. [42] proposed a graph neural network model to enhance the text classification task by modeling a whole corpus as a heterogeneous graph. Wang et al. [36] presented a framework to combine explicit and implicit representations of short text for classification. Due to the ability to extract latent features directly from the initial documents, deep learning models have become much more popular in recent years.

2.2 Label Embedding

Label embedding is proposed in the domain of zero-shot image classification [3, 4]. Each category is represented with predefined attributes and the side information is also required to score the value for each category. In the NLP community, label embedding is used to convert the categories into semantic vector space [21, 35, 45]. In other words, each category is regarded as a special *word* and the embedding of the labels is also inputted into the model to enhance the text classification task. Different from previous studies that embedded categories into semantic space, Wang and Zong [37] proposed a framework to represent the emotion categories in emotion space, and the emotion relations are further detected with the distributed representations of emotion categories. In this work, we derive the category distribution and soft targets directly from the trained neural network model. Furthermore, the derived soft targets are employed to enhance the accuracy performance in the text classification task.

2.3 Soft Label

Soft labels have a higher entropy than one-hot hard labels and have been applied in a variety of applications. Hinton et al. [14] introduced knowledge distillation by using soft targets output by a trained large model as the ground-truth label to train a relatively smaller model. Phuong and Lampert [27] discussed the mechanisms of knowledge distillation by studying the special case of deep linear classifiers. For the purpose of preventing neural network models from being over-confident, Szegedy et al. [32] presented a label smoothing mechanism by smoothing the initial one-hot label with uniform distribution. Vyas et al. [33] proposed a meta-learning framework where the instance labels are treated as learnable parameters and updated with the model during

training. Zhang et al. [44] introduced an online label smoothing algorithm for image classification, in which the soft label of each instance will be added to a one-hot vector in every training step. Based on the label smoothing, Guo et al. [12] proposed the **label confusion model (LCM)** to enhance the text classification model. On the one hand, LCM requires an additional neural network module to calculate the soft label for the input instance. On the other hand, LCM is an instance-level model and generates the soft label only for instances, not categories. In this work, we derive label distribution for each category rather than each instance. The derived category distribution can well express category relations. Importantly, our method doesn't require any additional neural network module.

2.4 Label Distribution Learning

Geng [10] proposed **label distribution learning (LDL)**. LDL is a somewhat new machine learning task that paralleled with the classification task. In LDL, the true label distribution of each instance in the dataset is required to be pre-annotated. However, a majority of existing datasets are annotated with discrete categories, and they are not applied for LDL. However, it is very hard and expensive to annotate the true label distribution for each instance. A majority of existing datasets are annotated with discrete categories, and they are not applied for LDL. Therefore, our work is fundamentally different from LDL. In this work, the label distributions for each category are learned from the trained model. Our algorithm doesn't require any human annotating of the soft label.

3 OUR METHOD

In this section, we first discuss the potential loss bias caused by hard labels or label smoothing. To address this problem, we derive the category distribution that can express category relations. Based on the learned category distribution, the soft targets are calculated to improve model classification performance. The detailed approaches of our algorithm are listed last.

3.1 Loss of Neural Network Models

In neural network models, cross-entropy is chosen as the loss function for training. For example, given a dataset $\mathcal{D} = \{(x^{(i)}, \boldsymbol{y}^{(i)})_{i=1}^{N}\}$ annotated with C categories, for an instance annotated as category K, we have the loss function formula:

$$\mathcal{L}(x^{(i)}, \boldsymbol{y}^{(i)}|\boldsymbol{\theta}) = \sum_{j=1}^{C} -\boldsymbol{y}_j^{(i)} \log \boldsymbol{y}_j^{'(i)}, \tag{1}$$

where $\boldsymbol{y}'$ is the soft label predicted by the model, and θ is the model parameters to be trained.

In one-hot representation, $\boldsymbol{y}^{(i)}$ is expressed as

$$\boldsymbol{y}_j^{(i)} = \begin{cases} 1, & if\ j = K, \\ 0, & else. \end{cases} \tag{2}$$

Applying Equation (2) into Equation (1), we have the entropy loss of one-hot labels:

$$\mathcal{L}^{hard}(x^{(i)}, \boldsymbol{y}^{(i)}|\boldsymbol{\theta}) = -\log \boldsymbol{y}_K^{'(i)}. \tag{3}$$

In label smoothing, $\boldsymbol{y}^{(i)}$ can be expressed as

$$\boldsymbol{y}_j^{(i)} = \begin{cases} (1-\alpha) + \alpha/C, & if\ j = K, \\ \alpha/C, & else. \end{cases} \tag{4}$$

After applying Equation (4) into Equation (1), we obtain the entropy loss of label smoothing:

$$\mathcal{L}^{LS}(x^{(i)}, \boldsymbol{y}^{(i)}|\boldsymbol{\theta}) = -(1-\alpha)\log \boldsymbol{y}_K^{'(i)} - \alpha \sum_{j=1}^{C} \log \boldsymbol{y}_j^{'(i)}. \tag{5}$$

Although label smoothing outperforms the one-hot label by introducing a uniform distribution, it still cannot express realistic category relations. The similarity between different categories is not the same. Therefore, both the hard label and label smoothing are unable to describe realistic category relations, which causes calculated cross-entropy loss to deviate from the actual loss during training. Accurate category relations are essential to obtain a more realistic cross-entropy loss.

3.2 Derivation of Category Distribution

Inspired by knowledge distillation [14] where the soft labels output by the trained model tend to have more useful information than the hard label, we derive category distribution from the soft labels.

In an annotated dataset, each instance is actually a sample of the corresponding annotated category. In this article, we regard the soft label output by the trained model as the estimation of label distribution for the corresponding instance. Therefore, our goal is to minimize the loss between the category distribution and the instance distribution. Considering all instances are annotated as category K, we have

$$\min \sum_{i=1}^{N_K} Dist(\boldsymbol{y}, \boldsymbol{y^{(i)}}), \tag{6}$$

where $Dist$ is the distance function, $\boldsymbol{y}$ is the distribution of category K to be solved, $\boldsymbol{y^{(i)}}$ is the label distribution of the ith instance annotated as category K, and N_K is the number of instances annotated as category K in the dataset.

There are many functions to measure the distance between two distributions. Since $\boldsymbol{y}$ is the actual distribution to be derived and $\boldsymbol{y^{(i)}}$ is the predicted distribution of the ith instance, we choose KL-Divergence to measure the distance between true distribution ($\boldsymbol{y}$) and fitted distribution ($\boldsymbol{y^{(i)}}$):

$$\min \sum_{j=1}^{C} \sum_{i=1}^{N_K} \boldsymbol{y}_j \log \frac{\boldsymbol{y}_j}{\boldsymbol{y}_j^{(i)}}, \tag{7}$$

where $\boldsymbol{y}$ is the distribution of category K to be solved, and $\boldsymbol{y}_j$ is the jth component of vector $\boldsymbol{y}$.

By solving Equation (7), we have the formula to calculate the Kth category distribution:

$$\boldsymbol{y}_j = \frac{1}{N_K} \sum_{i=1}^{N_K} \boldsymbol{y}_j^{(i)}. \tag{8}$$

By concatenating the distribution of all categories, we obtain the final category distribution matrix:

$$\boldsymbol{Y} = [\boldsymbol{Y}_1; \boldsymbol{Y}_2; \ldots; \boldsymbol{Y}_C], \tag{9}$$

where $\boldsymbol{Y}_i$ is the distribution for the ith category. $\boldsymbol{Y}$ is a square matrix. The ith row of $\boldsymbol{Y}$ represents the distribution for the ith category. The jth column of $\boldsymbol{Y}$ represents the jth component of each category distribution.

To improve the performance of the models on the classification task, the similarity matrix of our category distribution is calculated as soft targets to train the model. The new soft label matrix is calculated as

$$\boldsymbol{S}_{ij} = \frac{e^{\boldsymbol{s}_{ij}/T}}{\sum_{m=1}^{C} e^{\boldsymbol{s}_{im}/T}}, \tag{10}$$

where S is the new soft targets, S_{ij} is the ith row and jth column element in S, and s_{ij} is the cosine similarity between Y_i and Y_j. T is the temperature parameter to control confidence in learning samples. A higher value of parameter T produces a softer probability distribution over categories.

3.3 Algorithm

Our algorithm benefits the community in two aspects. On the one hand, the category relations can be revealed with our category distribution, although these categories are one-hot represented in the dataset. On the other hand, based on our category distribution, soft targets are calculated for further improving the model performance on the classification task.

It should be noted that our algorithm does not require any additional neural network module. Just from the soft labels predicted by the model, we can in turn employ the soft labels to improve the model performance on the classification task. The detailed steps can be seen in Algorithm 1.

ALGORITHM 1: Category Distribution Algorithm

Input:
$\mathcal{D} = \{(x^{(i)}, \boldsymbol{y}^{(i)})_{i=1}^{N}\}$ // Dataset
Parameter:
$\boldsymbol{\theta}_0$ // initial random model parameters
T // temperature parameter
Output:
Y_{CD} // category distribution matrix
$\boldsymbol{\theta}_{CD}$ // trained model parameters
1: $S_0 \leftarrow I_C$
// set initial soft targets with hard labels
2: $\boldsymbol{\theta}_{HL} \leftarrow \min \sum_{i=1}^{n} \mathcal{L}((x^{(i)}, \boldsymbol{y}^{(i)})|S_0, \boldsymbol{\theta}_0)$
// train initial model with hard labels
3: $Y_{CD} \leftarrow$ with Equations (8) and (9)
// calculate our category distribution
4: $S_{CD} \leftarrow$ with Equation (10)
// calculate our soft targets for fine-tuning
5: $\boldsymbol{\theta}_{CD} \leftarrow \min \sum_{i=1}^{N} \mathcal{L}((x^{(i)}, \boldsymbol{y}^{(i)})|S_{CD}, \boldsymbol{\theta}_{HL})$
// fine-tune the model with our soft targets
6: **return** Y_{CD}; $\boldsymbol{\theta}_{CD}$

4 EXPERIMENTS

In this section, we first validate the ability of our algorithm to improve the model performance on text classification. Then, experiments are conducted to detect the intrinsic quality of the learned category distribution in expressing category relations.

4.1 Experimental Setup

4.1.1 Datasets. Five datasets that vary in the domain, topic, and languages are chosen to validate the effectiveness of our algorithm.

20NG (bydata version):[1] This is an English news dataset that consists of 20 news topics. There are 11,314 samples for training and 7,532 samples for testing.

[1] http://qwone.com/~jason/20Newsgroups.

THUCNews:[2] This is a Chinese news dataset proposed by Sun et al. [31]. There are 50,000, 10,000, and 5,000 samples for training, validation, and testing, respectively. There are 10 categories (*sports*, *finance*, *real estate*, *home*, *education*, *technology*, *fashion*, *politics*, *game*, and *amusement*) contained in the dataset.

NHKNews:[3] This is a Japanese news dataset. There are 21,795 instances annotated with up to 10 topics (*culture*, *sports*, *drama*, *information*, *anime*, *welfare*, *variety*, *report*, *education*, and *music*) in this dataset.

KRNews:[4] This is a Korean news dataset. There are 45,654 samples annotated with seven categories (*science*, *economy*, *society*, *culture*, *world*, *sports*, and *politics*) in this dataset.

FaizalNews:[5] This is an Indonesian news classification dataset. There are 9,000 and 1,000 samples for training and testing, respectively. There are five categories (*ball*, *health*, *finance*, *automotive*, and *property*) contained in this dataset.

4.1.2 Models. In this article, three typical neural network models are chosen to conduct experiments.

TextCNN: Kim [17] introduced the convolution neural network structure for text classification. Different from the CNN in image classification, the width of the convolution kernel is equal to the dimension of word vectors; 300-dimensional random vectors are adopted in our experiments.

BiLSTM: The bidirectional long-short time memory model was proposed by Graves and Schmidhuber [11]. BiLSTM is an improved version of a bidirectional recurrent neural network [30]; 300-dimensional random vectors are adopted in our experiments.

BERT: Bidirectional Encoder Representations from Transformers were introduced by Devlin et al. [6]. We choose the BERT-based model to fine-tune the datasets.

4.1.3 Settings. For TextCNN, the width of our convolutional kernel is 100, which is equal to the dimension of employed word vectors. The height of the kernel is divided into three groups (2,3,4). There are 64 channels in each group. We tune the batch size and learning rate to 128 and 0.001, respectively. For BiLSTM, batch size and learning rate are set to 128 and 0.001, which are the same as for TextCNN. The hidden layer size is set to 32 in each direction. For the BERT model, a fully connected layer is added on top of the pre-trained BERT-based model. Batch size and learning rate are separately set to 128 and 2e-5 for fine-tuning. For label smoothing, we set α to 0.9. The Adam optimizer is employed to train all three neural models in our work [18]. Our model is trained on CPU Intel(R) Xeon(R) E5-2620 and GPU GeForce RTX 3090. For *20NG*, *THUCNews*, and *FaizalNews*, we use original data split to train the models. For *NHKNews* and *KRNews* without original data split, we randomly split them into train, valid, and test sets with the ratio of 0.6:0.2:0.2.

4.2 Improvements on Text Classification

4.2.1 Test Performance. Three metrics (accuracy, recall, and macro-F1) are employed to show the performance of the models. Table 1 shows the test performance of hard label, label smoothing, and our algorithm with three models on five datasets. Our algorithm generally outperforms hard label and label smoothing. Our method has the most obvious improvement with the TextCNN network. Comparing three basic neural models, we can find that BERT outperforms TextCNN and BiLSTM on all five datasets. Correspondingly, the improvement of our algorithm on BERT is less than that on TextCNN and BiLSTM.

[2] http://thuctc.thunlp.org/.
[3] https://github.com/danyelkoca/NHK.
[4] https://github.com/eepLearning/Text-Classification-Korean-.
[5] https://github.com/faizalfakhrii/Text-Classification-XLNet/tree/main/dataset.

Table 1. Test Performance on Different Datasets

Models	20NG			THUCNews			NHKNews			KRNews			FaizalNews		
	acc	rec	f1	acc	rec	f1	acc	rec	f1	acc	rec	f1	acc	rec	f1
TextCNN+HL	82.87	82.81	82.78	88.20	88.17	88.17	60.39	58.40	58.66	72.85	71.91	72.19	90.21	89.80	89.94
TextCNN+LS	83.22	83.33	83.19	88.18	88.18	88.17	61.24	57.61	59.09	72.61	72.24	72.34	89.23	88.50	88.68
TextCNN+CD	83.86	83.61	**83.70**	88.82	88.83	**88.81**	61.31	59.13	**59.88**	73.01	72.90	**72.79**	90.47	90.40	**90.43**
BiLSTM+HL	76.24	76.48	76.23	87.82	87.74	87.76	57.78	56.74	56.63	69.11	68.72	68.75	79.90	80.10	79.86
BiLSTM+LS	77.50	77.46	**77.42**	87.98	87.90	87.91	57.05	55.97	56.21	68.84	67.77	68.10	80.92	80.70	**80.75**
BiLSTM+CD	77.27	77.39	77.22	88.29	88.04	**88.08**	58.78	56.98	**57.67**	69.42	68.95	**69.08**	80.80	80.50	80.45
BERT+HL	92.24	92.09	92.14	97.18	97.18	97.18	76.07	73.94	74.88	80.58	80.43	80.47	92.72	92.60	92.64
BERT+LS	92.17	92.29	92.21	97.22	97.21	97.21	77.60	74.12	75.54	80.87	80.61	**80.69**	92.53	92.40	92.44
BERT+CD	92.48	92.66	**92.56**	97.23	97.23	**97.23**	76.46	76.42	**76.34**	80.86	80.50	80.63	93.09	92.80	**92.87**

HL, LS, and CD are the abbreviations of hard label, label smoothing, and category distribution, respectively. Best macro-F1 results are shown in bold.

Table 2. Test Performance of Each Category on 20NG Datasets with TextCNN Model

	Hard Label			Label Smoothing			Category Distribution		
	acc	rec	f1	acc	rec	f1	acc	rec	f1
alt.atheism	90.51	90.51	90.51	91.85	90.51	91.18	95.42	91.24	93.28
comp.graphics	71.68	68.89	70.25	68.36	67.22	67.79	67.18	72.78	69.87
comp.os.ms-windows.misc	68.02	74.44	71.09	78.95	75.00	76.92	79.04	73.33	76.08
comp.sys.ibm.pc.hardware	72.22	66.82	69.42	72.64	68.22	70.36	70.14	72.43	71.26
comp.sys.mac.hardware	74.24	76.17	75.19	79.69	79.27	79.48	73.85	74.61	74.23
comp.windows.x	73.41	67.55	70.36	67.31	74.47	70.71	71.81	71.81	71.81
misc.forsale	74.16	85.16	79.28	71.36	83.52	76.96	80.11	81.87	80.98
rec.autos	82.63	84.62	83.61	82.41	85.58	83.96	81.06	88.46	84.60
rec.motorcycles	89.01	89.47	89.24	91.89	89.47	90.67	92.97	90.53	91.73
rec.sport.baseball	82.49	89.50	85.85	83.73	87.50	85.57	83.25	87.00	85.09
rec.sport.hockey	88.61	89.50	89.05	87.88	87.00	87.44	88.61	89.50	89.05
sci.crypt	95.92	91.71	93.77	98.94	91.22	94.92	93.50	91.22	92.35
sci.electronics	78.95	78.57	78.76	80.95	72.86	76.69	78.95	78.57	78.76
sci.med	83.14	78.14	80.56	81.97	81.97	81.97	84.02	77.60	80.68
sci.space	89.09	91.59	90.32	87.73	90.19	88.94	89.25	89.25	89.25
soc.religion.christian	92.23	88.56	90.36	94.27	90.05	92.11	95.24	89.55	92.31
talk.politics.guns	87.88	90.62	89.23	91.44	89.06	90.24	86.87	89.58	88.21
talk.politics.mideast	95.92	92.16	94.00	95.92	92.16	94.00	95.94	92.65	94.26
talk.politics.misc	90.62	84.30	87.35	81.72	88.37	84.92	90.00	88.95	89.47
talk.religion.misc	76.67	77.97	77.31	75.38	83.05	79.03	80.00	81.36	80.67
macro average	82.87	82.81	82.78	83.22	83.33	83.19	**83.86**	**83.61**	**83.70**

Best macro-F1 results are shown in bold.

Table 2 shows the detailed test performance on each category in the 20NG dataset with the TextCNN model. Our algorithm outperforms label smoothing in 14 categories. There are three main categories with only one sub-category in the 20NG dataset (*alt.atheism*, *misc.forsale*, and *soc.religion.christian*). Particularly, our algorithm outperforms label smoothing in all three categories, with improvement ranging from 0.20 to 4.02 on a macro-F1 score.

4.2.2 Analysis. As mentioned above, our soft targets are calculated based on the learned category distribution, which means our algorithm improves the model by considering category relations. Therefore, for datasets where the categories are easy to distinguish, the improvement of our algorithm is limited. This view can be validated from Table 1. There are five topics annotated in

Table 3. Test Performance on Noisy Data of THUCNews Dataset

Models	5% Noise			10% Noise			20% Noise			30% Noise		
	acc	rec	f1	acc	rec	f1	acc	rec	f1	acc	rec	f1
TextCNN+HL	87.20	87.15	87.17	84.71	84.64	84.66	83.24	83.20	83.20	81.89	81.85	81.86
TextCNN+LS	87.31	87.28	87.27	85.09	85.03	85.05	83.56	83.52	83.53	82.57	82.51	82.52
TextCNN+CD	88.02	88.00	**87.99**	86.48	86.47	**86.45**	84.79	84.72	**84.73**	83.16	83.15	**83.13**
BiLSTM+HL	81.12	81.11	81.10	79.61	79.55	79.57	74.07	74.05	74.05	69.09	69.01	69.02
BiLSTM+LS	82.13	82.04	82.05	80.68	80.67	80.65	74.77	74.69	74.72	70.91	70.81	70.84
BiLSTM+CD	82.59	82.55	**82.54**	80.87	80.80	**80.81**	75.62	75.52	**75.55**	71.34	71.33	**71.32**
BERT+HL	97.07	97.06	97.06	96.67	96.67	96.67	96.56	96.55	96.55	95.67	95.66	95.66
BERT+LS	97.00	97.00	97.00	96.60	96.59	96.59	96.82	96.81	**96.81**	95.87	95.86	95.86
BERT+CD	97.14	97.13	**97.13**	96.93	96.92	**96.92**	96.67	96.66	96.66	96.03	96.01	**96.01**

HL, LS, and CD represent hard label, label smoothing, and category distribution, respectively. The percentage represents the proportion of the samples that are randomly re-labeled.

FaizalNews, where the categories are much easier to distinguish than other datasets. Three baseline models with hard labels all have a high performance. As a result, the improvement of our algorithm is not as significant as other datasets.

On the contrary, our algorithm is especially useful for datasets where the boundaries of categories are quite hard to define. This can also be validated from Table 1. *NHKNews* is a dataset with lots of noise. The incorrect annotated instances make the label relations fuzzy. As a result, the improvement of our algorithm is more significant than the others. It is worth pointing out that the easier it is to define the boundaries between categories, the easier it is to classify the dataset. For the datasets where the label boundaries are not clear, our algorithm is helpful to discover label relations and improve the model performance. Therefore, our algorithm is more helpful for hard datasets than easy datasets.

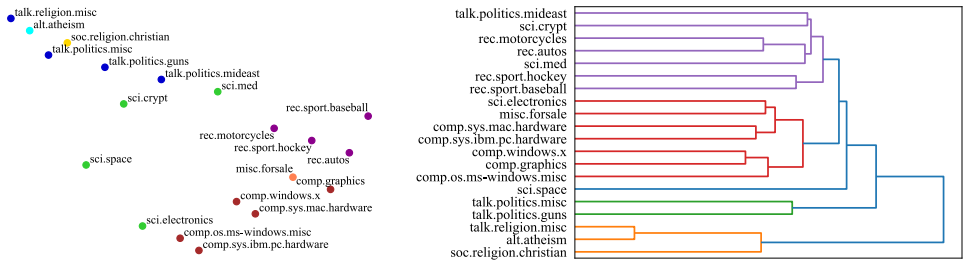
4.3 Tolerance to Noisy Data

It is hard to annotate all samples correctly, especially when the categories are similar to each other. Learning from noisy data is a problem with great practical importance. However, generalization of deep neural networks to noisy data is very harmful [43]. In this section, we find that our approach can improve the performance of neural networks by reducing the confidence in learning noisy data.

To better show the performance of our method on noisy data, we construct a series of noisy datasets based on THUCNews. For each noise data, only training data are randomly re-labeled with a certain noise proportion, and the test data remain unchanged. We construct four noise datasets with different noise proportions (5%, 10%, 20%, and 30%). We choose TextCNN, BiLSTM, and BERT as base prediction models. Three metrics (accuracy, recall, and macro-F1) are employed to evaluate the model performance. The detailed results are listed in Table 3.

With noise proportion increasing, the test accuracy, recall, and macro-F1 of all models decrease. On all four noisy datasets and all three models, our algorithm generally outperforms the hard label and label smoothing. For three neural models, the macro-F1 score of BiLSTM drops 12.08% on the noisy THUCNews dataset, which indicates that BiLSTM is most sensitive to noise. The macro-F1 score of BERT drops only 1.40%. As a pre-trained language model, BERT demonstrates its strong power against noise.

As listed in Table 3, our algorithm is especially useful on TextCNN and BiLSTM. With noise proportion increasing (from 5% to 30%), our algorithm outperforms label smoothing by 1.44 to 2.30 percentage points on the macro-F1 score. The baselines of the BERT model are so high that



(a) arrangement (b) clustering

Fig. 2. Category distribution obtained by our algorithm on the 20NG dataset. (a) Visualization of category distribution. There are 7 major categories and 20 sub-categories contained in the dataset. Seven major categories are represented with seven colors, respectively. (b) The cluster dendrogram of category distribution. The dendrogram is constructed using linkage clustering.

there a little room for improvement. Nonetheless, the improvement of our method increases (from 0.07 to 0.4) on the macro-F1 score with noise proportion increasing. The noisy experiments demonstrate the effectiveness of our algorithm on all three modern neural networks.

4.4 The Vice Product: Category Distribution

Existing classification datasets regard categories as independent ones. However, it is very important to detect category relations in many fields [2, 7]. In this section, we detect the intrinsic quality of the proposed category distribution in expressing category relations from two aspects: arrangement and clustering. We employ the 20NG dataset in this section as there are 7 major categories and 20 sub-categories in the 20NG dataset.

4.4.1 Arrangement of Category Distribution. The dimension of category distribution is equal to the number of categories. To better show the arrangement of the categories, we first use **singular value decomposition (SVD)** [34] to reduce the dimension of the category distribution from 20 to 2. Then, two-dimensional vectors are replaced with their rank order, which remains the relative relations among them. The two-dimensional vectors are displayed in Figure 2. All sub-categories that belong to the same major category are represented with the same color.

There are four major categories (*comp*, *rec*, *sci*, and *talk*) that contain multiple sub-categories. They are colored with brown, green, purple, and blue. For these four major categories, each one is linearly separated from the other three. Although the categories are annotated with one-hot vectors, our proposed category distribution can still well express potential relations among major categories.

As for the major category *talk* (in blue), the sub-category *talk.religion* (left top in blue) is far away from the other three sub-categories that belong to *talk.politics*, which is consistent with the fact that *talk.religion* and *talk.politics* belong to different sub-topics. As for *alt.atheism* (left top in cyan) and *soc.religion.christian* (left top in gold), they are very close to each other, although they belong to different major categories. What's more, the sub-category *talk.religion.misc* is close to both *soc.religion.christian* and *alt.atheism*. This is accordant to the fact that *talk.religion.misc*, *alt.atheism,* and *soc.religion.christian* are highly related with religion.

The sub-category *sci.electronics* (middle bottom in green) is close to the major category *comp* (right bottom in brown), which is consistent with the fact that electronics and computers are closely related. As for the major category *misc.forsale* (right middle in coral), it is very interesting that *misc.forsale* is located between *comp* (in brown) and *rec* (in purple) but far away from *talk.politics*

and religion-related categories. This is very reasonable as sub-categories in *comp* and *rec* are products that can be traded, but religion and politics are cultural concepts that cannot be traded.

From this experiment, we can conclude that our category distribution can not only well express the relations among major categories but also well capture the relations among sub-categories.

4.4.2 Clustering of Category Distribution. In this section, we perform the cluster analysis on our category distribution. We choose the *linkage* function in the scipy package[6] to conduct this experiment. For function parameters, we choose the Ward algorithm and Euclidean distance.

The cluster dendrogram of category distribution can be seen in Figure 2. Although the clustering results are not completely consistent with human clustering results, we can still find several common features.

All sub-categories that belong to *comp* are colored in red, which demonstrates that our category distribution can well distinguish the categories related to the computer topic. Moreover, *sci.electronic* is also marked in red, which means *sci.electronic* is close to *comp* and is accordant with the results in the above experiment.

All four sub-categories that belong to *rec* are colored in purple. In addition, two sub-categories in the *sci* topic and one sub-category in *talk* are also marked in purple. This indicates that two sub-categories in different major categories can be close to each other, which further suggests the complexity of clustering the categories.

As shown in Figure 2, *alt.atheism*, *soc.religion.christian,* and *talk.religion.misc* are very close to each other and colored orange. These three categories seem to be far away from the other categories. It is reasonable as they are highly related with the religion topic while the others are not. *sci* seems to be the most irregular major category. Although *sci* contains four sub-categories, these sub-categories are not clustered together.

5 LIMITATIONS

In this article, we discuss how to extract category relations from text classification datasets and further improve the classification performance of neural network models. However, it should be noted that the extracted category relations can only reflect the relations in the data space, not the semantic space. Also, the category relations may change with the choice of the dataset and the classification model. How to obtain dataset-unrelated category relations in semantic space remains a change.

6 CONCLUSION AND FUTURE WORK

In this article, we argue that label smoothing is unable to well express category relations. To address this problem, we propose an algorithm to obtain category distribution to reveal category relations. Based on the learned category distribution, new soft targets are calculated for further model fine-tuning. Experimental results demonstrate the effectiveness of our algorithm in improving model classification performance and the learned category distribution in expressing underlying category relations. Moreover, our algorithm doesn't require any additional neural network module and can be easily employed in existing neural network models.

There are two avenues of future work we would like to explore. On the one hand, existing deep models tend to be overconfident. Training with soft labels can reduce model confidence in making predictions. It is interesting to detect the ability of category distribution in confidence calibration. On the other hand, category distribution, as well as label smoothing, is useful only on single-label datasets. It is very meaningful to apply these methods to multi-label datasets.

[6] https://github.com/scipy/scipy/blob/v1.7.1/scipy/cluster/hierarchy.py#L837-L1081.

REFERENCES

[1] Jacob Abernethy, Olivier Chapelle, and Carlos Castillo. 2008. Web spam identification through content and hyperlinks. In *Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web*. 41–44.

[2] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. 2013. Label-embedding for attribute-based classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 819–826.

[3] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. 2015. Label-embedding for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 7 (2015), 1425–1438.

[4] Chen Chen, Haobo Wang, Weiwei Liu, Xingyuan Zhao, Tianlei Hu, and Gang Chen. 2019. Two-stage label embedding via neural factorization machine for multi-label classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33(1). 3304–3311.

[5] Jan Chorowski and Navdeep Jaitly. 2017. Towards better decoding and language model integration in sequence to sequence models. In *Proc. Interspeech 2017*. 523–527.

[6] 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, Volume 1 (Long and Short Papers)*. 4171–4186.

[7] Paul Ed Ekman and Richard J. Davidson. 1994. *The Nature of Emotion: Fundamental Questions*. Oxford University Press.

[8] Richard S. Forsyth and David I. Holmes. 1996. Feature-finding for text classification. *Literary and Linguistic Computing* 11, 4 (1996), 163–174.

[9] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. 2020. Recent advances in open set recognition: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43, 10 (2020), 3614–3631.

[10] Xin Geng. 2016. Label distribution learning. *IEEE Transactions on Knowledge and Data Engineering* 28, 7 (2016), 1734–1748.

[11] Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18, 5–6 (2005), 602–610.

[12] Biyang Guo, Songqiao Han, Xiao Han, Hailiang Huang, and Ting Lu. 2021. Label confusion learning to enhance text classification models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 12929–12936.

[13] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. 1321–1330.

[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[15] Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European Conference on Machine Learning*. Springer, 137–142.

[16] Sang-Bum Kim, Kyoung-Soo Han, Hae-Chang Rim, and Sung Hyon Myaeng. 2006. Some effective techniques for naive Bayes text classification. *IEEE Transactions on Knowledge and Data Engineering* 18, 11 (2006), 1457–1466.

[17] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. 1746–1751.

[18] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 2015 Conference on International Conference on Learning Representations (ICLR)*, 1–15.

[19] Assaf Kron. 2019. Rethinking the principles of emotion taxonomy. *Emotion Review* 11, 3 (2019), 226–233.

[20] Bing Liu and Lei Zhang. 2012. A survey of opinion mining and sentiment analysis. In *Mining Text Data*. Springer, 415–463.

[21] Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers (COLING'16)*. 171–180.

[22] Clara Meister, Elizabeth Salesky, and Ryan Cotterell. 2020. Generalized entropy regularization or: There's nothing special about label smoothing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 6870–6886.

[23] Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help? *Advances in Neural Information Processing Systems* 32 (2019), 4694–4703.

[24] Kamal Nigam, John Lafferty, and Andrew McCallum. 1999. Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering*, Vol. 1. 61–67.

[25] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine Learning* 39, 2 (2000), 103–134.

[26] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 2227–2237.

[27] Mary Phuong and Christoph Lampert. 2019. Towards understanding knowledge distillation. In *International Conference on Machine Learning*. 5142–5151.

[28] James A. Russell and James H. Steiger. 1982. The structure in persons' implicit taxonomy of emotions. *Journal of Research in Personality* 16, 4 (1982), 447–469.
[29] Walter J. Scheirer, Lalit P. Jain, and Terrance E. Boult. 2014. Probability models for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36, 11 (2014), 2317–2324.
[30] Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.
[31] Maosong Sun, Jingyang Li, Zhipeng Guo, Z. Yu, Y. Zheng, X. Si, and Z. Liu. 2016. Thuctc: An efficient Chinese text classifier. *GitHub Repository* (2016).
[32] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. IEEE, 2818–2826.
[33] Nidhi Vyas, Shreyas Saxena, and Thomas Voice. 2020. Learning soft labels via meta learning. *arXiv preprint arXiv:2009.09496* (2020).
[34] Michael E. Wall, Andreas Rechtsteiner, and Luis M. Rocha. 2003. Singular value decomposition and principal component analysis. In *A Practical Approach to Microarray Data Analysis*. Springer, 91–109.
[35] 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)*. 2321–2331.
[36] Jin Wang, Zhongyuan Wang, Dawei Zhang, and Jun Yan. 2017. Combining knowledge with deep convolutional neural networks for short text classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2915–2921.
[37] Xiangyu Wang and Chengqing Zong. 2021. Distributed representations of emotion categories in emotion space. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 2364–2375. https://doi.org/10.18653/v1/2021.acl-long.184
[38] Rui Xia, Feng Xu, Chengqing Zong, Qianmu Li, Yong Qi, and Tao Li. 2015. Dual sentiment analysis: Considering two sides of one review. *IEEE Transactions on Knowledge and Data Engineering* 27, 8 (2015), 2120–2133.
[39] Rui Xia, Chengqing Zong, and Shoushan Li. 2011. Ensemble of feature sets and classification algorithms for sentiment classification. *Information Sciences* 181, 6 (2011), 1138–1152.
[40] Yiming Yang. 1999. An evaluation of statistical approaches to text categorization. *Information Retrieval* 1, 1 (1999), 69–90.
[41] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R. Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems* 32 (2019).
[42] Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33(1). 7370–7377.
[43] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Commun. ACM* 64, 3 (2021), 107–115.
[44] Chang-Bin Zhang, Peng-Tao Jiang, Qibin Hou, Yunchao Wei, Qi Han, Zhen Li, and Ming-Ming Cheng. 2021. Delving deep into label smoothing. *IEEE Transactions on Image Processing* 30 (2021), 5984–5996.
[45] Honglun Zhang, Liqiang Xiao, Wenqing Chen, Yongkun Wang, and Yaohui Jin. 2018. Multi-task label embedding for text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 4545–4553.
[46] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems* 28 (2015), 649–657.
[47] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. 2010. Understanding bag-of-words model: A statistical framework. *International Journal of Machine Learning and Cybernetics* 1, 1–4 (2010), 43–52.
[48] Chengqing Zong, Rui Xia, and Jiajun Zhang. 2021. *Text Data Mining*. Springer.

Received 3 March 2022; revised 26 December 2022; accepted 6 February 2023