TC-DWA: Text Clustering with Dual Word-Level Augmentation

Bo Cheng[1,4,5], **Ximing Li**[2,3,*], **Yi Chang**[1,4,5]
[1]School of Artificial Intelligence, Jilin University, China
[2]College of Computer Science and Technology, Jilin University, China
[3] Key Laboratory of Symbolic Computation and Knowledge Engineering of MOE, Jilin University, China
[4]International Center of Future Science, Jilin University, China
[5]Engineering Research Center of Knowledge-Driven Human-Machine Intelligence, Ministry of Education, China
chengbo9691@gmail.com, liximing86@gmail.com, yichang@jlu.edu.cn

Abstract

The pre-trained language models, *e.g.,* ELMo and BERT, have recently achieved promising performance improvement in a wide range of NLP tasks, because they can output strong contextualized embedded features of words. Inspired by their great success, in this paper we aim to fine-tune them to effectively handle the text clustering task, *i.e.,* a classic and fundamental challenge in machine learning. Accordingly, we propose a novel BERT-based method, namely Text Clustering with Dual Word-level Augmentation (TC-DWA). To be specific, we formulate a self-training objective and enhance it with a dual word-level augmentation technique. First, we suppose that each text contains several most informative words, called anchor words, supporting the full text semantics. We use the embedded features of anchor words as augmented features, which are selected by ranking the norm-based attention weights of words. Second, we formulate an expectation form of word augmentation, which is equivalent to generating infinite augmented features, and further suggest a tractable approximation of Taylor expansion for efficient optimization. To evaluate the effectiveness of TC-DWA, we conduct extensive experiments on several benchmark text datasets. The results demonstrate that TC-DWA consistently outperforms the state-of-the-art baseline methods. Code available: https://github.com/BoCheng-96/TC-DWA.

Introduction

Text **C**lustering (**TC**) is a classic and fundamental challenge in the machine learning community with a wide spectrum of real applications. The basic goal of TC, as its name suggests, is to partition a collection of unlabeled texts into a number of clusters (*i.e.,* text subsets), so that the texts in each cluster share coherent semantic topics, often proximity according to some certain distance measures. During the past decades, many clustering methods, *e.g.,* k-means, Gaussian Mixture Model (GMM), and spectral clustering, have been long established. However, they may tend to be less effective for text data, primarily because the high-dimensional sparse nature of text features results in difficulties of accurately measuring the distances between texts.

The deep representation learning techniques aim to transform the original features to better dense embedded features with deep neural networks in unsupervised manner (Zhang et al. 2020). Due to unsupervised nature of deep representation learning, it is naturally to integrate the representation techniques with clustering objectives, leading to a new topic of *deep clustering* (Wu et al. 2019; Yang et al. 2019; Niu et al. 2020; Gansbeke et al. 2020; Tsai, Li, and Zhu 2021; Huang and Gong 2021). For instance, it solves for the embedded features and cluster memberships jointly under the auto-encoder (Xie, Girshick, and Farhadi 2016) or self-training paradigms (Chang et al. 2017). Thanks to the effectiveness of dense embedded features, the emerging deep clustering methods have empirically achieved clear performance gain comparing against the conventional methods (Tsai, Li, and Zhu 2021; Huang and Gong 2021).

In parallel with deep clustering, the pre-trained language models such as ELMo (Peters et al. 2018) and BERT (Devlin et al. 2019) can enjoy strong contextualized embedded features for text data by capturing high-order, long-range dependency in texts, and meanwhile they are often pre-trained on large-scale text corpora, enabling to capture generic linguistic knowledge for texts (Petroni et al. 2019). Directly fine-tuning them can bring promising performance improvement to many text-specific tasks, ranging from supervised (Chalkidis et al. 2019; Yang et al. 2022), semi-supervised (Li, Li, and Ouyang 2021), to weakly supervised learning tasks (Meng et al. 2020; Ouyang et al. 2022).

To achieve stronger embedded features for texts, there are dozens of studies further fine-tune the language models with contrastive learning in unsupervised manner (Yan et al. 2021; Gao, Yao, and Chen 2021). With this spirit, the recent deep TC method SCCL (Zhang et al. 2021) fine-tunes the language model by integrating the clustering objective with contrastive learning, and can achieve promising improvements compared with SOTA deep clustering methods on benchmark text datasets. Since one basic idea of contrastive learning is to push the embedded features of augmented samples of the same text closer, we regard that **how to generate high-quality augmented samples efficiently** is the key problem of SCCL to some extent. However, the text augmentation techniques used in SCCL, *i.e.,* word insertion, word substitution, and back translation, may generate noisy augmented samples and be costly, potentially hindering the further performance improvement for the TC task.

To remedy those problems, we propose a novel deep

*Corresponding author

TC method based on the language model BERT, namely **T**ext **Clustering** with **D**ual **W**ord-level **A**ugmentation (**TC-DWA**). Basically, BERT can output strong contextualized embedded word features, and the `[CLS]` token is a special word treated as the embedded feature of the text. For each text, we directly apply the embedded features of other words with dominant attention weights as its augmented features. Specifically in TC-DWA, we formulate a self-training objective using the `[CLS]` features of texts, where the prediction is treated as the cluster membership. We then suggest a **dual word-level augmentation** technique to enhance the objective. First, we suppose that each text contains several most informative words, called **anchor words**, supporting the full text semantics. We select anchor words by ranking the norm-based attention weights of words (Kobayashi et al. 2020), and make an indirect consistent constraint between predictive cluster memberships of anchor words and the `[CLS]` token. Second, inspired by the linear characteristic of deep embedded features (Bengio et al. 2013), we can draw augmented embedded features around the `[CLS]` and anchor word features. We formulate a robust expectation form, which is equivalent to generating infinite augmented features, and further suggest a tractable approximation of Taylor expansion for efficient optimization. We conduct extensive experiments on benchmark text datasets, where the results show the superior performance of TC-DWA compared against the existing baselines. The ablation studies indicate the effectiveness of the augmentation techniques.

To sum up, the contributions of this paper are listed as follows:

- We propose a novel BERT-based TC method named TC-DWA under the self-training manner.
- We propose a dual word-level augmentation technique to enhance the self-training objective.
- We conduct extensive experiments to empirically show the superior performance of TC-DWA

Related Work

Text Clustering

During the past decades, the traditional clustering methods, such as *k*-means, GMM, and spectral clustering, have been long investigated for TC tasks. However, the high-dimensional sparse nature of text features makes them less effective due to the difficulties of accurately measuring the text distances. Recently, deep clustering methods have received widespread attention, because they have the ability to jointly learn strong embedded features of texts and clustering assignments with deep neural networks. One branch of deep clustering is built on the auto-encoder paradigm (Xie, Girshick, and Farhadi 2016; Bo et al. 2020), which combines the reconstruction loss of auto-encoder and the specific loss of cluster memberships. Another branch is based on the self-training paradigm such as the techniques with pseudo-semi-supervised manner (Gupta et al. 2020) and contrastive learning (Zhang et al. 2021). In contrast to those methods, our TC-DWA enjoys the strong contextualized embedded features of texts with language models, and further applies a dual word-level augmentation technique to enhance the model.

Language Model

The pre-trained language models have attracted much more attention from the community, where the representatives include auto-regressive models such as (Peters et al. 2018) and auto-encoder models (Devlin et al. 2019; Lewis et al. 2020; Fedus, Zoph, and Shazeer 2021). Typically, they are pre-trained on large-scale collections of texts, so as to learn background linguistic knowledge from texts. On the other hand, they can capture contextualized dependency in texts, enabling to output strong word features. Due to those benefits, directly fine-tuning the language models has been proven to gain promising improvement to many basic tasks, ranging from supervised (Chalkidis et al. 2019; Yang et al. 2022), semi-supervised (Li, Li, and Ouyang 2021), to weakly supervised learning tasks (Meng et al. 2020; Ouyang et al. 2022). In this work, we fine-tune BERT with a self-training objective to effectively handle the unsupervised TC.

Data Augmentation

Data augmentation refers to the technique that increases the diversity of training data without further collecting new instances (Feng et al. 2021). Commonly, it has been widely explored in computer vision, where a number of basic data augmentation operations, *e.g.,* random flipping, rotation, and mirroring, can be directly applied to generate augmented images (He et al. 2016). The automatic data augmentation methods are further developed to select better operation policies (Cubuk et al. 2019). In parallel with those studies, there are also recent data augmentation techniques for texts. The token-level random perturbation operations include random insertion, swap, and deletion (Wei and Zou 2019), and they have been proven to improve the performance on text classification (Xie et al. 2020). The back translation method (Sennrich, Haddow, and Birch 2016) first translates the texts into another language and then back into the original language, so as to generate augmented texts. Seq2seq and language models have also been used for data augmentation (Thakur et al. 2021; Gangal et al. 2021). Besides, there are also several generic data augmentation techniques such as ISDA (Wang et al. 2019). In this work, we focus on the TC task, and suggest the dual word-level augmentation with anchor words and cluster-specific expectations.

The Proposed TC-DWA Method

We now introduce the proposed **TC-DWA**. In this work, we apply the 12-layer BERT-base model[1] (Devlin et al. 2019) as the backbone, but TC-DWA can be also extended to other well-established pre-trained language models.

TC-DWA

Formally, we are given by a dataset of n texts $\{\mathbf{x}_i\}_{i=1}^n$, where $\mathbf{x}$ denotes the raw content of text. TC aims to par-

[1]For convenience, we will call this model as BERT in the following parts of this paper.

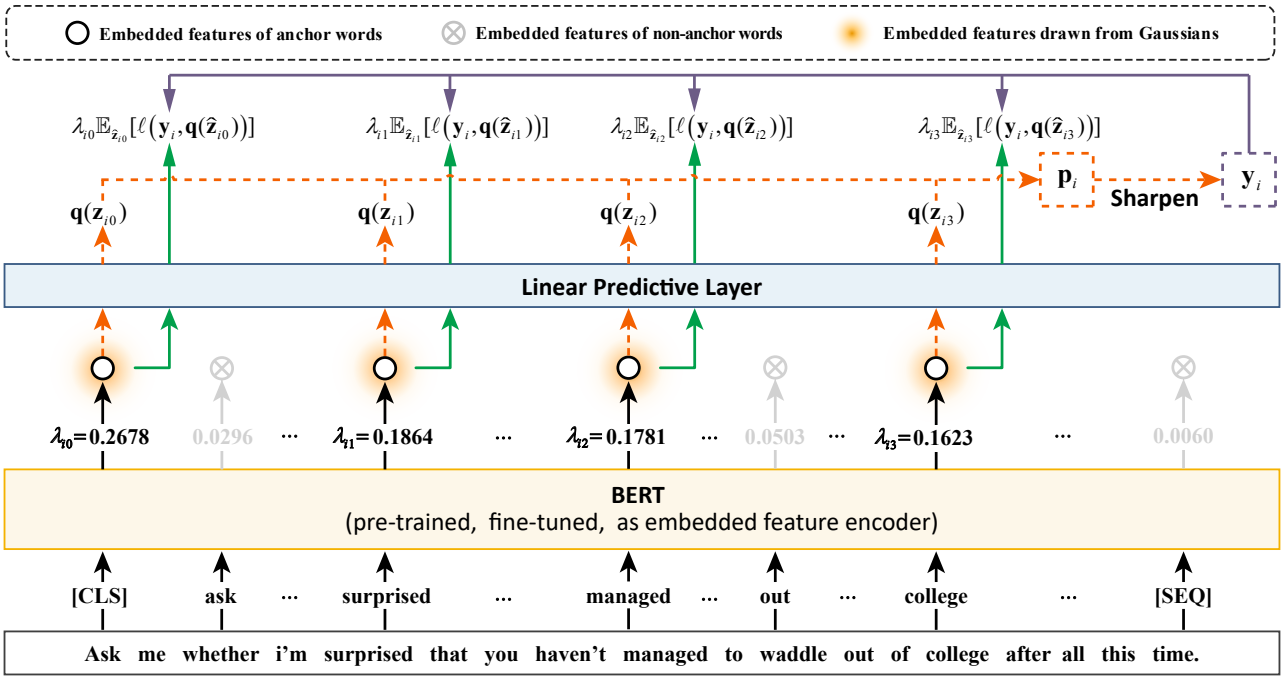

Figure 1: The overall flowchart of TC-DWA. The raw texts are fed into BERT to generate the embedded features and NA weights of words. The [CLS] token and other tokens with top-k NA weights are selected as the anchor words (*i.e.,* AW augmentation). Their embedded features are fed into the predictive layer to compute sharpened target cluster memberships. For each anchor word, we formulate an expectation form of the self-training loss between its embedded feature and the target cluster membership (*i.e.,* EW augmentation).

tition the text dataset into l clusters (*i.e.,* text subsets), so that the texts of each cluster share coherent semantic topics.

To handle the task of TC, we suggest the TC-DWA method, which fine-tunes BERT under the self-training manner with dual word-level augmentation. The overall flowchart of TC-DWA is illustrated in Fig.1. To be specific, for each text $\mathbf{x}_i$, we treat the special [CLS] token $\mathbf{z}_{i0} = f_{\mathbf{\Theta}}(\mathbf{x}_i)$ as the embedded feature of the text, where $\mathbf{\Theta}$ denotes the learnable parameters of BERT. We adopt $\mathbf{z}_{i0}$ to predict the cluster membership of $\mathbf{x}_i$ with a linear layer parameterized by $\mathbf{W}$:

$$q(\mathbf{z}_{i0}) = \mathbf{softmax}(\mathbf{W}\mathbf{z}_{i0}) \triangleq \mathbf{p}_i, \qquad i \in 1, \cdots, n, \quad (1)$$

and then construct a target cluster membership $\mathbf{y}_i$ by referring to (Xie, Girshick, and Farhadi 2016):

$$\mathbf{y}_{ij} = \frac{\mathbf{p}_{ij}^2/\mathbf{g}_j}{\sum_{h=1}^{l} \mathbf{p}_{ih}^2/\mathbf{g}_h}, \quad \mathbf{g}_j = \sum_{i=1}^{n} \mathbf{p}_{ij}, \quad j \in 1, \cdots, l \quad (2)$$

Accordingly, we can formulate a self-training objective as follows:

$$\mathcal{L}(\Theta, \mathbf{W}) = \sum_{i=1}^{n} \ell\left(\mathbf{y}_i, \mathbf{q}(\mathbf{z}_{i0})\right), \quad (3)$$

where $\ell(,\cdot,)$ denotes the loss function, *e.g.,* cross-entropy and KL-divergence.[2]

To enhance the objective of Eq.3, we suggest a dual word-level augmentation technique, including **A**nchor **W**ord (**AW**) augmentation and **E**xpectation **W**ord (**EW**) augmentation.

AW augmentation. The insight of AW augmentation is that for each text, it has several most informative words, called **anchor words**, supporting the full text semantics. Naturally, the predictive cluster memberships of anchor words from the same text must be consistent to each other and also the one of the [CLS] token. Supposing each text contains k anchor words, we use the notation[3] $\mathbf{z}_i = f_{\mathbf{\Theta}}(\mathbf{x}_i)$ to denote the set of embedded features of both the [CLS] token (*i.e.,* $\mathbf{z}_{i0}$) and anchor words $\{\mathbf{z}_{ij}\}_{j=1}^{k}$. We compute a weighted average of predictive cluster memberships by using all words within $\mathbf{z}_i$ as follows:

$$\mathbf{p}_i = \frac{1}{\sum_{j=0}^{k} \lambda_{ij}} \sum_{j=0}^{k} \lambda_{ij} q(\mathbf{z}_{ij}), \qquad i \in 1, \cdots, n, \quad (4)$$

where λ_{ij} is the corresponding weight of $\mathbf{z}_{ij}$. By constructing the target cluster membership $\mathbf{y}_i$ with Eq.2, we can formulate a novel objective with AW augmentation as follows:

$$\mathcal{L}_a(\Theta, \mathbf{W}) = \sum_{i=1}^{n} \sum_{j=0}^{k} \lambda_{ij} \ell\left(\mathbf{y}_i, \mathbf{q}(\mathbf{z}_{ij})\right), \quad (5)$$

[2]This work employs the KL-divergence as the loss function, and other popular ones will be further investigated in the future.

[3]Actually, the model $f_{\mathbf{\Theta}}(\mathbf{x}_i)$ outputs embedded features of all words in $\mathbf{x}_i$. To make notations compact, we reuse it to denote the set of the embedded features of [CLS] token and anchor words.

Algorithm 1: Training of TC-DWA

Input: text dataset $\{\mathbf{x}\}_{i=1}^{n}$ and parameters l, k, γ, β
for $\widehat{t} = 1, 2, \cdots, MaxEpoch$ **do**
  for $t = 1, 2, \cdots, MaxIter$ **do**
    Draw a mini-batch of texts randomly
    Update $\{\mathbf{\Theta}, \mathbf{W}\}$ with the mini-batch
  end for
  Update $\mathbf{\Sigma}$ using Eq.10
end for
Output: predictive cluster memberships $\{\mathbf{p}_i\}_{i=1}^{n}$

Dataset	#Word	#Text	#Class
AG News	49,141	10,000	4
DBPedia	72,179	10,000	14
Newsgroup	192,227	11,014	20

Table 1: Statistics of datasets used in our experiments.

Specially, we select the anchor words by ranking the **N**orm-based **A**ttention (**NA**) weights of words (Kobayashi et al. 2020). For each text $\mathbf{x}_i$, we measure each of word importance by $\|\alpha_{ij}\mathbf{z}_{ij}\|$, where α_{ij} denotes the corresponding attention weight. Further, we may use the NA weights from the shallower layer of BERT denoted by $\|\alpha_{ij}^{h}\mathbf{z}_{ij}^{h}\|, h \in \{1, \cdots, 12\}$. That is because each layer of BERT describes various kinds of information ranging from surface to semantics, and the $\{3, 4, 5, 6, 7, 9, 12\}$ layers have the most representation power (Jawahar, Sagot, and Seddah 2019). In this work, we specially notice that we refer to the `[CLS]` token as a special anchor word for convenience in the following. Besides, the top-k words with largest NA weights are selected as the anchor words finally, and the corresponding weight λ_{ij} is computed as follows:

$$\lambda_{ij} = \frac{\|\alpha_{ij}^{h}\mathbf{z}_{ij}^{h}\|}{\sum_{g=0}^{k}\|\alpha_{ig}^{h}\mathbf{z}_{ig}^{h}\|}, \qquad j \in 0, \cdots, k \tag{6}$$

EW augmentation. The insight of EW augmentation is that we can directly draw augmented embedded features around the ones of anchor words, because the deep features learned by the network usually tend to be linearized (Bengio et al. 2013). Specifically, for each anchor word we draw a number of augmented embedded features $\widehat{\mathbf{z}}_{ij}$ from a Gaussian with mean $\mathbf{z}_{ij}$ and cluster-specific covariance matrix $\mathbf{\Sigma}_{\mathbf{y}_i^*}$, described below:

$$\widehat{\mathbf{z}}_{ij} \sim \mathcal{N}\left(\mathbf{z}_{ij}, \gamma\mathbf{\Sigma}_{\mathbf{y}_i^*}\right), \quad \mathbf{y}_i^* = \underset{j}{\mathbf{argmax}}\ \mathbf{y}_{ij}, \tag{7}$$

where $\gamma \geq 0$ is the scaling parameter, and $\mathbf{y}_i^*$ is the biggest cluster in the current $\mathbf{y}_i$.

We can directly incorporate those augmented embedded features into the objective of Eq.5, but we apply a more robust expectation formulation, which is equivalent to generating infinite augmented features (Wang et al. 2019), described below.

$$\mathcal{L}_e(\Theta, \mathbf{W}, \mathbf{\Sigma}) = \sum_{i=1}^{n}\sum_{j=0}^{k}\lambda_{ij}\mathbb{E}_{\widehat{\mathbf{z}}_{ij}}\left[\ell\left(\mathbf{y}_i, \mathbf{q}(\widehat{\mathbf{z}}_{ij})\right)\right] \tag{8}$$

Unfortunately, this objective of Eq.8 is naturally intractable to solve due to the expectation form. Accordingly, we apply the second-order Taylor expansion at $\mathbf{z}_{ij}$ to form a tractable approximation:

$$\begin{aligned}\mathcal{L}_e(\Theta, \mathbf{W}, \mathbf{\Sigma}) &\approx \sum_{i=1}^{n}\sum_{j=0}^{k}\lambda_{ij}\left(\ell\left(\mathbf{y}_i, \mathbf{q}(\mathbf{z}_{ij})\right) + \frac{1}{2}\mathbf{Tr}\left(\bigtriangledown_{\mathbf{z}_{ij}}^{2}\gamma\mathbf{\Sigma}_{\mathbf{y}_i^*}\right)\right) \\ &\triangleq \mathcal{L}_e^{t}(\Theta, \mathbf{W}, \mathbf{\Sigma}),\end{aligned} \tag{9}$$

where $\mathbf{\Sigma} = \{\mathbf{\Sigma}_i\}_{i=1}^{l}$ are the learnable cluster-specific covariance matrices; $\mathbf{Tr}(\cdot)$ is the trace of a matrix; and $\bigtriangledown_{\mathbf{z}_{ij}}^{2}$ is the Hessian matrix of $\ell\left(\mathbf{y}_i, \mathbf{q}(\mathbf{z}_{ij})\right)$.

Remark: The work of (Wang et al. 2019) has proposed a similar expectation-based augmentation (called ISDA) to our EW augmentation, *i.e.,* Eq.8. The major difference between them is that the ISDA is built on a cross-entropy-specific upper bound of the expectation form, but we derive a more generic Taylor approximation for a much wider range of loss functions.

Training of TC-DWA

We optimize the learnable parameters $\{\mathbf{\Theta}, \mathbf{W}, \mathbf{\Sigma}\}$ by minimizing the approximating objective of Eq.9. The stochastic gradient descent method is adopted to solve the problem. At each iteration t, we draw a mini-batch of texts and then update $\{\mathbf{\Theta}, \mathbf{W}\}$ with backpropagtion by fixing $\mathbf{\Sigma}$. At each epoch $\widehat{t}$, we update $\mathbf{\Sigma}$ by using the current embedded features with target cluster memberships, directly estimated as follows:

$$\begin{aligned}\mu_j^{(\widehat{t})} &= \frac{1}{\sum_{i=1}^{n}\mathbf{y}_{ij}^{(\widehat{t})}\sum_{h=0}^{k}\lambda_{ih}}\sum_{i=1}^{n}\mathbf{y}_{ij}^{(\widehat{t})}\sum_{h=0}^{k}\lambda_{ih}\mathbf{z}_{ih}^{(\widehat{t})}, \\ \mathbf{\Sigma}_j^{(\widehat{t})} &= \frac{\sum_{i=1}^{n}\mathbf{y}_{ij}^{(\widehat{t})}\sum_{h=0}^{k}\lambda_{ih}\left(\mathbf{z}_{ih}^{(\widehat{t})} - \mu_j^{(\widehat{t})}\right)\left(\mathbf{z}_{ih}^{(\widehat{t})} - \mu_j^{(\widehat{t})}\right)^{\top}}{\sum_{i=1}^{n}\mathbf{y}_{ij}^{(\widehat{t})}\sum_{h=0}^{k}\lambda_{ih}}, \\ \mathbf{\Sigma}_j^{(\widehat{t})} &= \beta\mathbf{\Sigma}_j^{(\widehat{t}-1)} + (1-\beta)\mathbf{\Sigma}_j^{(\widehat{t})}, \qquad j \in [l],\end{aligned} \tag{10}$$

where β is a learning rate. After initialization, we alternatively update each learnable parameter, and take the final text cluster assignments by referring to the predictive cluster memberships of texts. For clarity, we summarize the training process of TC-DWA in Algorithm 1.

Time Complexity of TC-DWA

We now discuss the efficiency of TC-DWA. For brevity, we only concentrate on the major operations when optimizing Eq.9 with respect to the parameters of interest $\{\mathbf{\Theta}, \mathbf{W}, \mathbf{\Sigma}\}$. First, we need to compute the gradients of $\{\mathbf{\Theta}, \mathbf{W}\}$. The BERT ingests each text and outputs both the NA weights and embedded features of all its words. Therefore, for each text we can (1) neglect the cheaper anchor words selection with any sorting operation and (2) show the per-epoch complexity of the gradient of the first term in Eq.9 is about $O(\mathcal{T}_1^{\mathbf{\Theta}} + (k+1)\mathcal{T}_1^{\mathbf{W}})$, where $O(\mathcal{T}_1^{\mathbf{\Theta}})$ and $O(\mathcal{T}_1^{\mathbf{W}})$ denote

Method	*AG News*			*DBPedia*			*Newsgroup*		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
k-means	0.447•	0.251•	0.246•	0.543•	0.699•	0.429•	0.253•	0.260•	0.104•
GMM	0.629•	0.375•	0.391•	0.619•	0.694•	0.429•	0.259•	0.260•	0.104•
DEC	0.617•	0.306•	0.303•	0.599•	0.679•	0.488•	0.316•	0.339•	0.178•
SpectralNet	0.483•	0.334•	0.170•	0.589•	0.608•	0.453•	0.184•	0.221•	0.096•
SDCN	0.764•	0.468•	0.494•	0.762•	0.765•	0.651•	0.368•	0.378•	0.217•
BERT	0.817•	0.552•	0.583•	0.580•	0.656•	0.453•	0.245•	0.273•	0.123•
Finetuned-BERT	0.808•	0.530•	0.565•	0.776•	0.814	0.698•	0.470•	0.465•	0.306•
SimCSE	0.779•	0.483•	0.502•	0.743•	0.718•	0.608•	0.405•	0.403•	0.228•
Finetuned-SimCSE	0.797•	0.507•	0.545•	0.775•	0.761•	0.670•	0.444•	0.427•	0.279•
SCCL	0.727•	0.425•	0.408•	0.704•	0.681•	0.576•	0.306•	0.275•	0.121•
TC-DWA (Ours)	**0.827**	**0.566**	**0.604**	**0.795**	**0.823**	**0.711**	**0.491**	**0.487**	**0.334**

Table 2: Clustering results of all comparing methods on benchmark datasets. The best results are highlighted in bold. Besides, the notation • indicates the performance gain of TC-DWA is statistically significant (paired sample t-tests) at 0.01 level.

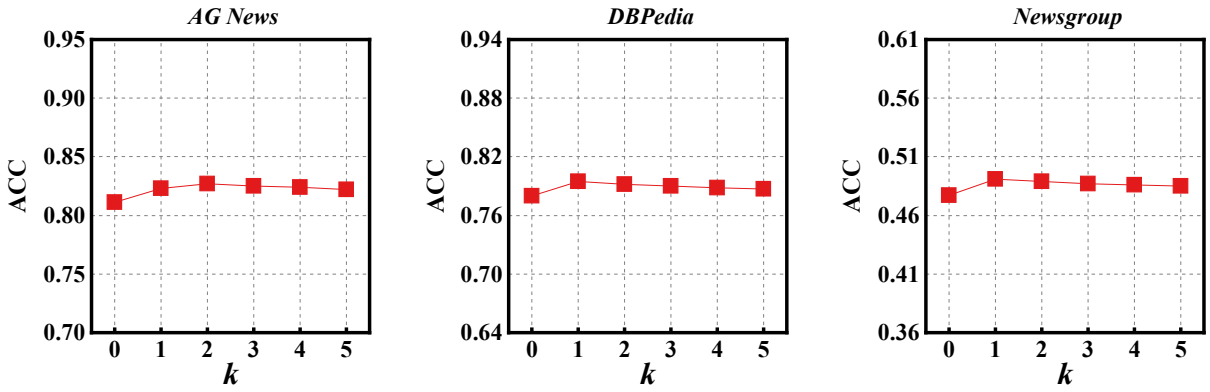

Figure 2: Sensitivity analysis of the anchor word number k with different values on benchmark datasets.

the corresponding complexities of gradients of $\mathbf{\Theta}$, and $\mathbf{W}$ for one anchor word. Analogously, the per-epoch complexity of the gradient of the second Hessian matrix term in Eq.9 is about $O(\mathcal{T}_2^{\mathbf{\Theta}} + (k+1)\mathcal{T}_2^{\mathbf{W}})$, where $O(\mathcal{T}_2^{\mathbf{\Theta}})$ and $O(\mathcal{T}_2^{\mathbf{W}})$ denote the corresponding complexities of gradients of $\mathbf{\Theta}$, and $\mathbf{W}$ for one anchor word. We express that the complexity of $O(\mathcal{T}_2^{\mathbf{\Theta}})$ is at most that of $O(\mathcal{T}_1^{\mathbf{\Theta}})$, and $O(\mathcal{T}_1^{\mathbf{W}})$ and $O(\mathcal{T}_2^{\mathbf{W}})$ are much cheaper with a single linear predictive layer. Second, at each epoch we need to update $\mathbf{\Sigma}$. By referring to Eq.10, the complexity is about $O(nl(k+1))$. In summary, the overall per-epoch time complexity of TC-DWA is about $O(\mathcal{T}_1^{\mathbf{\Theta}} + \mathcal{T}_2^{\mathbf{\Theta}} + (k+1)(\mathcal{T}_1^{\mathbf{W}} + \mathcal{T}_2^{\mathbf{W}} + nl))$.

Experiment

In this section, we first describe the experimental settings, and then compare TC-DWA against the existing TC baseline methods. Finally, we show the results of parameter evaluations, ablative study, and efficiency evaluations.

Experimental Setup

Datasets. In the experiments, we select three commonly used text datasets, *i.e.,* AG News, DBPedia, and Newsgroup.[4] For efficient evaluations, in terms of AG News and DBPedia, we randomly draw 10,000 texts from the full datasets containing massive samples; and in terms of Newsgroup, we employ its standard split of training set. The statistics of these text datasets are shown in Table 1.

Baseline methods. We select 8 existing TC methods as baselines. They are briefly introduced below:

- ***k*-means**: The traditional k-means clustering method with the Bag-of-Words features of texts.
- **GMM**: The traditional GMM clustering method with the Bag-of-Words features of texts.
- **DEC**[5] (Xie, Girshick, and Farhadi 2016): A deep clustering method with the auto-encoder paradigm.

[4]http://qwone.com/ jason/20Newsgroups/
[5]https://github.com/piiswrong/dec

Layer	*AG News*			*DBPedia*			*Newsgroup*		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
$h = 7$	0.800	0.518	0.551	0.761	0.798	0.671	0.401	0.405	0.234
$h = 9$	0.801	0.524	0.556	0.775	0.810	0.692	0.447	0.444	0.280
$h = 12$	**0.827**	**0.566**	**0.604**	**0.795**	**0.823**	**0.711**	**0.491**	**0.487**	**0.334**

Table 3: Sensitivity analysis of h on benchmark datasets. The best results are highlighted in bold.

- **SpectralNet**[6] (Shaham et al. 2018): A deep spectral clustering method with the orthogonality constraint.
- **SDCN**[7] (Bo et al. 2020): A deep clustering method with the auxiliary graph of texts.
- **BERT**[8] (Devlin et al. 2019): The pre-trained BERT model. We feed the raw texts into BERT and then apply k-means with the embedded features. And, **Finetuned-BERT** is the fine-tuned version using Eq.3.
- **SimCSE**[9] (Gao, Yao, and Chen 2021): A BERT-based sentence representation model fine-tuned with contrastive learning. We feed the raw texts into SimCSE and then apply k-means with the embedded features. And, **Finetuned-SimCSE** is the version that we further use Eq.3 to fine-tune SimCSE.
- **SCCL**[10] (Zhang et al. 2021): A deep learning method with instance-wise contrastive learning.

Specifically, for k-means, GMM, DEC, SpectralNet, and SDCN, we employ the TF-IDF text features of the 2,000 most frequent words after removing the stopwords.

Implementation details of TC-DWA We employ the backbone BERT. For all three datasets, limited by the storage of the experimental environment, we set the max sequence length to 128 tokens and the training batch size to 16. By convention, we set the cluster number l as the number of classes for each dataset. We use the Adam optimizer, and the initial learning rates are 5e-6 and 1e-3 for training the parameters of BERT and predictive parameters, respectively. The anchor word number k is set to 1 or 2. For the layer index h of computing NA weights, we fix it to 12 for all the three datasets. Besides, we set the scaling parameter γ defined in Eq.7 as $\gamma = \frac{t}{MaxIter}\gamma_0$, where $\gamma_0 = 0.1$ and t is the iteration number. This setting can reduce the impact of the estimated covariances in the early training stage. Finally, the parameter β defined in Eq.10 is set to 0.9 and the number of epochs is set to 5 for all datasets. All experiments are run on a Linux server with 2 NVIDIA TITAN GTX GPUs and 512G memory.

Evaluation metrics. To evaluate the clustering performance from different views, we employ three metrics: Accuracy (ACC), Normalized Mutual Information (NMI) and Average Rand Index (ARI). Let $\mathbb{L}$ and $\mathbb{C}$ be the sets of ground-truth labels and cluster assignments, respectively. First, the ACC is defined below:

$$\mathrm{ACC} = \frac{\sum_{i=1}^{n} \mathbb{I}\left(\mathbf{l}_i = \mathrm{h}\left(\mathbf{c}_i\right)\right)}{n}, \qquad \mathbf{l}_i \in \mathbb{L},\ \mathbf{c}_i \in \mathbb{C},$$

where $\mathbb{I}(\cdot)$ denotes the indicator function, and $\mathrm{h}(\cdot)$ is the Hungarian mapping function (Kuhn 1955). Second, the NMI is defined as follows:

$$\mathrm{NMI} = \frac{2 \times \mathrm{MI}(\mathbb{C}; \mathbb{L})}{\mathrm{H}(\mathbb{C}) + \mathrm{H}(\mathbb{L})},$$

where $\mathrm{MI}\left(\cdot\right)$ and $\mathrm{H}(\cdot)$ are the mutual information and entropy, respectively. Finally, the ARI is computed as follows:

$$\mathrm{ARI} = \frac{\mathrm{RI}(\mathbb{C}; \mathbb{L}) - \mathbb{E}\left[\mathrm{RI}(\mathbb{C}; \mathbb{L})\right]}{\max\left(\mathrm{RI}(\mathbb{C}; \mathbb{L})\right) - \mathbb{E}\left[\mathrm{RI}(\mathbb{C}; \mathbb{L})\right]},$$

where $\mathrm{RI}(\cdot)$ is the rand index.

Clustering Results

For each dataset, we run all comparing methods 5 times and finally report the average scores of clustering metrics. As shown in Table 2, we have the following observations.

Overall speaking, TC-DWA consistently performs the highest scores on all three metrics across all datasets. Comparing with the traditional k-means and GMM, our TC-DWA and other deep clustering methods always achieve higher scores by a large margin, indicating that the significant advantage of deep embedded features to the shallow features such as Bag-of-Words features in TC tasks. Besides, we can observe that TC-DWA performs better than the existing auto-encoder methods DEC and SDCN, and surprisingly we sometimes gain significant improvements on the competitor SDCN which captures the information of the auxiliary graph of instances, *e.g.,* about $0.03 \sim 0.06$ improvements on DBPedia. This can be an empirical evidence that the contextualized embedded features are more discriminative for texts. Finally, it can be also seen that TC-DWA beats other language model-based baseline methods. For example, on Newsgroup, the improvements over SimCSE are about $0.08 \sim 0.11$. More importantly, TC-DWA achieves better clustering performance than the two fine-tuned versions using Eq.3. For example, compared to Finetuned-BERT, TC-DWA achieves improvements of $0.02 \sim 0.03$ on Newsgroup. The results demonstrate the effectiveness of the proposed dual word-level augmentation technique fairly. More detailed empirical analysis of augmentations will be shown in the ablation study.

[6]https://github.com/KlugerLab/SpectralNet

[7]https://github.com/bdy9527/SDCN

[8]https://github.com/huggingface/transformers

[9]https://github.com/princeton-nlp/SimCSE

[10]https://github.com/amazon-research/sccl

Variant	*AG News*			*DBPedia*			*Newsgroup*		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
TC-DWA	**0.822**	**0.557**	**0.593**	**0.795**	**0.823**	**0.711**	**0.491**	**0.487**	**0.334**
TC-DWA (-A)	0.810	0.530	0.567	0.780	0.821	0.699	0.477	0.470	0.311
TC-DWA (-E)	0.819	0.550	0.588	0.784	0.818	0.701	0.484	0.480	0.325
TC-DWA (-A-E)	0.808	0.530	0.565	0.776	0.814	0.698	0.470	0.465	0.306

Table 4: Clustering results of ablation study. The best results are highlighted in bold.

Variant	*AG News*	*DBPedia*	*Newsgroup*
TC-DWA	353.9 (s)	347.7 (s)	378.7 (s)
TC-DWA (-A-E)	277.1 (s)	278.2 (s)	297.7 (s)

Table 5: Per-epoch running times (s: second) of TC-DWA and TC-DWA (-A-E) on benchmark datasets.

Sensitivity Analysis of Parameters

We verify the impact of the anchor word number k by varying it from 0 to 5. These three metrics have similar trends with respect to k, so we take ACC as an example and report the clustering results in Fig.2. We find that with the introduction of anchor words ($k > 0$), the experimental results are better than when using only the `[CLS]` token as described in Eq.3 ($k = 0$), which once again proves the positive influence of anchor words on clustering. In addition, as the value of k increases, the performance trend of the model on the three datasets can be roughly summarized as follows: rise to the maximum, and then start to decline. This demonstrates that selecting too many anchor words will introduce too much non-representative information, making it difficult to cluster texts effectively.

As shown in Table 3, we search h in the range of $\{7, 9, 12\}$ to verify the influence of the selected layer of BERT on experimental results. We find that when h is 12, TC-DWA performs best on all the three datasets. It demonstrates that richer semantic information has stronger representative power, which leads to more accurate clustering.

Ablation Study

To get a better understanding of the effectiveness of AW and EW augmentations, we conduct a series of ablation experiments. To be specific, we consider the following variants of TC-DWA:

- **TC-DWA (-A)**: The variant without AW augmentation.
- **TC-DWA (-E)**: The variant without EW augmentation.
- **TC-DWA (-A-E)**: The variant without both AW and EW augmentations.

The clustering results of ablation experiments are reported in Table 4. **First**, TC-DWA significantly beats TC-DWA (-A) on all settings, where the performance gain on ACC, NMI, and ARI are about averagely 0.01, 0.02, and 0.02, respectively. Those empirical evidences exactly show that the AW augmentation is beneficial to TC, because the selected anchor words are informative and their augmented features can better represent the full text semantic. **Second**, the improvements of TC-DWA over TC-DWA (-E) are about 0.01 on ACC, 0.01 on NMI, and 0.01 on ARI averagely, exactly demonstrating the effectiveness of the EW augmentation.

Efficiency Evaluation

To evaluate the efficiency of the proposed augmentation technique, we examine the running times of TC-DWA and TC-DWA (-A-E). We report the per-epoch running times across all datasets in Table 5. It can be clearly observed that the running times of the two variants are almost at the same level, where, for example, TC-DWA requires only 76 and 69 seconds more than TC-DWA (-A-E) on AG News and DBPedia, respectively. The empirical efficiency evidences are consistent to the analysis of time complexity. To sum up, we consider that the proposed dual word-level augmentation technique can be efficient and practical in real applications.

Conclusion

In this paper, we propose a novel language model-based TC method named TC-DWA. Basically, we formulate a self-training objective with embedded features of BERT. We then enhance the objective by suggesting a dual word-level augmentation technique, including AW and EW augmentations. First, we use the embedded features of anchor words as augmented features, which are selected by the NA weights of words; Second, we suggest a Taylor expansion approximation of expectation of anchor words, which is equivalent to generating infinite augmented features. We conduct extensive experiments to evaluate the clustering performance of TC-DWA on three benchmark text datasets. The experimental results indicate that TC-DWA significantly outperforms the existing TC methods and the ablation studies show the effectiveness of AW and EW augmentation techniques.

Acknowledgments

This work is supported by the project from the National Key R&D Program of China (No.2021ZD0112501, No.2021ZD0112502), the National Natural Science Foundation of China (NSFC) (No.62276113, No.61976102, No.U19A2065, No.62006094), the Fundamental Research Funds for the Central Universities, JLU.

References

Bengio, Y.; Mesnil, G.; Dauphin, Y. N.; and Rifai, S. 2013. Better mixing via deep representations. In *ICML*, 552–560.

Bo, D.; Wang, X.; Shi, C.; Zhu, M.; Lu, E.; and Cui, P. 2020. Structural Deep Clustering Network. In *WWW*, 1400–1410.

Chalkidis, I.; Fergadiotis, M.; Malakasiotis, P.; and Androutsopoulos, I. 2019. Large-scale multi-label text classification on EU legislation. In *ACL*, 6314–6322.

Chang, J.; Wang, L.; Meng, G.; Xiang, S.; and Pan, C. 2017. Deep Adaptive Image Clustering. In *CVPR*, 5880–5888.

Cubuk, E. D.; Zoph, B.; Mané, D.; Vasudevan, V.; and Le, Q. V. 2019. AutoAugment: Learning Augmentation Strategies From Data. In *CVPR*, 113–123.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, 4171–4186.

Fedus, W.; Zoph, B.; and Shazeer, N. 2021. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *arXiv preprint arXiv:2101.03961*.

Feng, S. Y.; Gangal, V.; Wei, J.; Chandar, S.; Vosoughi, S.; Mitamura, T.; and Hovy, E. H. 2021. A Survey of Data Augmentation Approaches for NLP. In *Findings of ACL*, 968–988.

Gangal, V.; Feng, S. Y.; Hovy, E. H.; and Mitamura, T. 2021. NAREOR: The Narrative Reordering Problem. *arXiv preprint arXiv:2104.06669*.

Gansbeke, W. V.; Vandenhende, S.; Georgoulis, S.; Proesmans, M.; and Gool, L. V. 2020. SCAN: Learning to Classify Images Without Labels. In *ECCV*, 268–285.

Gao, T.; Yao, X.; and Chen, D. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *EMNLP*, 6894–6910.

Gupta, D.; Ramjee, R.; Kwatra, N.; and Sivathanu, M. 2020. Unsupervised Clustering using Pseudo-semi-supervised Learning. In *ICLR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*, 770–778.

Huang, J.; and Gong, S. 2021. Deep Clustering by Semantic Contrastive Learning. *arXiv preprint arXiv:2103.02662*.

Jawahar, G.; Sagot, B.; and Seddah, D. 2019. What does BERT learn about the structure of language? In *ACL*, 3651–3657.

Kobayashi, G.; Kuribayashi, T.; Yokoi, S.; and Inui, K. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *EMNLP*, 7057–7075.

Kuhn, H. W. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2): 83–97.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, 7871–7880.

Li, C.; Li, X.; and Ouyang, J. 2021. Semi-Supervised Text Classification with Balanced Deep Representation Distributions. In *ACL*, 5044–5053.

Meng, Y.; Zhang, Y.; Huang, J.; Xiong, C.; Ji, H.; Zhang, C.; and Han, J. 2020. Text Classification Using Label Names Only: A Language Model Self-Training Approach. In *EMNLP*, 9006–9017.

Niu, C.; Zhang, J.; Wang, G.; and Liang, J. 2020. GATCluster: Self-supervised Gaussian-Attention Network for Image Clustering. In *ECCV*, 735–751.

Ouyang, J.; Wang, Y.; Li, X.; and Li, C. 2022. Weakly-supervised Text Classification with Wasserstein Barycenters Regularization. In *IJCAI*, 3373–3379.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep Contextualized Word Representations. In *NAACL*, 2227–2237.

Petroni, F.; Rocktäschel, T.; Riedel, S.; Lewis, P. S. H.; Bakhtin, A.; Wu, Y.; and Miller, A. H. 2019. Language models as knowledge bases? In *EMNLP*, 2463–2473.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Improving neural machine translation models with monolingual data. In *ACL*, 86–96.

Shaham, U.; Stanton, K. P.; Li, H.; Basri, R.; Nadler, B.; and Kluger, Y. 2018. SpectralNet: Spectral Clustering using Deep Neural Networks. In *ICLR*.

Thakur, N.; Reimers, N.; Daxenberger, J.; and Gurevych, I. 2021. Augmented SBERT: Data Augmentation Method for Improving Bi-Encoders for Pairwise Sentence Scoring Tasks. In *NAACL*, 296–310.

Tsai, T. W.; Li, C.; and Zhu, J. 2021. MiCE: Mixture of Contrastive Experts for Unsupervised Image Clustering. In *ICLR*.

Wang, Y.; Pan, X.; Song, S.; Zhang, H.; Huang, G.; and Wu, C. 2019. Implicit Semantic Data Augmentation for Deep Networks. In *NeurIPS*, 12614–12623.

Wei, J. W.; and Zou, K. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification task. In *EMNLP*, 6381–6387.

Wu, J.; Long, K.; Wang, F.; Qian, C.; Li, C.; Lin, Z.; and Zha, H. 2019. Deep Comprehensive Correlation Mining for Image Clustering. In *CVPR*, 8149–8158.

Xie, J.; Girshick, R. B.; and Farhadi, A. 2016. Unsupervised deep embedding for clustering analysis. In *ICML*, 478–487.

Xie, Q.; Dai, Z.; Hovy, E. H.; Luong, T.; and Le, Q. 2020. Unsupervised Data Augmentation for Consistency Training. In *NeurIPS*, 6256–6268.

Yan, Y.; Li, R.; Wang, S.; Zhang, F.; Wu, W.; and Xu, W. 2021. ConSERT: A Contrastive Framework for Self-Supervised Sentence Representation Transfer. In *ACL-IJCNLP*, 5065–5075.

Yang, X.; Deng, C.; Zheng, F.; Yan, J.; and Liu, W. 2019. Deep Spectral Clustering using Dual Autoencoder Network. In *CVPR*, 4066–4075.

Yang, Z.; Ma, J.; Chen, H.; Lin, H.; Luo, Z.; and Chang, Y. 2022. A Coarse-to-fine Cascaded Evidence-Distillation

Neural Network for Explainable Fake News Detection. In *COLING*, 2608–2621.

Zhang, D.; Nan, F.; Wei, X.; Li, S.; Zhu, H.; McKeown, K.; Nallapati, R.; Arnold, A.; and Xiang, B. 2021. Supporting Clustering with Contrastive Learning. In *NAACL*, 5419–5430.

Zhang, D.; Yin, J.; Zhu, X.; and Zhang, C. 2020. Network Representation Learning: A Survey. *IEEE Transactions on Big Data*, 6(1): 3–28.