Short Text Clustering with Transformers

Leonid Pugachev
Moscow Institute of
Physics and Technology
9 Institutskiy per., Dolgoprudny,
Moscow Region, 141701,
Russian Federation
leonid.pugachev@phystech.edu

Mikhail Burtsev
Moscow Institute of
Physics and Technology
9 Institutskiy per., Dolgoprudny,
Moscow Region, 141701,
Russian Federation
burtsev.m@gmail.com

Abstract

Recent techniques for the task of short text clustering often rely on word embeddings as a transfer learning component. This paper shows that sentence vector representations from Transformers in conjunction with different clustering methods can be successfully applied to address the task. Furthermore, we demonstrate that the algorithm of enhancement of clustering via iterative classification can further improve initial clustering performance with classifiers based on pre-trained Transformer language models.

Keywords: short text clustering, language models, transformers
DOI: 10.28995/2075-7182-2021-20-571-577

Кластеризация коротких текстов с помощью трансформеров

Пугачёв Леонид
Московский
физико-технический
институт
141707, Московская область,
г. Долгопрудный,
Институтский пер., д. 9
leonid.pugachev@phystech.edu

Бурцев Михаил
Московский
физико-технический
институт
141707, Московская область,
г. Долгопрудный,
Институтский пер., д. 9
burtsev.m@gmail.com

Аннотация

Методы для решения задачи кластеризации коротких текстов часто используют векторные представления слов для переноса обучения. В этой статье показано, что для решения задачи вместе с различными методами кластеризации могут успешно применяться векторные представления предложений из трансформеров. Более того, показано что алгоритм улучшения кластеризации с помощью итеративной классификации может дополнительно улучшить качество исходной кластеризации с помощью классификаторов, которые основываются на предобученных трансформенных языковых моделях.

Ключевые слова: кластеризация коротких текстов, языковые модели, трансформеры

1 Introduction

There are currently a lot of techniques developed for short text clustering (STC), including topic models and neural networks. The most recent and successful approaches leverage transfer learning through the use of pre-trained word embeddings. In this work, we show that high quality for STC on the range of datasets can be achieved with modern sentence level transfer learning techniques as well. We use deep sentence representations obtained using the Universal Sentence Encoder (USE) [16, 9].

Training of deep architectures can be effective for particular clustering tasks as well. However, application of deep models to clustering directly is difficult since we do not have labels a priori. We show that

fine-tuning of classifiers such as BERT [2] and RoBERTa [11] for clustering can be done with the Enhancement of Clustering by Iterative Classification (ECIC) algorithm [3]. Thus, we develop a combined approach to STC, which benefits from the usage of deep sentence representations obtained using USE and fine-tuning of Transformer models.

The main contributions of the work are as follows. First, we demonstrate that sentence level Transformer transfer learning for clustering gives good results on the range of datasets for STC. Second, fine-tuning of deep models for clustering is hindered because of the lack of labeled data and we propose to use the ECIC algorithm with deep Transformer models which has not been done before to tackle this problem. We called our method Transformer-based Enhancement of Clustering by Iterative Classification (TECIC). Third, we analyzed different combinations of components as constitutional parts of the algorithm, tested different schemes to handle weights during fine-tuning over iterations and developed a new stopping criterion for the algorithm.

2 Related work

One major direction in STC is based on Dirichlet multinomial mixture topic models [17, 15] including GSDPMM [18]. Some variants of these models incorporate word embeddings [6, 4, 15]. These models assume that each document contains only one or a few topics. The models have several advantages over conventional topic modeling such as latent Dirichlet allocation, when used for short texts. First, they better cope with the sparseness of short texts, which carry limited information about word co-occurrences. Second, these models can automatically infer the number of topics. Since only one topic is presented for each document, it is straightforward to use these topic models for clustering, assuming all documents with the same topic as belonging to the same cluster.

Recent works have considered a neural approach for STC. In [14, 12], authors propose to encode texts by pre-trained binary codes. Embeddings of words are then fed in the convolutional neural network which is trained to fit the binary codes. Finally, the obtained representations are used as features with k-means clustering algorithm. The work of [20] uses a somewhat similar strategy called Self-Taught Approach (STA). An autoencoder is pre-trained to obtain low-dimensional features and then learn it together with clustering algorithm by iteratively updating the weights of the autoencoder and centroids of clusters. Finally, they use the resulting features with k-means clustering algorithm. Another idea is to use attentive representation learning with adversarial training for STC [1]. The work of [3] sets the state-of-the-art results on the range of short text datasets using the ECIC algorithm which is simpler than in [20]. They use averaged word embeddings as features for short texts and clustering algorithms such as k-means, to get the initial label assignment. The clustering performance is then improved with iterative outlier detection and classification.

3 Model

In our work, we made several important modifications to the ECIC algorithm [3] to improve their results. Namely, we included modern deep learning components such as USE, BERT and RoBERTa in the algorithm as well tested various methods to handle weights during fine-tuning over iterations such as resumption and re-initialization and developed a new stopping criterion for the algorithm. The main steps of the algorithm are the following:

- Take a dataset D with N texts and K clusters.
- Apply initial clustering and labeling L.
- Set the number of iterations T.
- While $j \leq T$ and the stopping criterion δ is not reached do:
  - Sample P uniformly from $[P_1, P_2]$.
  - Apply outlier detection for each cluster from L to remove outliers from D.
  - If the number of texts in any cluster $n \geq P * N/K$ remove texts randomly from that cluster until $n \geq P * N/K$.
  - Add the rest of D to the train set and add all removed samples to the test set.

- Train a classifier on the train set and update L based on predictions of the classifier on the test set.
- Calculate the criterion δ and update j.

At the initial stage, clustering is carried out using one of the widely used clustering methods (see below). An algorithm for outlier detection is then used to split the dataset into train and test parts. Additional samples can be moved from the train to the test set based on the P number sampled randomly in the range from P_1 to P_2. The train part is used to train the classifier. Outliers and some number of the additional samples are used as a test set and predictions for the test set are used to relabel the dataset. Steps with outlier detection, classification, and relabeling are then repeated until the stopping criterion is reached or the maximum number of iterations is exceeded. As will be shown below, this iterative procedure leads to improved clustering results in many cases.

Averaged word embeddings were used as features in [3, 12]. One of the differences of our study is that we used USE representations[1] [16, 9] for short texts to plug them into one of the clustering algorithms: k-means, Hierarchical Agglomerative Clustering (HAC) or Spectral Clustering (SC). We did not consider the DBSCAN family of algorithms in this work since they infer the number of clusters automatically, while we studied the case when the number of clusters in a dataset is fixed. For k-means we chose the number of initializations 1000, the maximum number of iterations 300, the relative tolerance 10^{-4}. We used a full similarity matrix as well as k-NN and similarity distribution-based sparsification of the similarity matrix [5] with HAC. In both methods of sparsification, we set the number of non-zero elements in each row of the similarity matrix equal to the ratio of the number of samples in the dataset to the number of clusters. In addition, we tested all available linkage criteria for HAC such as single, complete, average, weighted, centroid and Ward. We used the euclidian metric with these criteria. For SC we chose ARPACK eigensolver, the stopping criterion for eigendecomposition of the Laplacian matrix equal 0, and the k-means strategy to assign labels in the embedding space with the number of initializations 10. We tried the Isolation Forest (IF) [8] and Local Outlier Factor (LOF) [7] for outlier detection. For IF we chose the number of base estimators in the ensemble 100. All train samples were used to train each estimator. The proportion of outliers in the dataset was determined automatically as in the original paper [8]. For LOC we chose the number of neighbours 20 and the euclidian metric. We used clustering and outlier detection algorithms implemented in the scikit-learn[2] and scipy[3] python libraries.

In contrast with [3], we used Transformer models such as BERT [2] and RoBERTa [11] for iterative fine-tuning and classification. For these models we used Adam optimizer and tried learning rates values among 2×10^{-5}, 3×10^{-5}, 5×10^{-5}. The number of training epochs per each iteration of the TECIC was varied among 2, 3 and 5. Constant and linear decay learning rate schedules were tested in different runs. We tried different models weight handling such as re-initialization after each iteration of the TECIC or resumption i.e. training with weights obtained at the previous iteration. We used batch size 32 and maximum sequence length 64 for both Transformer models. In addition, we used Multinomial Logistic Regression (MLR) as in other works. For MLR we tried different values of the maximum number of iterations for the solver to converge among 100, 1000, 10000 and the tolerance for stopping criteria among 10^{-4}, 10^{-5}, 10^{-6}. The rest of the parameters for MLR were taken as default in the scikit-learn.

The number of iterations T was set to be 10 for neural classifiers and 50 for MLR. We tried values for P_1 in the range from 0.5 to 0.8 and for P_2 in the range from 0.8 to 0.99. We consider two different stopping criteria. The first stopping criterion [3] is defined as follows $\delta = \frac{1}{N}\sum_i |c_i - c'_i| < \epsilon$ where c_i and c'_i are sizes of clusters determined by the current labeling L and previous labeling L', respectively, and i is a cluster number. For the first stopping criterion we tried ϵ equal to 0.03 and 0.05. The second criterion is reached immediately when δ has a minimum value.

[1] https://tfhub.dev/google/collections/universal-sentence-encoder/1

[2] https://scikit-learn.org/stable/index.html

[3] https://www.scipy.org/

Dataset	K	N	M
Stack Overflow	20	20000	8.2
AG News	4	8000	22.5
Biomedical corpus	20	20000	12.9
Search Snippets	8	12340	17.0
Tweet	89	2472	8.4
Google News TS	152	11109	28.0
Google News T	152	11109	6.2
Google News S	152	11109	21.8

Table 1: Statistics on the datasets used in the study. K is the number of clusters, N is the number of samples, M is the average number of words in a document.

4 Datasets

Our study uses the same datasets as those in a number of previous studies [10, 12, 20, 3] on STC. The statistics on the datasets are presented in Table 1. The Search Snippets dataset is composed of Google search results of 8 different domains [10]. The texts in the Search Snippets dataset represent sets of key words, rather than being coherent texts. The Biomedical corpus is a subset of one of the BioAsQ[4] challenge datasets, where 20000 paper titles were randomly selected from 20 groups [12]. The texts in this dataset contain special terms from biology and medicine. The Stack Overflow is a subset of the challenge data published on Kaggle[5], where question titles 20000 from 20 categories were randomly selected [12]. AG News is a subset of a news titles dataset that was used in [19], where 2000 samples of news titles with descriptions from each of the four categories were taken randomly. The Tweet dataset consists of 2472 tweets which are highly relevant to 89 queries [18]. The Google News TS consists of 11109 news articles titles and snippets about 152 events, while T version of the dataset contains only titles and S contains only snippets of these articles [18].

Note that the former and the latter four datasets can be grouped by the number of clusters. The first group contains relatively low numbers of clusters, while the second has greater numbers of clusters.

5 Results

To measure the performance of our algorithm, we used such metrics as accuracy and Normalized Mutual Information (NMI). The same metrics were used in the number of the previous studies [12, 20, 3, 15]. The value of NMI does not depend on the absolute values of labels. The accuracy is calculated using the Hungarian algorithm [12]. It allows one to rearrange absolute label values to maximize accuracy.

Our experiments on initial clustering tested which of the USE versions and which clustering algorithm should be used to obtain the best quality in terms of both aforementioned metrics. As a result, the old version of USE [16] proved to be better (by a few percent) than the newer one [9] in terms of both metrics on all 8 datasets. We tested k-means, HAC, and Spectral Clustering algorithms with these sentence embeddings. Interestingly, we found that the best clustering method was k-means for the whole group of datasets with the smaller number of clusters (see Table 2). Since k-means is not a deterministic algorithm and its result depends on a particular initialization, we averaged the results over 5 runs. On the contrary, HAC proved to be the best clustering method for datasets with the greater number of clusters (see Table 3). Note we does not provide variance for HAC since this algorithm is determenistic. Overall, k-NN sparsification with the average linkage criterion gave the best results for the four datasets with the greater number of clusters. This differs from the results of [3], where a sparsification based on similarity distribution and the Ward linkage criterion are described as the most effective ones.

[4] http://bioasq.org

[5] https://www.kaggle.com/c/predict-closed-questions-on-stack-overflow/download/train.zip

Method	Metric	Stack Over.	AG News	Biomedical c.	Search Snip.
k-means	Acc.	**81.84±0.01**	**83.87±0.02**	**43.84±0.20**	**74.76±0.13**
	NMI	**80.80±0.01**	**61.88±0.04**	**37.85±0.13**	**54.25±0.16**
HAC Ward	Acc.	74.90	56.89	36.74	61.64
full	NMI	75.73	51.45	32.60	49.79
HAC Ward	Acc.	80.23	57.21	41.27	55.93
d.-b. spar.	NMI	80.44	58.48	37.00	50.23
HAC Ward	Acc.	80.53	58.10	41.79	64.98
k-NN spar.	NMI	80.69	55.74	36.75	52.59

Table 2: Comparison of accuracy and NMI scores for various clustering algorithms for datasets with the smaller number of clusters. Four best performing algorithms are presented.

Method	Metric	Tweet	G. News TS	G. News T	G. News S
HAC weight.	Acc.	**81.59**	64.81	62.95	70.71
full	NMI	**91.97**	86.49	85.95	91.03
HAC Ward	Acc.	74.11	74.71	**80.09**	**85.72**
full	NMI	90.49	90.21	90.67	94.47
HAC aver.	Acc.	78.20	**84.64**	77.56	80.34
k-NN spar.	NMI	91.28	**94.77**	91.14	91.96
HAC weight.	Acc.	74.96	79.56	79.15	83.95
k-NN spar.	NMI	90.42	91.28	**91.23**	**94.39**

Table 3: Comparison of accuracy and NMI scores for various clustering algorithms for datasets with the larger number of clusters. Four best performing algorithms are presented.

We obtained highly competitive results for two (Stack Overflow and AG News) of the four datasets from the first group of datasets (see Table 4). However, we did not get comparable results on the other two datasets (Search Snippets and Biomedical corpus), which can be easily explained. The Search Snippets dataset texts are sets of key words, rather than being coherent texts. Since USE was trained on coherent texts, it cannot produce a good result. The Biomedical dataset almost completely consists of special terms. USE probably did not see many of these terms during training, which explains its poor performance on this dataset. We got the best results for all four datasets from the second group in terms of NMI but not in terms of accuracy (see Table 5).

To improve the results of initial clustering, we tested the iterative classification algorithm with MLR and with neural pre-trained classifiers, such as BERT and RoBERTa. For the neural classifier, the best value for the learning rate was found to be 3×10^{-5} and the number of epochs to train during each iteration was found to be 2. The use of the warm start i.e. training resumption after each iteration instead of re-initialization, and learning rate linear decaying schedule instead of the constant learning rate, did not show any considerable improvement. RoBERTa gave approximately one half percent improvement over the BERT performance. We set T to be 50 for MLR, since the algorithm worked more stable and had potential to improve for the more iterations than for neural classifiers. We found that the use of the second stopping criterion with neural classifiers gives better results than the first one. We did not use any criterion for MLR and collected the metrics at the end of 50 iterations, since both considered metrics grew monotonically for this classifier. We found that P_1 equal 0.7 and P_2 equal 0.95 were the best values for both types of classifiers. We averaged our results over 3 runs in both cases. We did not find any difference in the use of IF or LOF for outlier detection with all classifiers.

The iterative classification achieved the state-of-the-art results on the Stack Overflow and AG News datasets with both types of classifiers and improved the good initial clustering result further (see Table 4).

Method	Metric	Stack Over.	AG News	Biomedical c.	Search Snip.
ECIC [3]	Acc.	78.73±0.17	84.52±0.50	47.78±0.51	**87.67±0.63**
	NMI	73.44±0.35	59.07±0.84	41.27±0.36	**71.93±1.04**
STA [20]	Acc.	59.8±1.9	-	**54.8±2.3**	77.1±1.1
	NMI	54.8±1.0	-	**47.1±0.8**	56.7±1.0
Init. clust.	Acc.	**81.84±0.01**	83.87±0.02	43.84±0.20	74.76±0.13
k-means	NMI	**80.80±0.01**	**61.88±0.04**	37.85±0.13	54.25±0.16
Iter. class.	Acc.	**84.72±0.20**	**84.64±0.08**	44.85±0.20	74.97±0.15
RoBERTa	NMI	**80.63±0.97**	**62.69±0.20**	38.40±0.13	55.17±0.26
Iter. class.	Acc.	**83.31±0.05**	**86.53±0.1**	44.96±0.17	75.87±0.15
Log. Reg.	NMI	**80.68±0.01**	**65.99±0.28**	39.18±0.04	57.36±0.08

Table 4: Comparison with published results of accuracy and NMI scores for datasets with the smaller number of clusters.

Method	Metric	Tweet	G. News TS	G. News T	G. News S
PYPM [13]	NMI	89.8±0.5	**94.9±0.1**	89.0±0.3	91.6±0.2
ECIC [3]	Acc.	**91.52±0.99**	93.56±0.27	**87.18±0.21**	**89.02±0.12**
	NMI	86.87±0.13	94.40±0.11	87.87±1.00	89.96±0.11
GSDPMM [18]	NMI	87.5±0.5	91.2±0.3	87.3±0.2	89.1±0.4
Init. clust.	Acc.	78.20	84.64	77.56	80.34
HAC	NMI	**91.28**	94.77	**91.14**	**91.96**

Table 5: Comparison with published results of accuracy and NMI scores for datasets with the larger number of clusters.

The neural classifier showed a one percent better performance for the Stack Overflow in terms of accuracy than MLR. We did not get comparable results for the Biomedical and Search Snippets datasets, since the iterative classification algorithm can improve the initial clustering result by a limited number of percent and it was low efficient for these two datasets. We did not observe any improvement for the second group of datasets, since it is more difficult for the algorithm to converge to the correct solution during iterations in the case of greater number of clusters.

6 Conclusions

The sentence embeddings based algorithm for enhanced clustering by iterative classification was applied to 8 datasets with short texts. The algorithm demonstrates state-of-the-art results for the 5 out of 8 datasets in terms of NMI and for 2 of 8 in terms of accuracy. We argue that the lack of coherent and common texts causes an inferior performance of the algorithm for the remaining datasets.

The quality of the whole algorithm strongly depends on the initial clustering quality. Initial clustering with USE representations has already allowed us to achieve a competitive performance for a number of datasets. Therefore, due to transfer learning these representations can be readily applied to other datasets even without iterative classification.

References

[1] Attentive Representation Learning with Adversarial Training for Short Text Clustering / Wei Zhang, Chao Dong, Jianhua Yin, Jianyong Wang // arXiv preprint arXiv:1912.03720. — 2019.

[2] Bert: Pre-training of deep bidirectional transformers for language understanding / Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova // arXiv preprint arXiv:1810.04805. — 2018.

[3] Enhancement of Short Text Clustering by Iterative Classification / Md Rashadul Hasan Rakib, Norbert Zeh, Magdalena Jankowska, Evangelos Milios // Natural Language Processing and Information Systems / Ed. by Elisabeth Métais, Farid Meziane, Helmut Horacek, Philipp Cimiano. — Cham : Springer International Publishing, 2020. — P. 105–117.

[4] Enhancing topic modeling for short texts with auxiliary word embeddings / Chenliang Li, Yu Duan, Haoran Wang et al. // ACM Transactions on Information Systems (TOIS). — 2017. — Vol. 36, no. 2. — P. 1–30.

[5] Improving Short Text Clustering by Similarity Matrix Sparsification / Md Rashadul Hasan Rakib, Magdalena Jankowska, Norbert Zeh, Evangelos Milios // Proceedings of the ACM Symposium on Document Engineering 2018. — 2018. — P. 1–4.

[6] Improving topic models with latent feature word representations / Dat Quoc Nguyen, Richard Billingsley, Lan Du, Mark Johnson // Transactions of the Association for Computational Linguistics. — 2015. — Vol. 3. — P. 299–313.

[7] LOF: identifying density-based local outliers / Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, Jörg Sander // Proceedings of the 2000 ACM SIGMOD international conference on Management of data. — 2000. — P. 93–104.

[8] Liu Fei Tony, Ting Kai Ming, Zhou Zhi-Hua. Isolation forest // 2008 Eighth IEEE International Conference on Data Mining / IEEE. — 2008. — P. 413–422.

[9] Multilingual Universal Sentence Encoder for Semantic Retrieval / Yinfei Yang, Daniel Cer, Amin Ahmad et al. // CoRR. — 2019. — Vol. abs/1907.04307. — 1907.04307.

[10] Phan Xuan-Hieu, Nguyen Le-Minh, Horiguchi Susumu. 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. — 2008. — P. 91–100.

[11] Roberta: A robustly optimized bert pretraining approach / Yinhan Liu, Myle Ott, Naman Goyal et al. // arXiv preprint arXiv:1907.11692. — 2019.

[12] Self-taught convolutional neural networks for short text clustering / Jiaming Xu, Bo Xu, Peng Wang et al. // Neural Networks. — 2017. — Vol. 88. — P. 22–31.

[13] Short text clustering based on Pitman-Yor process mixture model / Jipeng Qiang, Yun Li, Yunhao Yuan, Xindong Wu // Applied Intelligence. — 2018. — Vol. 48, no. 7. — P. 1802–1812.

[14] Short text clustering via convolutional neural networks / Jiaming Xu, Peng Wang, Guanhua Tian et al. // Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing. — 2015. — P. 62–69.

[15] Short text topic modeling techniques, applications, and performance: a survey / Qiang Jipeng, Qian Zhenyu, Li Yun et al. // arXiv preprint arXiv:1904.07695. — 2019.

[16] Universal Sentence Encoder / Daniel Cer, Yinfei Yang, Sheng-yi Kong et al. // CoRR. — 2018. — Vol. abs/1803.11175. — 1803.11175.

[17] Yin Jianhua, Wang Jianyong. A dirichlet multinomial mixture model-based approach for short text clustering // Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. — 2014. — P. 233–242.

[18] Yin Jianhua, Wang Jianyong. A model-based approach for text clustering with outlier detection // 2016 IEEE 32nd International Conference on Data Engineering (ICDE) / IEEE. — 2016. — P. 625–636.

[19] Zhang Xiang, LeCun Yann. Text understanding from scratch // arXiv preprint arXiv:1502.01710. — 2015.

[20] A self-training approach for short text clustering / Amir Hadifar, Lucas Sterckx, Thomas Demeester, Chris Develder // Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019). — 2019. — P. 194–199.