# Cluster-gated Convolutional Neural Network for Short Text Classification

**Haidong Zhang, Wancheng Ni, Meijing Zhao, Ziqi Lin**
Institute of Automation, Chinese Academy of Sciences, China
haidong_zhang14@yahoo.com
{wancheng.ni, meijing.zhao, linziqi2013}@ia.ac.cn

## Abstract

Text classification plays a crucial role for understanding natural language in a wide range of applications. Most existing approaches mainly focus on long text classification (e.g., blogs, documents, paragraphs). However, they cannot easily be applied to short text because of its sparsity and lack of context. In this paper, we propose a new model called cluster-gated convolutional neural network (CGCNN), which jointly explores word-level clustering and text classification in an end-to-end manner. Specifically, the proposed model firstly uses a bi-directional long short-term memory to learn word representations. Then, it leverages a soft clustering method to explore their semantic relation with the cluster centers, and takes linear transformation on text representations. It develops a cluster-dependent gated convolutional layer to further control the cluster-dependent feature flows. Experimental results on five commonly used datasets show that our model outperforms state-of-the-art models.

## 1 Introduction

With the rapid development of social media, e-commerce and on-line communication, the Internet has been generating an increasing amount of short texts, including texts, search snippets, user reviews for products, etc., which poses an urgent demand for understanding them. Short text classification, assigning predefined categories to texts, is a fundamental technique in natural language processing, and plays an important role in a wide range of applications, such as sentiment analysis, web searching, and ads matching.

In prior research, much progress has been made on text classification, including traditional approaches based on human-designed features (Lazaridou et al., 2013; Zhang et al., 2015a) and neural networks based on deep architectures (Lai et al., 2015; Yang et al., 2016). However, such methods prefer to deal with documents and paragraphs, and still have limitations for short texts. Each short text does not have enough words, which may result in data sparsity and lack of contexts (Wang et al., 2017).

Some researchers incorporated knowledge bases into traditional approaches (Feng et al., 2013; Wang et al., 2014) or neural networks (Wang et al., 2017) to overcome these challenges. Extra resources can provide abundant semantic information for short text classification, but the performance of such methods is strongly dependent on the quality of knowledge bases and constructing a large-scale knowledge base is time-consuming and labor-intensive. Another strategy is to explore latent topics (Chen et al., 2011; Ren et al., 2016) or clustering features (Ma et al., 2015; Revanasiddappa and Harish, 2018) for texts and input them into some classifiers as features. Such methods can reduce high dimensionality and terms' sparse distribution problems. Their shortness is to use pre-trained topics or clusters as features, which might be hard to explore the potential association between clustering and classification.

To address the limitations, we construct a joint architecture to embed a soft clustering method into the classification task, because joint architectures can leverage mutual information for each other and have been useful in many studies for understanding natural language (Luo et al., 2015; Shao et al., 2017; Schmitt et al., 2018). In addition, convolutional neural network and the gated mechanisms have been proven effectiveness in sentence-level language modeling (Dauphin et al., 2016; Gehring et al., 2017), and cluster centers of words contain semantic closeness of similar ones, which motivate



*Proceedings of the 23rd Conference on Computational Natural Language Learning*, pages 1002–1011
Hong Kong, China, November 3-4, 2019. 

us to utilize them to auto-extract and highlight the cluster-related features for classification.

Based on the above analysis, we propose a joint model called cluster-gated convolutional neural network (CGCNN), coupling clustering and classification methods, to construct an end-to-end deep architecture. It integrates a soft clustering method into a gated convolutional neural network, which can explore the semantic relation of word-level context and the global corpus. And it can also guide the gating mechanism unit to control cluster-dependent feature flows. Specifically, it firstly uses a bi-directional long short-term memory model (BiLSTM) to learn word representations and capture local context in text. Then, it performs a soft clustering method on word representations for the probability of each word assigning to each cluster, which can build a bridge between word and the global corpus. And we develop a linear transformation to calculate cluster-dependent text representations. Based on the gating mechanism, we uses the cluster centers to further highlight the cluster-dependent convolutional features for the corresponding cluster. At last, we perform max-over-time pooling and concatenation operations to combine the selected features for classification.

The main contributions of this study are summarized as follows:

- We develop a joint model that combines clustering and classification methods in an end-to-end manner. The model leverages the semantic relation of words and the global corpus by learning from a soft clustering method to assist the classification task.

- To the best of our knowledge, our model is the first to incorporate a clustering method into the gating mechanism for convolutional neural network, which can help to control related features with clusters.

- We conduct extensive experiments on five real-world datasets to verify the effectiveness of our model. The experiment results show that the proposed method outperforms state-of-the-art methods.

## 2 Related Work

In this section, we review the related work from the following two aspects: text classification and short text classification.

### 2.1 Text Classification

Traditional text classification methods generally rely on manual features, such as bag-of-words, short n-grams, POS tagging. Most recent studies design more complex features for specific applications. For example, Lazaridou et al. (2013) considered discourse connectives (such as "*but*", "*and*") in the Bayesian model for sentiment classification. Post and Bergsma (2013) used multiple explicit and implicit syntactic features (e.g., unigrams, bigrams, and grammar tree patterns) for text classification. Zhang et al. (2015a) integrated word embeddings learned by word2vec into support vector machine model.

Recently, deep learning methods have been proven to be effective in text classification. Kim (2014) proposed a convolutional neural network (CNN) architecture that utilized multiple parallel convolutional layers with varying filter window sizes and concatenated the selected important features into a dense softmax layer for sentence classification. Lai et al. (2015) applied a recurrent structure to learn contextual information of each word and employed a max-pooling layer to capture the important features in texts. Another state-of-the-art method is hierarchical attention networks for document classification (Yang et al., 2016). Based on documents' hierarchical structure, it performed attention mechanisms on word-level and sentence-level representations extracted by BiLSTMs.

Such methods have good performance for long texts, especially for documents or paragraphs, but they are inferior when directly applied for short text classification task. Short texts tend to span over a wide range of words, resulting in data sparsity and lack of enough contexts (Chen et al., 2011; Wang et al., 2017).

### 2.2 Short Text Classification

According to our review, there generally exist two strategies for short text classification.

The first strategy is to leverage an external knowledge base to expand the context of short texts. For example, Feng et al. (2013) calculated the correlation between each short text and domain knowledge for classification. Wang et al. (2014) leveraged a large-scale taxonomy knowledge base to learn the concepts of words and ranked the similarities between short texts and concepts. Wang et al. (2017) associated each short text with its relevant concepts in the knowledge base. They

combined the words and relevant concepts of the short text to generate its embedding. A high-quality knowledge base is vital for their performance, but its construction is time-consuming and labor-intensive, or even worse, it may be unavailable for some domains (Li et al., 2016).

The second strategy is to explore latent topics or clustering features for classification. For example, Chen et al. (2011) derived multi-granularity topics through latent Dirichlet allocation (LDA) as features for traditional classifiers. Ren et al. (2016) used LDA to extract topics and extended existing recursive autoencoder to effectively incorporate topic information. Ma et al. (2015) used Gaussian models to describe the distribution of words embeddings and classified new short texts using the Bayesian rule to get the posterior probability. Revanasiddappa and Harish (2018) developed a fuzzy c-means clustering method and built the match degree between cluster and categories. Such methods can reduce high dimensionality and terms' sparse distribution problems. But their pipeline architecture (i.e., using clustering or topic models to derive clusters or topics, and then integrating them into classifiers as features), might be hard to leverage the mutual dependency of clustering and classification methods.

## 3 Method

In this paper, we propose a joint model called cluster-gated convolutional neural network (CGCNN), coupling a soft clustering method and a gated CNN for classification. In this section, we mainly introduce the overall architecture of our model, and define the objective function for training.

### 3.1 Overall Architecture of the Model

Figure 1 presents the CGCNN structure, composing of five major components: (1) a word encoder layer based on BiLSTM to learn word representations in each short text, (2) a clustering layer that calculates words' distributions and performs a linear transformation to get cluster-dependent text representations, (3) a cluster-gated convolutional layer that integrates cluster centers into a gated CNN for further controlling cluster-related feature flows, (4) a max-pooling layer to select most important features and concatenate them as the final text features, and (5) a fully connected layer with softmax function for classification. We update all the parameters in these five components simultaneously, which is introduced in the next subsection.

**Word Encoder.** Suppose each short text has a maximum of $T$ words, and the $t$-th word can be

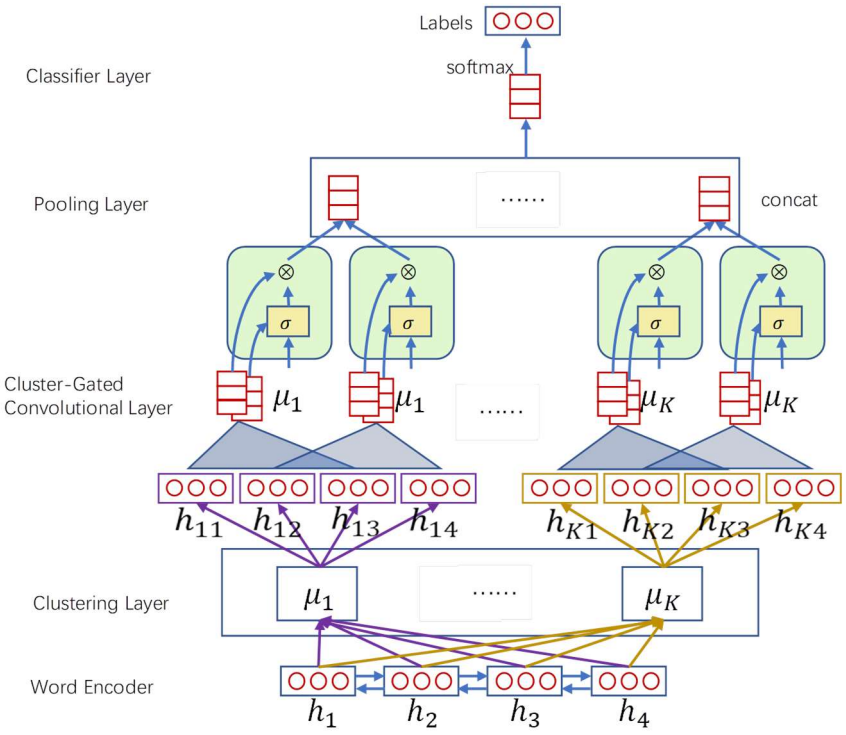



Figure 1: Cluster-gated convolutional neural network.

denoted as $w_t$, $t \in [1,T]$. We can embed the short text to vectors through an embedding matrix $W_e$. To capture the local context in text, we employ a BiLSTM to derive the forward representation $f_t$ and backward representation $b_t$. We concatenate them as word representation, i.e., $h_t = [f_t, b_t]$. Specifically, the input text is represented as a matrix $X = [h_1, h_2, \dots, h_T]$. In some cases with weak sequential text, we will directly use word embedding as the corresponding word representation $h_t$, i.e., $h_t = W_e w_t$. This will be further discussed in the experiment section.

**Clustering Layer.** Cluster centers contain semantic closeness of similar words, which is used to selectively control related word flows in the next layer. Here we employ a soft clustering method (Maaten and Hinton, 2008; Xie et al., 2016) to explore words' cluster centers. And then we build a projection function $f_\theta: (h_t, \mu_k) \rightarrow h_{kt}$ to get cluster-dependent text representations, where $\mu_k$ refers to the *k*-th cluster center, and $h_{kt}$ refers to the *t*-th representation dependent on the *k*-th cluster center. We set the number of clusters as *K*, i.e., $k \in [1,K]$. The soft clustering method uses the student's *t*-distribution as a kernel to calculate the similarity between word representation $h_t$ and cluster center $\mu_k$, as formula (1).

$$q_{t,k} = \frac{(1+ \| h_t - \mu_k \|^2)^{-1}}{\sum_{k'}(1+ \| h_t - \mu_{k'} \|^2)^{-1}} \quad (1)$$

where $q_{t,k}$ is the probability of *t*-th word belonging to *k*-th cluster. A higher value of $q_{t,k}$ indicates the word is more closed to the cluster.

With the help of the probability, we build a linear function to get the cluster-dependent text representations, as formula (2). It can reduce the role of words unrelated with the cluster, and ensures the sum of all cluster-dependent word representations at position *t* to the corresponding word representation $h_t$, as formula (3).

$$f_\theta : h_{k,t} = h_t q_{t,k} \quad (2)$$

$$\sum_k h_{k,t} = \sum_k h_t q_{t,k} = h_t \sum_k q_{t,k} = h_t \quad (3)$$

In this way, we can transfer the matrix of a short text to *K* cluster-dependent matrices, as formula (4).

$$X_k = [h_{k,1}, h_{k,2}, ..., h_{k,T}], k \in [1,K] \quad (4)$$

**Cluster-Gated Convolutional Layer.** The gating mechanism can control information flows in the network, which have been proven effective in LSTM and CNN (Dauphin et al., 2016). With the help of cluster centers, we would further explore related features with clusters in this layer. We employ a convolutional filter $W_k \in R^{D \times n}$ for mapping *n* words into a phrase-level feature, where *D* and *n* refer to the dimention of $h_{k,t}$ and the filter window size respectively. As shifting the filter across the *k*-th cluster-dependent text representation $X_k$, as formula (5), we can obtain a sequence of new features $C_k = [c_{k,1}, c_{k,2}, \dots, c_{k,L}]$. Here we use no-padding mode, i.e., *L=T-n+1*.

$$c_{k,i} = relu(h_{k,i:i+n} * W_k + b_k) \quad (5)$$

where $b_k$ is the term bias.

Based on gated linear units (GLU) (Dauphin et al., 2016), we use word representations and cluster center to together decide the information passed on, as formulas (6) and (7).

$$g_{k,i} = \sigma(h_{k,i:i+n} * U_k + V_k \mu_k + d_k) \quad (6)$$

$$s_{k,i} = c_{k,i} \otimes g_{k,i} \quad (7)$$

where $U_k \in R^{D \times n}$, $V_k \in R^D$, $d_k \in R$ are learned parameters. $\sigma$ is the sigmoid function, and $\otimes$ is the element-wise product between vectors. $g_{k,i}$ refers to the cluster-gated value, which is used to control the convolutional feature $c_{k,i}$. And $s_{k,i}$ is the final gated convolutional feature at position *t* for *k*-th cluster-dependent text representation.

**Pooling Layer**. In this layer, we apply a max-over-time pooling operation over each cluster-gated convolutional features to capture the maximum value as the feature for the corresponding cluster-dependent text representation, as formula (8). And then we concatenate all of them for the next classification layer, as formula (9).

$$s_k = \max\{s_{k,i}, i \in [1,T]\}, k \in [1,K] \quad (8)$$

$$s = s_1 \oplus s_2 \oplus \dots \oplus s_K \quad (9)$$

**Classifier Layer**. For each short text instance, we generate the high-level representations of the combination of multiple clusters' related information. To make full use of them, we use a fully connection with softmax function for prediction. The probability assigning a category label to this instance, can be calculated as formula (10).

$$p(\hat{y} = j) = soft\max(Ws + b) \quad (10)$$

where $j$ is the category label. To avoid over-fitting, we can also employ dropout in this layer.

### 3.2 Training

The entire CGCNN model integrates a clustering method into the gated-CNN for classification, which can be updated simultaneously in one framework. Hence, we combine their loss effects into one objective function as formula (11).

$$L = L_{CLF} + \lambda L_{CLU} \quad (11)$$

where $L_{CLF}$ is the cross-entropy loss of the classifier, and $L_{CLU}$ is the clustering loss with Kullback-Leibler divergence (KL divergence) minimization. $\lambda > 0$ is a tradeoff parameter controlling the degree of clustering loss. The classifier loss can be defined as formula (12).

$$L_{CLF} = -\sum_i \sum_j 1\{y_i = j\} \log P(\hat{y}_i = j) \quad (12)$$

where $i$ is the $i$-th sample instance, $y_i$ is the ground truth label, and $1\{*\}$ is the indicator function.

For the clustering loss, we use KL divergence between the distribution of soft labels $q_{t,k}$ and the auxiliary distribution $p_{t,k}$ as (Maaten and Hinton, 2008; Xie et al., 2016), as formula (13).

$$L_{CLU} = \sum_i \sum_t \sum_k p_{t,k} \log \frac{p_{t,k}}{q_{t,k}} \quad (13)$$

where $p_{t,k}$ is the target distribution, as formula (14).

$$p_{t,k} = \frac{{q_{t,k}}^2 / \sum_{t'} q_{t',k}}{\sum_{k'} ({q_{t,k'}}^2 / \sum_{t'} q_{t',k'})} \quad (14)$$

As (Xie et al., 2016), this target distribution is computed by first raising the second power of $q_{t,k}$ to its corresponding soft cluster frequencies $\sum_{t'} q_{t',k}$ and then performing normalization to prevent large clusters from distorting the hidden feature space. It can not only improve cluster purity, but also emphasize the data points assigned to clusters with high confidence.

## 4 Experiments

### 4.1 Datasets and Preprocessing

To illustrate the effectiveness of our model, we conduct experiments on five public datasets: AG News, Sogou News, Amazon Reviews, Yahoo! Answers, and Search Snippets. The first three datasets are adopted from (Zhang et al., 2015b). The last two datasets are from the Yahoo! Webscope program and (Phan et al., 2008) respectively. For each dataset, we use 80% of the data for training, 10% for validation, and the remaining 10% for test. To construct short texts, we only use titles or some partial information of the datasets.

**AG News and Sogou News.** These two original datasets include 127,600 samples from 4 categories and 510,000 samples from 5 categories respectively. Sogou News is a dataset in Chinese, and Zhang et al. (2015b) combined pypinyin package and a Chinese segmentation tool to produce Pinyin – Roman spelling in Chinese. For both of them, each sample contains both title and content of news. To test for short texts, we remove contents and only use the titles in our experiment.

**Amazon Reviews.** The full dataset contains 3.65 million samples from one-to-five rating labels. In order to test for short texts, we remove the review contents and only use the review titles in our experiment.

**Yahoo! Answers.** This corpus includes 4,483,032 question titles, question contexts and their answers. We use 10 largest classes to construct a topic classification task. We randomly choose 50,000 samples for each class. Here we only use the question titles for classification.

**Search Snippets**. This dataset, released by Google search engine, includes 12,340 samples with predefined 8 categories by (Phan et al., 2008).

Note that we filter out punctuation and use Natural Language Toolkit (NLTK) for stemming. We do not remove stopwords since some of them may carry classification information, especially for users' reviews. The details of each dataset are listed in Table 1.

| Datasets | Size | Classes | Avg. Len |
|---|---|---|---|
| AG News | 127,600 | 4 | 7.0 |
| Sogou News | 510,000 | 5 | 15.4 |
| Amazon Review | 3,650,000 | 5 | 4.6 |
| Yahoo! Answers | 500,000 | 10 | 11.2 |
| Search Snippets | 12,340 | 8 | 17.9 |

Table 1: A summary of datasets.

### 4.2 Implementation Detail

The model hyper-parameters are tuned based on the AG News dataset. We also conduct experiments with the model directly using word embeddings instead of BiLSTM, representing as CGCNN*. We firstly set the dimension of word embeddings to 300, and pre-train word embeddings on each dataset with word2vec. The dimensions of all hidden vectors are set to 200. For the clustering method, we set the number of clusters to the number of ground-truth categories, and randomly initialize cluster center vectors. We set $\lambda$=0.5 and $\lambda$=0.6 for CGCNN* and CGCNN respectively to control the effects of clustering method. To avoid model over-fitting, we use dropout with rate of 0.2. We train the parameters by using Adam method with a learning rate of 0.001, and set the batch size to 64. The filter sizes of all convolution layers are set to 3 in these two methods.

### 4.3 Baselines and Experimental Settings

In this paper, we choose the following baseline algorithms for comparison:

**CNN (Kim, 2014).** It builds a multi-channel convolutional architecture with varying filter window sizes, and concatenates the important features extracted by a max-over-time pooling operation.

**CNNM.** To further illustrate the effectiveness of our model, we develop the multi-channel convolutional architecture (Kim, 2014) with multiple fixed size filters. As our model hyperparameters, the number of filters is equal to the number of ground-truth categories, and all their sizes are set to 3.

**RCNN (Lai et al., 2015).** It develops a recurrent convolutional structure. It employs a bi-directional recurrent structure to capture word context embeddings and uses a max-pooling layer to select the important features.

**CNN-LSTM (Zhou et al., 2015).** This method uses a multi-channel convolutional layer to extract higher-level phrase features, and employs a BiLSTM to capture their sequences for classification.

**AttBiLSTM (Lin et al., 2017).** It uses a BiLSTM to explore the sequences of texts, and develops a self-attention mechanism to get sentence-level representations.

For the multiple-channel convolutional architecture of CNN and CNN-LSTM, the filter sizes are 3, 4 and 5, as Kim (2014)'s default settings. For the hidden vectors of BiLSTM in these methods, we also set their dimensions to 200.

### 4.4 Results

We use accuracy as the evaluation metric, and Table 2 reports the different algorithms' performance on the five real-world datasets. We highlight the highest value in each column. As we can see, either CGCNN or CGCNN* has the best performance on the datasets. The CNN-LSTM outperforms the other baseline methods on AG News, Amazon Review and Yahoo! Answers datasets, while CNN and CNNM have the best performance on Sogou News and Search Snippets respectively. As compared with CNN-LSTM, CGCNN has about 1.5% performance improvements on Amazon Review and Yahoo! Answers, and 0.49% on AG News dataset. CGCNN* can achieve 0.6%~0.8% performance improvements over the second best baseline method on Sogou News and Search Snippets datasets. The AttBiLSTM method has poor performance. We suspect that lack of enough context might cause the failure of the self-attention mechanism.

| | AG News | Sogou News | Amazon Review | Yahoo! Answers | Search Snippets |
|---|---|---|---|---|---|
| CNN | 87.62% | 90.47% | 46.71% | 61.64% | 93.60% |
| CNNM | 87.89% | 90.17% | 46.65% | 61.59% | 93.84% |
| CNN-LSTM | 88.12% | 89.67% | 47.80% | 62.61% | 93.11% |
| RCNN | 87.69% | 88.43% | 46.95% | 61.90% | 93.68% |
| AttBiLSTM | 87.63% | 87.63% | 46.96% | 61.92% | 91.09% |
| CGCNN* | 88.24% | **91.02%** | 47.17% | 63.01% | **94.57%** |
| CGCNN | **88.55%** | 90.95% | **48.65%** | **63.55%** | 92.30% |

Note: CGCNN* represents that our model directly inputs word embeddings into the clustering layer.

Table 2: Accuracy comparison on different datasets.

The CNNM method using category number of convolutional filters, has similar performance with the CNN method using three convolutional filters, which illustrates increasing number of convolutional filters might have no active impact on performance. Differently, our CGCNN* method using category number of clusters for gated CNN, achieves better accuracies than both of them. It shows that our proposed architecture, integrating a clustering-gated mechanism into CNN, can significantly improve the performance in short text classification.

The CNN-LSTM method uses CNN and BiLSTM to capture phrase features and their sequences, outperforms CNN and CNNM on AG News, Amazon Reviews and Yahoo! Answers datasets, while it has poorer performance on Sogou News and Search Snippets datasets. Such cases also exist in the comparison between CGCNN and CGCNN* methods. We analyze the datasets, and suspect that weak sequential relationship in texts may result in the decreasing performances on Sogou News and Search Snippets. The original Sogou News was transferred from Chinese characters to Pinyin format (Zhang et al., 2015b). It might cause a homophone problem. For example, word "与(and)" and word "雨(rain)" have the same pronunciation but different meanings in Chinese. It breaks sequential patterns in texts. For Search Snippets, each sample is consisted of multiple keywords, and there are no obvious sequences among them. For example, a sample likes "… calorie count calories item ...", containing weak sequential semantics.

### 4.5 Clustering Analysis

To further study the impact of clustering method, we conduct additional experiments on AG News dataset by varying the tradeoff parameter $\lambda$ and the cluster number $K$, and assess the sensitivity of our model.

Figure 2 reports the change of performance with increasing values of tradeoff parameter $\lambda$ from 0.1 to 0.9 while keeping the cluster number $K$ constant (as the category number). We can observe that CGCNN* and CGCNN reach the best performances when $\lambda = 0.5$ and $\lambda = 0.6$ respectively. When tradeoff parameter $\lambda$ varies from 0.1 to the values of their best performances, their performances generally show an increasing trend, which implies the clustering effect can benefit for understanding short texts. When tradeoff parameter $\lambda$ increases from the optimal values to 0.9, the performances of these two methods generally have a slight decrease, which shows excessive clustering might have a bad influence on short text classification.

Figure 3 reports the results of adjusting the number of clusters ($K$) in CGCNN* and CGCNN when we set $\lambda$ to the optimal values (i.e., $\lambda = 0.5$ and $\lambda = 0.6$ respectively). For CGCNN method, we can observe that it reaches the best performance when the cluster number equals to the category number (i.e., $K = 4$). No matter the cluster number increases or decreases, its performance would have a decrease tendency. While the CGCNN* method's performance generally show an increasing trend, which relatively stabilizes when $K$ reaches category number (although its performance has a slight decrease at $K = 5$). That is the reason that we set the cluster number to the category number.

### 4.6 Case Study

In this section, we take several concrete samples from AG News dataset to illustrate how the

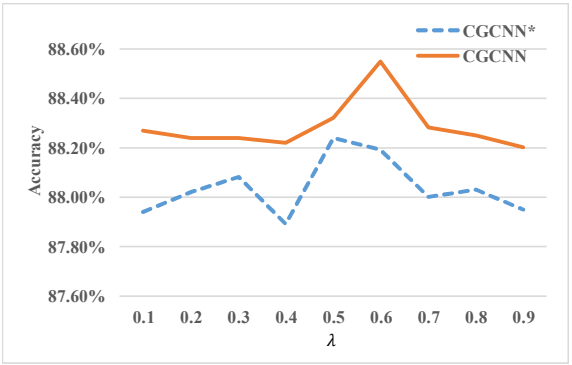



Figure 2: Performance with tradeoff parameter $\lambda$ on AG News.

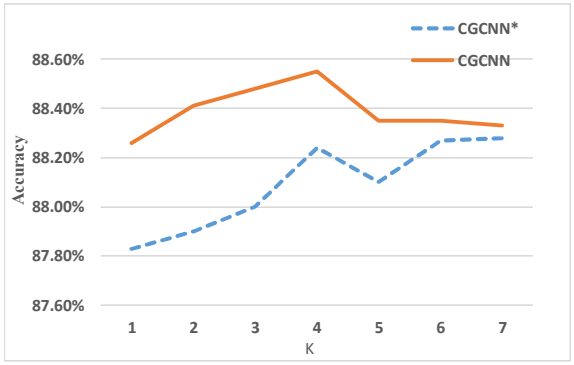
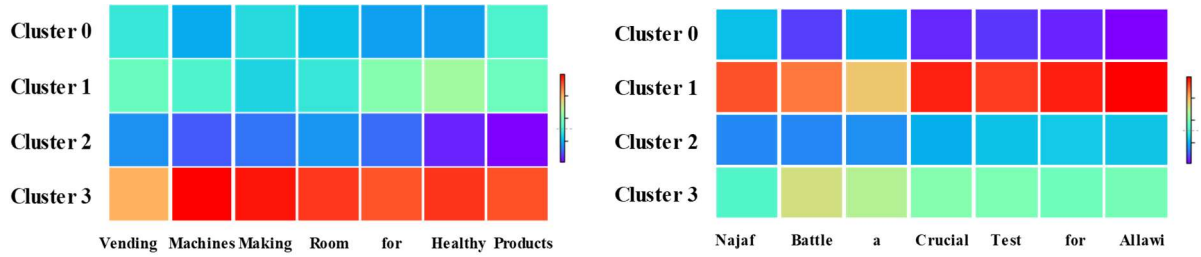
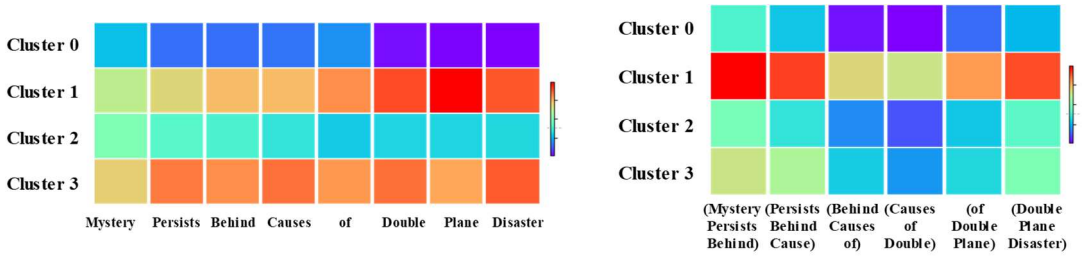



Figure 3: Performance with cluster number K on AG News.




(a) An instance with category "Sci/Tech" (b) An instance with category "World"

Figure 4: The similarities between words and clusters in a short text.




(a) The similarities between words and clusters in an instance with category "World"

(b) The outputs of the corresponding cluster-gated convolutional features

Figure 5: The role of the cluster-gated layer.

proposed method works. Here we use CGCNN method, because of strong sequential relationship in this dataset.

In the clustering layer, we leverage a soft clustering method to explore words' cluster centers, and build a linear function to project the representation of a short text to $K$ cluster-dependent representations. Here we calculate the similarity between words and cluster centers, and normalize the values of each word belonging to clusters. Figure 4 (a) and (b) show two different instances from categories "Sci/Tech" and "World" respectively. We can observe the linear projection can strengthen the words' representations dependent on some cluster, and weaken them on others. These two figures have different distributions on the same word "for", which is due to different contexts explored by BiLSTM.

There might exist some instances closely related with two or more clusters, as figure 5 (a). To further control the information flows, we leverage cluster centers and phrase-level features for the gated mechanism. We use Xue and Li (2018)'s method to visualize the gated mechanism: summing the representation of each phrase-level feature and normalizing them according to clusters. Figure 5 (a) shows the similarities between words and clusters in an instance with category "World", while figure 5 (b) shows the corresponding cluster-gated convolutional features. We can observe that cluster-gated layer can further strengthen the corresponding cluster-dependent representation, and weaken others.

## 5 Conclusion

In this paper, we propose a joint model that couples clustering and classification methods. It employs a BiLSTM to learn word representations for local contexts in short texts. We take a soft clustering method to calculate the probability of each word assigning to each cluster, which can derive the semantic relation of word representations and the global corpus. We also perform a linear transformation to explore cluster-dependent text representations. Moreover, we develop a cluster-gated CNN by integrating cluster centers into GLU, which can select cluster-related features for classification. Experiments on five real-world datasets show that our model does better than the state-of-the-art methods for short text classification task.

In the future work, we will further analyze the mutual effects of document-level clustering and classification methods for long text, and attempt to develop more effective joint model for text classification. Moreover, we will study some other

mechanisms (e.g., highway units, attention mechanism) to further improve the performance.

## 6 Acknowledge

We thank Junjie Li, Jianwei Guo, Yiqiang Shi, and the anonymous reviewers for the constructive suggestions on various aspects of this work.

## References

Chen, M., X. Jin and D. Shen. 2011. Short text classification improved by learning multi-granularity topics. Proceedings of the 22th International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, AAAI Press, pages 1776-1781.

Dauphin, Y. N., A. Fan, M. Auli and D. Grangier. 2016. Language modeling with gated convolutional networks. Proceedings of the 34th International Conference on Machine Learning. 70**,** pages 933-941.

Feng, X., Y. Shen, C. Liu, W. Liang and S. Zhang. 2013. Chinese Short Text Classification Based on Domain Knowledge. International Joint Conference on Natural Language Processing, Nagoya, Japan, pages 859–863.

Gehring, J., M. Auli, D. Grangier, D. Yarats and Y. Dauphin. 2017. Convolutional Sequence to Sequence Learning. Proceedings of the 34th International Conference on Machine Learning (ICML), pages 1243-1252.

Kim, Y. 2014. Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) University of Waterloo, pages 1746-1751.

Lai, S., L. Xu, K. Liu and J. Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification. Proceedings of the 29-th AAAI Conference on Artificial Intelligence, pages 2267-2273.

Lazaridou, A., I. Titov and C. Sporleder. 2013. A Bayesian Model for Joint Unsupervised Induction of Sentiment, Aspect and Discourse Representations. Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, Association for Computational Linguistics, pages 1630-1639.

Li, C., H. Wang, Z. Zhang, A. Sun and Z. Ma. 2016. Topic Modeling for Short Texts with Auxiliary Word Embeddings. Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval. Pisa, Italy, ACM**,** pages 165-174.

Lin, Z., M. Feng, C. N. dos Santos, M. Yu, B. Xiang, B. Zhou and Y. Bengio. 2017. A Structured Self-Attentive Sentence Embedding. International Conference on Learning Representations 2017 (ICLR), pages.

Luo, G., X. Huang, C. Y. Lin and Z. Nie. 2015. Joint Named Entity Recognition and Disambiguation. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, ACL, pages 879-888.

Ma, C., W. Xu, P. Li and Y. Yan. 2015. Distributional Representations of Words for Short Text Classification. Proceeding of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics. Denver, Colorado**,** pages 33-38.

Maaten, L. v. d. and G. Hinton. 2008. Visualizing data using t-SNE. Journal of machine learning research. 9(Nov): 2579-2605.

Phan, X.-H., L.-M. Nguyen and S. Horiguchi. 2008. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. Proceedings of the 17th International Conference on World Wide Web. Beijing, China, ACM**,** pages 91-100.

Post, M. and S. Bergsma. 2013. Explicit and Implicit Syntactic Features for Text Classification. Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria,, Association for Computational Linguistics, pages 866-872.

Ren, Y., R. Wang and D. Ji. 2016. A topic-enhanced word embedding for Twitter sentiment classification. Information Sciences. 369: 188-198.

Revanasiddappa, M. and B. Harish. 2018. A New Feature Selection Method based on Intuitionistic Fuzzy Entropy to Categorize Text Documents. International Journal of Interactive Multimedia & Artificial Intelligence. 5(3): 106-117.

Schmitt, M., S. Steinheber, K. Schreiber and B. Roth. 2018. Joint Aspect and Polarity Classification for Aspect-based Sentiment Analysis with End-to-End Neural Networks. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, pages 1109–1114.

Shao, Y., C. Hardmeier, J. Tiedemann and J. Nivre. 2017. Character-based Joint Segmentation and POS Tagging for Chinese using Bidirectional RNN-CRF. Proceedings of the The 8th International Joint Conference on Natural Language Processing, Taipei, Taiwan, Asian Federation of Natural Language Processing, pages 173-183.

Wang, F., Z. Wang, Z. Li and J.-R. Wen. 2014. Concept-based Short Text Classification and Ranking. Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, Shanghai, China, pages 1069-1078.

Wang, J., Z. Wang, D. Zhang and J. Yan. 2017. Combining knowledge with deep convolutional neural networks for short text classification. Proceedings of the 26th International Joint Conference on Artificial Intelligence. Melbourne, Australia, AAAI Press, pages 2915-2921.

Xie, J., R. Girshick and A. Farhadi. 2016. Unsupervised deep embedding for clustering analysis. Proceedings of the 33rd International Conference on Machine Learning, pages 478-487.

Xue, W. and T. Li. 2018. Aspect Based Sentiment Analysis with Gated Convolutional Networks. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics Association for Computational Linguistics, pages 2514-2523.

Yang, Z., D. Yang, C. Dyer, X. He, A. Smola and E. Hovy. 2016. Hierarchical attention networks for document classification. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, California, pages 1480-1489.

Zhang, D., H. Xu, Z. Su and Y. Xu. 2015a. Chinese comments sentiment classification based on word2vec and SVMperf. Expert Systems with Applications. 42(4): 1857-1863.

Zhang, X., J. Zhao and Y. LeCun. 2015b. Character-level convolutional networks for text classification. Advances in neural information processing systems, pages 649-657.

Zhou, C., C. Sun, Z. Liu and F. C. M. Lau. 2015. A C-LSTM Neural Network for Text Classification. Computing Research Repository. arXiv:1511.08630.