Classification Over Clustering: Augmenting Text Representation with Clusters Helps!

Xiaoye Tan[1], Rui Yan[1,2⋆], Chongyang Tao[2], and Mingrui Wu[3]

[1] Center for Data Science, Peking University, Beijing, China
[2] Institute of Computer Science and Technology, Peking University, Beijing, China
{txye,ruiyan,chongyangtao}@pku.edu.cn
[3] Alibaba Group, Seattle
wu.mingrui@yahoo.com

Abstract. Considering that words with different characteristic in the text have different importance for classification, grouping them together separately can strengthen the semantic expression of each part. Thus we propose a new text representation scheme by clustering words according to their latent semantics and composing them together to get a set of cluster vectors, which are then concatenated as the final text representation. Evaluation on five classification benchmarks proves the effectiveness of our method. We further conduct visualization analysis showing statistical clustering results and verifying the validity of our motivation.

Keywords: classification · semantic · clustering · visualization.

1 Introduction

Text classification is an important task in natural language processing with many applications. Suitable text encoding scheme will benefit it a lot. Early studies use discrete and context-insensitive approaches such as TF-IDF [15] and weighted bag-of-words [4]. Along with the prosperity of research on the distributed representation of words [10, 2], linear combination of word embeddings in a sentence [14] is widely used for classification. However, this method captures the information from individual words only. Many studies that make use of contextual information commonly adopt neural networks as an encoder to capture the semantic information of the text, including recurrent neural networks (RNN) [16], tree-structure recursive networks [17] and convolutional neural networks (CNN) [7, 6, 3].

However, the aforementioned methods lack consideration on the semantic segmentation of text. It is intuitive that the words in text play different roles thus contribute to different groups of semantic functionality. Words like "the", "an", "of" act as connection between the syntax structures of the text and contribute little for classification. Other words like "internet", "linux" or "wireless" indicating a specific domain (science) tend to have a huge impact on the classification performance. Some recent studies attempt to capture different types of word importance via hierarchical attention [21] or multi-head self-attention [8]. However, these attention mechanisms still only consider

⋆ Corresponding author. (Email: ruiyan@pku.edu.cn)

the relative importance of separate words, which does not group words with similar semantics together to strengthen the semantic expression of the whole text.

As classification is a word sensitive task, people tend to integrate all related words when making decisions about the category of a sentence. Therefore, in this paper, we propose to augment text representation from a higher level: cluster level. Concretely, we divide the words in a text into different latent semantic clusters and get cluster representations by combining contextual embeddings of the words together according to their cluster probability distribution. The cluster representations are then concatenated as the final representation of the text. We further introduce two regularization terms to better guide the clustering process. Considering that not all semantic clusters contain useful information for classification, we design a gating mechanism to dynamically control their contributions for classification.

Experiments are conducted on five standard benchmark datasets of text classification. Quantitative results show that our method outperforms or is at least on a par with the state-of-the-art methods. We also perform visualized and statistical analysis on the intermediate word clustering results which proves the effectiveness of the clustering process.

In summary, the contributions of this paper include:

- We propose an intuitive architecture for text classification with a novel semantic clustering process for better capturing distant topical information in text representation. The semantic clusters in our framework are automatically calculated on-the-fly instead of being fitted in advance.
- Due to the probabilistic nature of soft semantic clustering, we introduce two regularization schemes to better guide the behaviors of our model.
- We conduct extensive experiments to evaluate our proposed method on five text classification benchmarks. Results show that our model could obtain competitive performance compared with state-of-the-art approaches.
- We provide statistical and visualization analysis on cluster distribution captured by our learned model further corroborating our motivation.

2 Model

We propose a latent semantic clustering representation (LSCR) framework as shown in Figure 1 consisting of four parts: 1) the word representation layer at the bottom converts words into vector representations (embeddings); 2) the encoding layer transforms a series of word embeddings to their corresponding hidden vector representations; 3) the semantics clustering layer attributes all the words into different clusters and composes them together respectively so as to get a set of cluster representations; 4) the aggregation layer combines those cluster vectors into a single vector as the final representation of the text.

Formally, given a text consisting of n words $(w_1, w_2, \cdots, w_n)$, the word representation layer converts them to their corresponding word embeddings, represented as $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n)$. Then in encoding layer, we employ a bi-directional LSTM [13] as the encoder to aggregate information along the word sequence getting final states

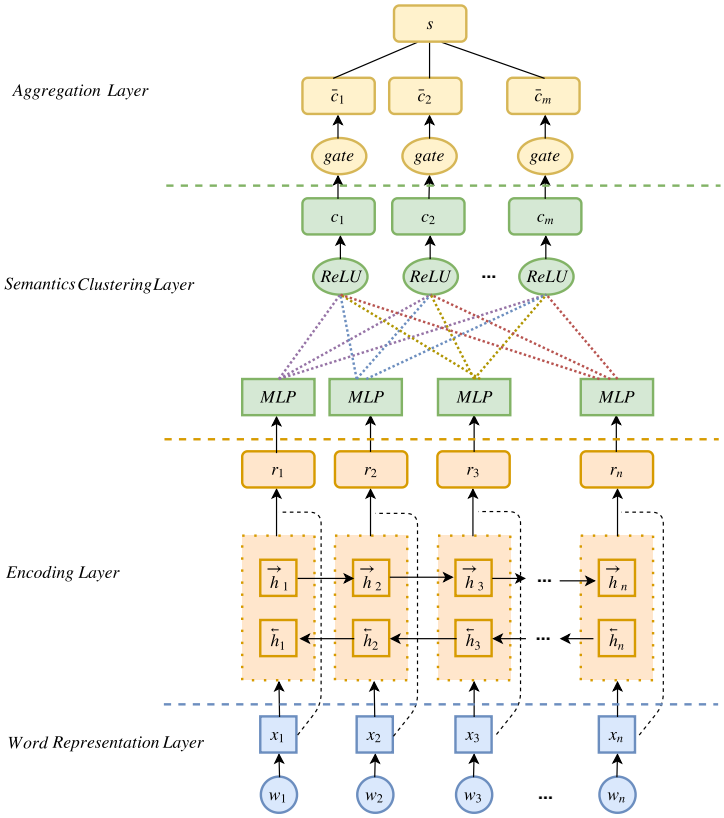


Fig. 1. The framework of our model.

$\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}}_t; \overleftarrow{\boldsymbol{h}}_t]$. We concatenate the hidden state from the encoder with the initial word embedding to enrich the representations. The output of the t-th word from this layer takes the form: $\boldsymbol{r}_t = [\boldsymbol{x}_t; \boldsymbol{h}_t]$.

The final representation of words generated from this layer is defined as $\boldsymbol{R} = (\boldsymbol{r}_1, \boldsymbol{r}_2, \cdots, \boldsymbol{r}_n)$. In the semantics clustering layer, suppose that words from the text could be assigned to m semantic clusters, where m is a tunable hyperparameter. For each word, we employ a MLP to determine the probabilities of dividing it into every cluster, defined as:

$$\boldsymbol{A} = f_1(\boldsymbol{W}_2 \cdot f_2(\boldsymbol{W}_1 \cdot \boldsymbol{R} + \boldsymbol{b}_1) + \boldsymbol{b}_2) \tag{1}$$

We use the softmax function for f_1, and ReLU for f_2. Concretely, $A_{i,j}$ indicates the probability of the j-th word being clustered into the i-th cluster. For each word w_j of the text, $\sum_{i=1}^{m} A_{i,j} = 1$. After getting the probabilities, the vector representation of the i-th cluster is given by the weighted sum of the contextualized word representations ($\boldsymbol{R}$) in the text, followed by a nonlinear transformation. The process is formulated as:

$$\boldsymbol{C} = \text{ReLU}(\boldsymbol{W}_s(\boldsymbol{A} \cdot \boldsymbol{R}) + \boldsymbol{b}_s) \tag{2}$$

The i-th row of $\boldsymbol{C}$, $\boldsymbol{c}_i$, refers to the vector representation of the i-th cluster. Considering that not all clusters are helpful for text classification. There may exist redundant clusters that contain little or irrelevant information for the tasks. Therefore, in the aggregation layer, we add a gating mechanism on the cluster vector to control the information flow. Concretely, the gate takes a cluster vector $\boldsymbol{c}_i$ as input and output a gate vector $\boldsymbol{g}_i$:

$$\boldsymbol{g}_i = \sigma(\boldsymbol{W}_g \boldsymbol{c}_i + \boldsymbol{b}_g) \tag{3}$$

We do the same operation on other cluster vectors as well, leading to a series of gate vectors $\boldsymbol{G} = (\boldsymbol{g}_1, \boldsymbol{g}_2, \cdots, \boldsymbol{g}_m)$ from the cluster vectors. Then the gated cluster vectors $\bar{\boldsymbol{C}} = (\bar{\boldsymbol{c}}_1, \bar{\boldsymbol{c}}_2, \cdots, \bar{\boldsymbol{c}}_m)$ are calculated as:

$$\bar{\boldsymbol{C}} = \boldsymbol{G} \odot \boldsymbol{C} \tag{4}$$

At last, we concatenate vector representations of all the semantic clusters to form the text representation:$\boldsymbol{s} = [\bar{\boldsymbol{c}}_1, \bar{\boldsymbol{c}}_2, \cdots, \bar{\boldsymbol{c}}_m]$. For classification, the text representation $\boldsymbol{s}$ is followed by a simple classifier which includes a fully connected hidden layer and a softmax output layer to get the predicted class distribution $\boldsymbol{y}$. The basic loss function is the cross-entropy loss $\mathcal{L}$ between the ground truth distribution and the prediction.

2.1 Regularization Terms

Due to the probabilistic nature of our soft semantic clustering scheme, it is natural to integrate probabilistic prior knowledge to control and regularize the model learning process. We consider two regularization terms in word level and class level. In the semantics clustering layer, we get $\boldsymbol{a}_i = (a_{i1}, a_{i2}, \cdots, a_{im})$ indicating the probability distribution of the i-th word to each cluster. The *word-level entropy regularization* term is defined as:

$$\mathcal{L}_{word} = -\sum_{t=1}^{N}\sum_{k=1}^{m} a_{tk} \log(a_{tk}) \tag{5}$$

We expect the probability distribution for a specific word over all the clusters is sparse, which means a word would be attributed to only one or few clusters with greater probability instead of being evenly distributed to all clusters. Thus our optimization goal is to minimize the word-level entropy.

Another class-level regularization term is specifically designed for text classification. Suppose there is a vector $\boldsymbol{v}_{c_i}$ indicating the i-th class' probability distribution over m clusters, which is calculated by averaging the cluster probability distribution of the text belonging to i-th class within a mini-batch during training. We take the average of the word's cluster probability distribution $\boldsymbol{a}$ in the text as the text-level cluster probability distribution $\boldsymbol{v}_s = \frac{1}{N_w}\sum_{i=1}^{N_w} \boldsymbol{a}_i$, where N_w is the number of words in the text. Thus the i-th class' cluster probability distribution is:

$$\boldsymbol{v}_{c_i} = \frac{1}{N_{c_i}} \sum_{k=1}^{N_{c_i}} \boldsymbol{v}_{s_k} \tag{6}$$

Table 1. Data statistic on the five benchmarks.

Dataset	# Classes	# Training	# Testing	# AvWords	# MaxWords
AGNews	4	120K	7.6K	45	208
Yah.A.	10	1.4M	60K	104	146
Yelp P.	2	560K	38K	153	301
Yelp F.	5	650K	50K	155	501
DBPedia	14	560K	70K	55	151

where N_{c_i} is the number of samples belonging to i-th class in a mini-batch, $\boldsymbol{v}_{s_k}$ indicates the k-th text-level cluster probability distribution. We hope that different cluster can capture different category related semantics. Thus the distribution between every two classes needs to be different. We take an intuitive and practical method that we expect the peak value of different class-level distribution exists in different cluster. To implement this, we add the maximum value of all the class-level distributions and expect the summation greater. The *class-level regularization* term is defined as:

$$\mathcal{L}_{class} = \sum_{i=1}^{m} \max_{j=1:N_C}(\boldsymbol{v}_{c_j}^i) \tag{7}$$

where N_C is the number of category and $\boldsymbol{v}_{c_j}^i$ means the i-th dimension (corresponding to i-th cluster) in j-th class probability distribution vector. The larger the summation, the distributions between classes tend to be more different. The final objective function for classification is defined as:

$$\mathcal{L}_{total} = \frac{1}{N}\sum_{i=1}^{N}(\mathcal{L} + \lambda_1 \mathcal{L}_{word}) - \lambda_2 \mathcal{L}_{class} \tag{8}$$

where N is the number of samples in a mini-batch. The training objective is to minimize $\mathcal{L}_{total}$.

3 Experiment

3.1 Datasets

To evaluate the effectiveness of our proposed model, we conduct experiments on the text classification task. We test our model on five standard benchmark datasets (**AGNews**, **DBPedia**, **Yahoo! Answers**, **Yelp P.**, **Yelp F.**) including topic classification, sentiment classification and ontology classification as in [23]. **AGNews** is a topic classification dataset that contains 4 categories: world, business, sports and science. **DBPedia** ontology dataset is constructed by choosing 14 non-overlapping classes from DBPedia 2014. The fields used for this dataset contain the title and abstract of each Wikipedia article. **Yahoo! Answers** is a 10-categories topic classification dataset obtained through the Yahoo! Webscope program. The fields include question title, question content and best answer. **Yelp P.** and **Yelp F.** are Yelp reviews obtained from the Yelp Dataset Challenge in 2015. Yelp p. predicts a polarity label by considering stars 1 and 2 as negative, 3 and 4 as positive. Yelp F. predicts the full number of stars from 1 to 5. We use the preprocessed datasets published by [18] and the summary statistics of the data are shown in Table 1.

Table 2. Accuracy of all the models on the five datasets. The result marked with ⋄ is re-printed from [19].

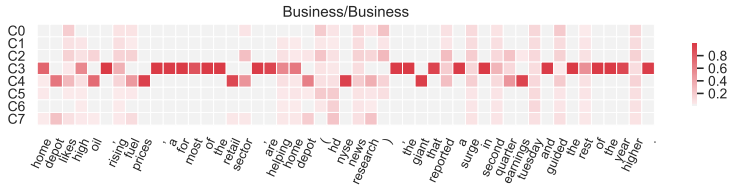
Method	AGNews	Yah.A.	DBPedia	Yelp P.	Yelp F.
n-gram TF-IDF [23]	92.4	68.5	98.7	95.4	54.8
BoW [23]	88.8	68.9	96.6	92.2	58.0
FastText [5]	92.5	72.3	98.6	95.7	63.9
SWEM [14]	92.2	73.5	98.4	93.8	61.1
DeepCNN(29 Layer)[3]	91.3	73.4	98.7	95.7	64.3
Small word CNN [23]	89.1	70.0	98.2	94.5	58.6
Large word CNN [23]	91.5	70.9	98.3	95.1	59.5
Dynamic-Pool$^{\diamond}$ [6]	91.3	-	98.6	95.7	63.0
Densely Connected CNN [19]	**93.6***	-	**99.2**	96.5	66.0
LSTM [23]	86.1	70.8	98.6	94.7	58.2
char-CRNN [20]	91.4	71.7	98.6	94.5	61.8
ID-LSTM [22]	92.2	-	-	-	-
HS-LSTM [22]	92.5	-	-	-	-
Self-Attentive$^{\diamond}$ [8]	91.5	-	98.3	94.9	63.4
LEAM [18]	92.5	77.4	99.0	95.3	64.1
W.C.region.emb [12]	92.8	73.7	98.9	96.4	64.9
LSCR (our work)	**93.6***	**78.2***	99.1	**96.6***	**67.7***

3.2 Compared Models

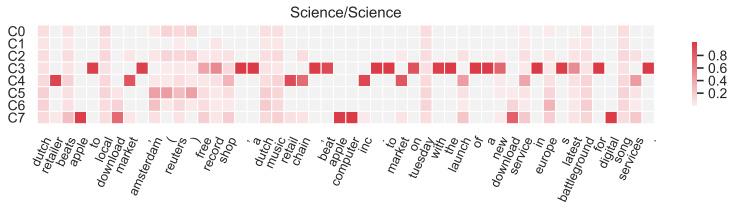
We compare our models with different types of baseline models: traditional feature based models i.e. n-gram TF-IDF and bag-of-words (BoW) [23]; word embedding based models such as FastText [5] and SWEM [14]; RNNs like LSTM [23]; reinforcement learning based models including ID-LSTM and HS-LSTM [22]; CNNs consisting of DeepCNN [3], small/large word CNN [23], CNN with dynamic pooling [6] and densely connected CNN with multi-scale feature attention [19]; CNN combined with RNN [20]; self-attentive based model [8]; other models specifically designed for classification [18, 12].

3.3 Implementation Details

For word representation, we use the pre-trained 300-dimensional GloVe word embeddings [11]. They are also updated with other parameters during training. We split 10% samples from the training set as the validation set and tuned the hyper parameters on validation set. The input texts are padded to the maximum length appeared in the training set. In the encoding layer, the hidden state of the bi-LSTM is set to 300 dimensions for each direction. The MLP hidden units are 800 for semantic clustering. The dimension of the clustering vectors is set to 600. The cluster number is set to 8 on AGNews and 10 on other datasets. The MLP used for classification has 1000 hidden units. The coefficients of the regularization terms are set to 0.001. We adopt Adam as the optimizer with a learning rate of 0.0005. The batch size is 64. Our models are implemented with Tensorflow [1] and trained on one NVIDIA 1080Ti GPU. For all datasets, training converges within 4 epochs. We will release our codes later.



(a) A sentence about business.



(b) A sentence about science.

Fig. 2. The visualization on AGNews about topic classification. The heat map shows the distribution of words among different clusters in the text. The X-axis are words in text. The Y-axis represents the clusters. The title of each figure indicates the predicted class / the ground truth.

3.4 Evaluation Results

We compared our method to several state-of-the-art baselines with respect to accuracy, the same evaluation protocol with [23] who released these datasets. We present the experimental results in Table 2. The overall performance is competitive. It can be seen from the table that our method improves the best performance by 1.0%, 0.1% and 2.5% on Yah A., Yelp P. and Yelp F respectively and is comparable on AGNews.

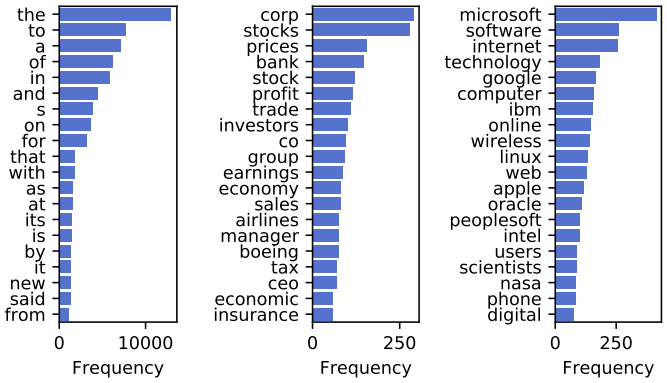
Compared with the baselines, our results exceed the traditional feature-based models (n-gram TF-IDF, BoW), the word embedding based representation models (FastText, SWEM) and LEAM by a large margin. Furthermore, we gain great improvement over the LSTM-based models. Though our model and the self-attentive model all aim at getting multiple vectors indicating different semantic information, as our model gather the words into clusters to enrich the representations, we achieve a better performance over theirs. Compared with the deep CNNs, our shallow model with a relatively simpler structure outperforms them as well. The models proposed by [19] and [12] aim at capturing features from different regions. Our results are on par with [19] and win over [12].

3.5 Ablation Study

In this section, we randomly select two datasets (Yelp P. and Yah.A.) and conduct ablation studies on them. We aim at analyzing the effect of gating mechanism and two kinds of regularization terms: word-level and class-level. The results without each part separately are shown in Table 3. “Full Model” denotes the whole model with nothing

Table 3. The ablation results on Yelp P. and Yah.A.

Model	Yelp P.	Yah.A.
Full Model	96.6	78.2
w/o Gating Mechanism	96.3	77.9
w/o Word Level	96.4	78.0
w/o Class Level	96.4	78.0



(a) Cluster 3. (b) Cluster 4. (c) Cluster 7.

Fig. 3. The statistical clustering results on AGNews in Cluster 3, 4 and 7.

absent. Models trained with the gating mechanism and the regularization terms outperform their counterpart trained without, which demonstrates the effect of each part.

3.6 Discussions

Analysis on Clustering Results In this section, we take out the intermediate clustering results calculated when we test on AGNews which is a 4-category topic classification dataset including science, business, sports, and world.

Firstly, we visualize the distribution of words in clusters using heat map as shown in Figure 2. Each column means the probabilities of a word attributed to the clusters which sum up to 1. We can see that most of the meaningless words, high frequency words and punctuation are divided into cluster 3, like *"a"*, *"for"*, *"that"*, etc. The text in Figure 2(a) belongs to business category and the model predicts it right. We can find that the words about business are in cluster 4 such as *"depot"*, *"oil"*, *"prices"*. Likewise, from Figure 2(b) we find words about science are assigned to cluster 7 like *"apple"*, *"download"*, *"computer"*. In Figure 2(b), the text contains words about science and business, which are divided into their corresponding clusters separately. The words like *"beats"*, *"second"*, *"local"*, which are domain independent nouns or adjectives or verbs have a more average probability distribution among all clusters.

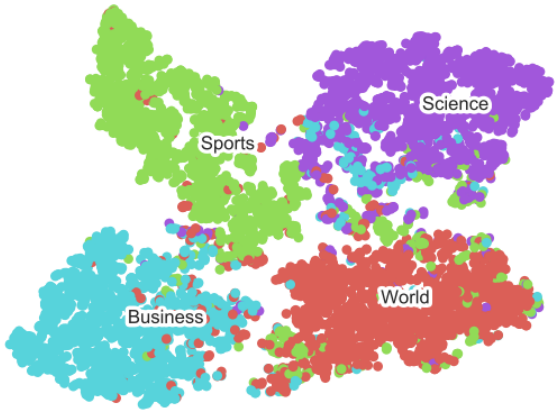


Fig. 4. t-SNE plot of the intermediate clustering probability distribution vector of text on AGNews test set.

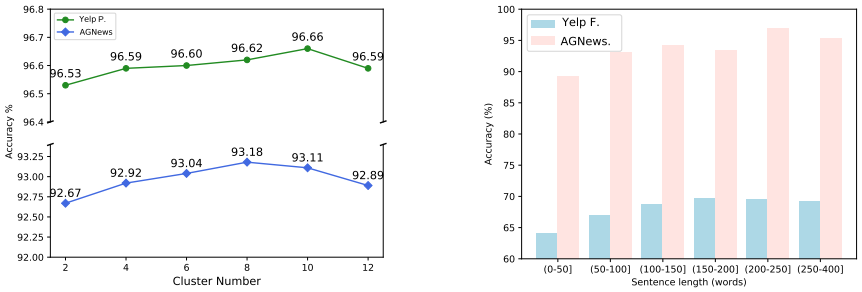
We further make statistics about the clustering results over all the words. To be specific, we count the frequency of all the words being assigned to each cluster. A word is assigned to some cluster means that it is attributed to which with maximum probability. For each cluster, we sort the belonged words according to their frequency. We display the top 20 words in cluster 4, cluster 7 and cluster 3 as shown in Figure 3. We can see that the words in cluster 4 are about business while the words in cluster 7 are science related and in cluster 3, the words are almost meaningless for classification. They are inconsistent with the cluster phenomenon in the heat map.

From the above visualization we have the following observations: first, words that indicate the same topic or with the similar semantics are divided into the same clusters; second, different categories correspond to different clusters; third, representative keywords have much greater probabilities (deeper color in heat map) in the specific cluster. These results exactly correspond to the motivation of the two regularization terms.

To evaluate the relevance between the clustering distribution and the classification results, we utilize t-SNE [9] to visualize the text-level cluster probability distribution on a two-dimensional map as shown in Figure 4. Each color represents a different class. The point clouds are the text-level cluster probability distribution calculated by averaging the words' cluster probability distribution[4]. The location of the class label is the median of all the points belonging to the class. As can be seen, samples with the same class can be grouped together according to their text-level cluster probability distribution and the boundary between different class is obvious, which again demonstrates the strong relevance between the clustering distribution and the classification results.

Analysis on The Number of Semantic Clusters. As the number of semantic clusters is a tuned hyper parameter, we also analyze how performance is influenced by the num-

[4] The cluster probability distribution is the intermediate result during testing. The initial dimension of the points is the cluster number. Each dimension represents the probability that the text belongs to a cluster.



(a) Performance across different cluster numbers.

(b) Performance across different sentence length.

Fig. 5. Quantitative analysis on cluster number and sentence length.

ber of semantic clusters by conducting experiments on AGNews and Yelp P., varying the cluster number m among $\{2,4,6,8,10,12\}$. From Figure 5(a) we can find that the accuracy increases as the number of clusters increase and begins to drop after reaching the upper limit. Obviously, the cluster number does not in line with the class number to gain the best performance.

Analysis on Different Text Length. As the text length in datasets varies, we visualize how test accuracy changes with different text length. We perform experiments on AGNews and Yelp F. while the former has shorter text length than the latter. We divide the text length into 6 intervals according to the length scale. As Figure 5(b) shows, our model performs better on relatively longer texts. With the increase in text length, our model tends to gather more information from the text. This visualization is in line with the overall performance of our model that it performs better on Yah.A., Yelp P. and Yelp F. rather than AGNews and DBPedia as the former three datasets have longer average text length.

4 Related Work

Text representation is an important and fundamental step in modern natural language processing, especially with the current development of approaches based on deep neural networks. Bi-LSTM [13] is a widely-used representation model which conveys information in both directions. Although it alleviates the problem of information vanishment due to the increase of sentence length, our framework can further integrate information by clustering words with similar semantics and give a visual explanation for the results. Another popular representation models are those based on CNN [23, 6, 3] which can capture the word features locally while our model can break the limitation of distance. A burgeoning and effective model is the structured self-attentive sentence embedding model proposed by [8]. They propose a multi-head self-attention mechanism mapping sentence into different semantic spaces and get sentence representation for each semantic space through attention summation. However, they focus on extracting different aspects of a sentence based on the automatically learned relative importance instead of

composing similar aspects together to strengthen the information of each part as our work does.

Except for the above representation models, there are other several models specifically designed for classification. Wang et al. [18] takes the label information into consideration by jointly embedding the word and label in the same latent space. Wang et al. [19] uses a densely connected CNN to capture variable n-gram features and adopts multi-scale feature attention to adaptively select multi-scale features. Qiao et al. [12] utilizes the information of words relative positions and local context to produce region embeddings for classification.

5 Conclusion and Future Work

In this paper, we propose to transform the flat word level text representation to a higher cluster level text representation for classification. We cluster words due to their semantics contained in their contextualized vectors gotten from the concatenation of the initial word embeddings and the outputs of the bi-LSTM encoder. We further introduce regularization schemes over words and classes to guide the clustering process. Experimental results on five classification benchmarks suggest the effectiveness of our proposed method. Further statistical and visualized analysis also explicitly shows the clustering results and provides interpretability for the classification results. In the future, we will try other encoding basement for capturing the words' semantics and we are interested in considering phrases instead of individual words as the basic elements for clustering. The idea of clustering representation is worth trying on other NLP tasks.

Acknowledgments

We thank the reviewers for their valuable comments. This work was supported by the National Key Research and Development Program of China (No. 2017YFC0804001), the National Science Foundation of China (NSFC No. 61876196, NSFC No. 61828302, and NSFC No. 61672058).

References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: Tensorflow: a system for large-scale machine learning. In: OSDI. vol. 16, pp. 265–283 (2016)
2. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. JMLR **3**(Feb), 1137–1155 (2003)
3. Conneau, A., Schwenk, H., Barrault, L., Lecun, Y.: Very deep convolutional networks for text classification. In: EACL. pp. 1107–1116. Association for Computational Linguistics, Valencia, Spain (April 2017), http://www.aclweb.org/anthology/E17-1104
4. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: ECML. pp. 137–142 (1998)
5. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. In: EACL. pp. 427–431 (2017)

6. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: ACL. pp. 655–665 (2014)
7. Kim, Y.: Convolutional neural networks for sentence classification. In: EMNLP. pp. 1746–1751 (2014)
8. Lin, Z., Feng, M., Santos, C.N.d., Yu, M., Xiang, B., Zhou, B., Bengio, Y.: A structured self-attentive sentence embedding. In: ICLR (2017)
9. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008)
10. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
11. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: EMNLP. pp. 1532–1543 (2014)
12. Qiao, C., Huang, B., Niu, G., Li, D., Dong, D., He, W., Yu, D., Wu, H.: A new method of region embedding for text classification. In: ICLR (2018)
13. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing **45**(11), 2673–2681 (1997)
14. Shen, D., Wang, G., Wang, W., Renqiang Min, M., Su, Q., Zhang, Y., Li, C., Henao, R., Carin, L.: Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. In: ACL. pp. 440–450 (2018)
15. Sparck Jones, K.: A statistical interpretation of term specificity and its application in retrieval. Journal of documentation **28**(1), 11–21 (1972)
16. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS. pp. 3104–3112 (2014)
17. Tai, K.S., Socher, R., Manning, C.D.: Improved semantic representations from tree-structured long short-term memory networks. In: ACL. pp. 1556–1566 (2015)
18. Wang, G., Li, C., Wang, W., Zhang, Y., Shen, D., Zhang, X., Henao, R., Carin, L.: Joint embedding of words and labels for text classification. In: ACL. pp. 2321–2331 (2018)
19. Wang, S., Huang, M., Deng, Z.: Densely connected cnn with multi-scale feature attention for text classification. In: IJCAI. pp. 4468–4474 (2018)
20. Xiao, Y., Cho, K.: Efficient character-level document classification by combining convolution and recurrent layers. arXiv preprint arXiv:1602.00367 (2016)
21. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: NAACL-HLT. pp. 1480–1489 (2016)
22. Zhang, T., Huang, M., Zhao, L.: Learning structured representation for text classification via reinforcement learning. In: AAAI. pp. 6053–6060 (2018)
23. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: NIPS. pp. 649–657 (2015)