IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 15 July 2023, accepted 31 July 2023, date of publication 7 August 2023, date of current version 14 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3302913

RESEARCH ARTICLE

Self-Supervised and Few-Shot Contrastive Learning Frameworks for Text Clustering

HAOXIANG SHI[1] AND TETSUYA SAKAI[2]
[1]Department of Computer Science and Communications Engineering, Waseda University, Shinjuku-ku, Tokyo 169-8050, Japan
[2]Department of Computer Science and Engineering, Waseda University, Shinjuku-ku, Tokyo 169-8050, Japan

Corresponding author: Haoxiang Shi (hollis.shi@toki.waseda.jp)

ABSTRACT Contrastive learning is a promising approach to unsupervised learning, as it inherits the advantages of well-studied deep models without a dedicated and complex model design. In this paper, based on bidirectional encoder representations from transformers (BERT) and long-short term memory (LSTM) neural networks, we propose self-supervised contrastive learning (SCL) as well as few-shot contrastive learning (FCL) with unsupervised data augmentation (UDA) for text clustering. BERT-SCL outperforms state-of-the-art unsupervised clustering approaches for short texts and for long texts in terms of several clustering evaluation measures. LSTM-SCL also shows good performance for short text clustering. BERT-FCL achieves performance close to supervised learning, and BERT-FCL with UDA further improves the performance for short texts. LSTM-FCL outperforms the supervised model in terms of several clustering evaluation measures. Our experiment results suggest that both SCL and FCL are effective for text clustering.

INDEX TERMS Clustering methods, data augmentation, self-supervised learning, semisupervised learning.

I. INTRODUCTION

Text is a fundamental task for various downstream applications, such as opinion mining, automatic topic labeling [1], language modeling, recommendation [2], and query expansion in retrieval [3]. It involves grouping a set of unlabeled texts, regardless of their sizes, so that texts in the same cluster are more similar to each other than to those in other clusters. Due to the following two reasons, achieving good performance in clustering can often be challenging in academic settings: (1) texts are high-dimensional data and (2) the cluster centers and the number of clusters are unknown. Without proper representations to reduce the dimension, the usage of only traditional distance-based clustering methods (e.g., *K-means* and *spectral clustering (SC)*) cannot be effective. Recent progress has suggested that neural language paradigms can assist clustering through learning of representations, and supervised approaches can further improve the effectiveness of representations. However, often there are no ''gold clusters'' (true labels) in real corpus to guide supervised learning. Moreover, obtaining gold clusters is time-consuming and costly, and requires complex statistics algorithms to reduce subjective labeling bias.

Therefore, unsupervised learning is a practical approach to clustering. In early generation, unsupervised neural structures are generative (e.g., generative adversarial network and autoencoder families, etc.), specifically they usually employ a bottleneck structure, where an encoding network learns latent representations for input texts and a generative network transfers the latent representations into input-like information. By optimizing the representation in high-dimension, the most appropriate latent representations can be learned. Then, the best latent representations are used to cluster the texts. However, these generative approaches usually require novel and complex design of architectures (e.g., autoencoder (AE), graph autoencoder (GAE), and variational graph autoencoder (VGAE), etc.) to do without labels, and they are limited in terms of effectiveness.

On the other hand, the recent success of a discriminative contrastive learning (CL) framework in the image classification field [4] inspired similar approaches in the natural language processing (NLP) field (see Section II). CL changes the traditional supervised training manner and achieves weak-label and label-free model optimization. More

The associate editor coordinating the review of this manuscript and approving it for publication was Sotirios Goudos.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

specifically, the original CL introduces a contrastive loss to measure the significance of the similarity of a pair of samples in the sum of the similarities of all sample pairs inside a batch. Thus, the clustering problem is reduced to binary classification. Notably, CL is different from a similarity task (e.g., sentence BERT, a Siamese BERT structure), which only learns the similarity of every single pair. Additionally, to construct positive and negative samples in a batch without any annotations of texts, data augmentation is mandatory (i.e., categorized as Instance-CL by Zhang. et al. [11]). More specifically, augmented samples are positive samples of the original samples, others are regarded as negative samples. As a result, binary classification is transformed into unsupervised learning.

In addition to being free of text annotations, another advantage of contrastive learning (CL) for clustering, as opposed to generative models, is its ability to readily utilize classic and practical neural networks. For instance, among earlier latent representation encoders, Long-short term memory (LSTM) is widely utilized. However, the network structure and the number of parameters tend to limit its performance. Subsequently, language models (LMs) such as bidirectional encoder representations from transformers (BERT) were proposed to improve the effect of representation learning. The origin BERT is pre-trained on domain-free datasets. Thus, fine tuning is necessary to improve performance on domain-specific datasets.

Consequently, in the present study, to combine the effectiveness of the non-generative models with the advantage in terms of label-free CL, we explore the dedicated usage of CL to tackle the text clustering task. More specifically, our contributions are as follows:[1]

- We first propose a few-shot contrastive learning (FCL) framework, which transfers multi-class labels into binary labels. In addition, we adopt the unsupervised data augmentation (UDA) [5] to improve the performance.
- To further remove the requirements of labels, we propose a self-supervised contrastive learning (SCL) framework, where two data augmentations, i.e., multi-language *back translation* (BT) and *random masking* (RM) are used.

We use either LSTM [6] or BERT [7] as the core of the proposed frameworks, to represent the powerful neural architectures and the cornerstone pre-trained LMs, respectively. LSTM or BERT can also be replaced by other models. To verify generalization ability, we evaluated the aforementioned frameworks on two short-text datasets (SearchSnippets [7] and Stackoverflow [8]) and two long-text datasets (20Newsgroup [9] and Reuters [10]).

We enhance the latent representations and do clustering through *K-means* and *SC* accordingly. Our SCL outperforms the existing unsupervised clustering models (e.g., Supporting Clustering with Contrastive Learning [11] (SCCL)) and achieve state-of-the-art accuracy on three datasets except SearchSnippets. In addition to the evaluation in terms of accuracy, we also evaluated the stability of the model to the number of the gold classes, i.e., a hyperparameter, and the results show that SCL has a good ability to estimate the proper number of classes.

On the other hand, FCL is mainly compared to supervised learning, as it needs weak labels. The results show that FCL (using a few of the training data) performs close to supervised learning (using entire training data), and UDA additionally improve the performance for short texts.

[1]An early version of the present study was presented at IEEE SMC 2021 [12].

II. RELATED WORK

A. CONTRASTIVE LEARNING

CL was seen to be first proposed by Hadsell et al. [13], only supervised way was considered (0-1 labels were required). Until 2020, in the visual representation task, Chen et al. [4] proposed an unsupervised CL paradigm based on data augmentation, and achieved a performance improvement of 7%, detonating CL attempts in different fields. For instance, Inoue et al. [14] used semi-supervised CL in speaker recognition; Hassani et al. [15] did graph representation by CL.

B. LEARNING METHODS IN NLP

As the neural network is introduced into NLP tasks, at early stages, long-short term memory (LSTM) families [6], [16] were used in a supervised manner. In recent three years, after the Transformer [17] was proposed, BERT works in a pre-trained and fine-tuning manner, and a lot of tasks were achieved by few-shot learning accordingly. Although pre-trained BERT [7] is capable of unsupervised learning, it is not effective in obtaining latent representations for domain-specific datasets. Several generative models were proposed to enhance unsupervised learning, such as the Autoencoder (AE) [19] families. Generative learning requires complex models and fails to take advantage of the powerful pre-trained LMs.

C. CONTRASTIVE LEARNING IN NLP

Gunel et al. [20] proposed a supervised CL architecture for multiple language tasks. Fanget et al. [21] proposed contrastive self-supervised encoder representations from transformers to predict whether two augmented sentences originated from the same sentence. Li et al. [22] proposed CL with mutual information maximization for cross-domain sentiment classification. Wu et al. [23] designed a metric that covers both linguistic qualities and semantic informativeness using BERT-based CL. Xiong et al. [24] proposed approximate nearest neighbor negative contrastive estimation for dense text retrieval. However, the aforementioned methods neither explored the CL in an unsupervised manner nor applied the CL on a text clustering task.

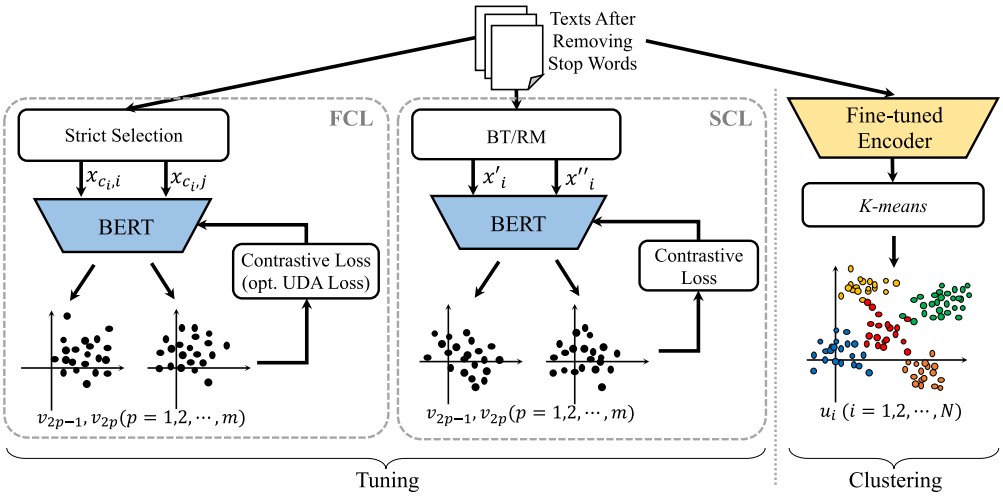

FIGURE 1. Learning framework.

D. TEXT CLUSTERING

In traditional unsupervised clustering, Yuan et al. [25] proposed feature clustering hashing (FCH), and Li et al. [26] proposed subspace clustering guided convex non-negative matrix factorization to perform text clustering. Subsequently, Wang et al. [27] proposed Gaussian bidirectional adversarial topic (G-BAT) models to achieve higher accuracy among these methods. Graph autoencoder (GAE) and variational graph autoencoder (VGAE) [28] can be used for generative unsupervised clustering. Chiu et al. [1] provided a summary of comparisons of these methods. Semi-supervised deep clustering methods have also been proposed based on deep architectures, such as deep clustering network (DCN) [29] and text convolutional Siamese network [30]. It can be seen that the previous researches were semi-supervised or generative unsupervised, and did not consider discriminative unsupervised learning.

III. METHODOLOGY

Figure 1 illustrates the learning framework for FCL and SCL (they have the same input and output format): m pairs of texts (i.e., named as a mini-batch) are fed into the encoder (i.e., the aforementioned "core" of the framework), where intra-pair texts and inter-pair texts are treated as positive samples and negative samples, respectively; the encoder learns the latent representations, and they are used to calculate the contrastive loss. We train/fine-tune the encoder mainly through the contrastive loss. The trained/fine-tuned encoder transfers texts into their latent representations u_i $(i = 1, 2, \cdots, N)$, where N is the number of items in the dataset for clustering), and a cluster (e.g., K-means and spectral clustering, etc.) completes clustering based on these latent representations.

Two main differences between FCL and SCL are the mini-batch construction methods and the loss function choices, which are detailed in Section III-A and Section III-B.

A. FEW-SHOT CONTRASTIVE LEARNING (FCL)

A mini-batch of FCL has m text pairs from the gold clusters, thus the size of a mini-batch is also $2m$. Each pair contains two original texts, $x_{c_i,i}$ and $x_{c_i,j}$, which have the same cluster label c_i. Any two pairs are strictly from two different clusters. To guarantee this, we suggest $m = n$, where n is the number of the gold clusters, which can also help to make full contrasts of the texts in all the clusters.

Note that the aforementioned mini-batch construction procedure is designed upon the gold clusters, which usually contain labels covering all categories (the number of categories is usually greater than 2). To prepare an adapted format of datasets to utilize FCL, one can label the data as (x_i, x_j, t), where x_i and x_j are any two texts, and $t \in \{Negative, Positive\}$ to judge the similarity of the text pair, which essentially makes the clustering weak label guided.

We first give the basic contrastive loss L_{CL}, as shown in Equation 1:

$$L_{CL} = \frac{1}{2m}\sum_{p=1}^{m}[l(2p-1, 2p) + l(2p, 2p-1)] \quad (1)$$

where $l(i, j)$ is a pair loss, and $i = 2p - 1, j = 2p(p = 1, 2, \cdots, m)$ in the p-th pair. The contrastive loss is the average of the pair losses. The pair loss is shown in Equation 2:

$$l(i, j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1,k\neq i}^{2m} \exp(s_{i,k}/\tau)} \quad (2)$$

where $s_{i,j}$ is the cosine similarity of the learned latent representation v_i and v_j $(i = 2p - 1, j = 2p(p = 1, 2, \cdots, m))$ of a text pair ($x_{c_i,i}$ and $x_{c_i,j}$). The τ (> 0) denotes a temperature parameter that can impact the intra-cluster and inter-cluster distance, thereby impacting the clustering accuracy.

One can directly use the L_{CL} (Equation 1) as the FCL loss, L_{FCL}. Additionally, as FCL is without any data augmentation, to improve FCL performance, we propose UDA

TABLE 1. Dataset samples.

	Text	Label
SearchSnippets	wikipedia wiki parliamentary democracy parliamentary system wikipedia encyclopedia utilizing parliamentary systems denoted orange—the constitutional monarchies authority vested	7
Stackoverflow	What type scope Haskell use What type of scope Haskell uses What type of viewer Haskell uses	19
20Newsgroup	I wondering enlighten car I day It door sports car looked late s s It called Bricklin The doors small In addition front bumper separate rest body This I know If tellme model name engine specs production car made history info funky car email I wonder light up the car I'm day The sports car door looked late ss It was called Bricklin The doors small In addition front bumper body of separate rest This I know if the engine model name specs production car made history funky car info email I wonder light up car I day It door sports car seemed late s s It was called Bricklin The small doors In addition to the front bumper separate rest body This I know If tell me the model name engine specifications production car made history information funky car email	7
Reuters	seton co said its board has received a proposal from chairman and chief executive officer philip d kaltenbacher to acquire seton for 15 75 dlrs per share in cash seton said the acquisition bid is subject to kaltenbacher arranging the necessary financing it said he intends to ask other members of senior management to participate the company said kaltenbacher owns 30 pct of seton stock and other management members another 7 5 pct seton said it has formed an independent board committee to consider the offer and has deferred the annual meeting it had scheduled for march 31	4

as an optional enhancement to FCL. UDA was originally proposed for binary emotion classification [5]. When we apply UDA for FCL, every text in a dataset D is back translated to construct D'. A core takes a text x_i in D and three texts $x'_{i,q}(q = 1, 2, 3)$ in D' as inputs, and feeds outputs into a UDA model with parameter set θ to obtain the distributions, $p_\theta(y|x_i)$ and $p_\theta(y|x'_{i,q})$. In the process of FCL, one needs to calculate the average KL-divergences of $\{p_\theta(y|x_{c_i,i}), p_\theta(y|x'_{c_i,i,q})\}$ and $\{p_\theta(y|x_{c_i,j}), p_\theta(y|x'_{c_i,j,q})\}$. The UDA loss L_{UDA} (Equation 3) in a mini-batch is the summation of each average KL-divergence. The loss of the FCL combing the UDA is then shown in Equation 4.

$$L_{UDA} = \sum_{i=1}^{m} \mathcal{KL}(p_\theta(y|x_i) \parallel p_\theta(y|x'_i)) \tag{3}$$

$$L_{FCL+UDA} = L_{CL} + L_{UDA} \tag{4}$$

B. SELF-SUPERVISED CONTRASTIVE LEARNING (SCL)

For SCL, the entire dataset was used for tuning. m texts are first randomly selected from the dataset. We do not require that the texts are from the different classes. For a selected text $x_i, i = 1, 2, \cdots, m$, x'_i and x''_i are two texts generated by a data augmentation methods. The two commonly accepted data augmentation methods are back translation (BT) and random masking (RM). In BT, given a text in language A, this text is translated into language B, and then is translated back to language A again. RM is to randomly mask some words in the original text and then to obtain the augmented test. The original text is excluded in a mini-batch. Thus, the size of a mini-batch is $2m$.

Note that during the mini-batch construction in the SCL, the selected m original texts are likely to be from repeated categories. Therefore, the augmented texts should be positive samples of each other but will be treated as positive samples. This adds noise to the learning process of SCL, potentially limiting performance gains. However, removing this noise requires additional input of information. Therefore, it can be expected that SCL loses some performance in exchange for the advantage of not using any labels. The SCL shares the same loss function with FCL without UDA, that is in Equation 1.

Notably, data augmentation plays different roles in FCL and SCL, respectively. FCL can be used without data augmentation, and in this paper we try to explore the additional performance boost of data augmentation over FCL. SCL requires data augmentation to be label-free. Without data augmentation, sampling in SCL cannot be conducted, and the model the degenerates back to general CL.

IV. EXPERIMENT

A. DATA

Following previous work in text clustering [1], [2], [27], [34], we evaluated text clustering methods using four text categorization datasets. The dataset samples are exhibited in Table 1. The dataset statistics are shown in Table 2, where N is the total number of texts in a dataset, T is the total number of tokens, $L_{Avg.}$ is the average length of the texts (average word counts), and n is the number of the gold clusters. As for Reuters, due to the imbalance in the amount of texts in different classes, we select the top 10 classes from all 20 classes.

B. SETTINGS

In our main experiments (to perform both SCL and FCL), we use the basic and uncased BERT released by Hugging Face[2] and LSTM as the encoder, respectively. The dimension of a latent representation v_i is 20. The mini-batch size is $2m = 160$ and $2m = 40$ for the short texts and the long texts, respectively. For LSTM, we adopt Glove 300-dimension pre-trained vector to initial the embedding matrices. The hidden dimension in LSTM is 300 for both FCL and SCL. The activation function is *Tanh*() and the rate of dropout is 0.2. Max sentence length is set to be 128. The learning rate is 2×10^{-5}, and the optimizer is *Adam*. In the contrastive loss

[2]https://huggingface.co/bert-base-uncased

TABLE 2. Dataset statistics.

	Dataset	N	T	$L_{Avg.}$	n
Short Texts	SearchSnippets	12.3k	31k	17.9	8
	Stackoverflow	20k	23k	8.3	20
Long Texts	20Newsgroup	1.8k	56k	245	20
	Reuters	7.6k	28k	141	10

function, $\tau = 0.5$, we verified that this value is the best setting for clustering task in Section V-A.

In SCL, we chose Google Translate[3] to perform English-Spanish-English and English-French-English BT to generate positive pairs. 30% and 15% of words are masked for a short and a long text in RM, respectively. This is because a small percentage setting may cause no words to be masked in a short text, whereas a large percentage may cause a long text to lose contexts. For FCL as well as FCL with UDA (FCL + UDA), we take 1% and 10% labeled items to tune BERT and cluster all the texts for each dataset. We chose *K-means* as the method to cluster the texts. Note that as we are not sure if the learned latent presentations can be clustered in the Euclidean space, we also tried using the *SC* as an alternative clustering algorithm. As a result, *SC* slightly underperformed *K-means*. This suggests the latent representations are linearly space separable.

C. EVALUATION MEASURES

For SCL, to enable comparisons with previous works [1], [2], we evaluated clustering methods using clustering accuracy (ACC) and normalized mutual information (NMI) for the short texts, and ACC and adjusted mutual information (AMI) [31] for the long texts. These measures assume that the desired number of clusters n (i.e., the number of gold classes) is given to the system. However, this assumption may not be entirely practical. We therefore additionally adopted BCubed F1 Score [33] to evaluate our frameworks while varying the estimated number of gold classes $\hat{n}$. For FCL, we choose ACC, AMI, and adjusted random index (ARI) for comparisons. We illustrate the definitions and usage of ACC, as the clustering ACC is different from classification accuracy. We additionally detail BCubed F1 because it is an essential metric to evaluate the performance of our methods.

1) ACC

The ACC requires the mapping between the predicted clusters and the gold clusters, which is defined as Equation 5:

$$ACC = \frac{\sum_{i=1}^{n} \delta(c_{t_i}, map(\hat{c}_{t_i}))}{N} \tag{5}$$

where n is the number of the clusters, and N is the total number of texts. c_{t_i} is the ground truth cluster of text i, and $\hat{c}_{t_i}$ is the predicted cluster of text i. If $c_{t_i} = map(\hat{c}_{t_i})$, then $\delta(c_{t_i}, map(\hat{c}_{t_i})) = 1$. The function $map(\cdot)$ indicates a permutation mapping that best matches the predicted clusters to the ground truth classes.

The method to calculate clustering accuracy follows CoClust [35]. First, we construct a $L \times L$ matrix, where $L = \max(|C|, |\hat{C}|)$, and C and $\hat{C}$ are the "ground truth" partition (i.e., the gold clusters) and the predicted partition of a dataset. This measure assumes a previously known $|C| = n$, thus $|\hat{C}| = |C| = n$. Then, we count the number of repeated texts in C_i and $\hat{C}_j$ as $L(i, j)$, where C_i or $\hat{C}_j$ is the i-th class or j-th cluster of C or $\hat{C}$, respectively. L now can be seen as an adjacent matrix of a bi-graph, and we can apply a method such as the *Hungarian* algorithm to find the maximum perfect match of this bi-graph. Thus, we can get the best mapping between C and $\hat{C}$. We can calculate ACC and other metrics based on this mapping.

2) BCubed F1 SCORE

The precision and recall of an item i in the dataset is:

$$p(i) = \frac{|\hat{C}(i) \cap C(i)|}{|\hat{C}(i)|}, \; r(i) = \frac{|\hat{C}(i) \cap C(i)|}{|C(i)|} \tag{6}$$

where $\hat{C}(i)$ is the predicted cluster that i belongs to, and $C(i)$ is the gold cluster that i belongs to.

The F score for i is defined as:

$$f(i) = (1 + \beta^2)\frac{p(i) \cdot r(i)}{\beta^2 p(i) + r(i)} \tag{7}$$

where we let $\beta = 1$. Finally, the BCubed Precision (P), Recall (R), and F1 (F) are defined as:

$$P = \frac{1}{N}\sum_i p(i), \; R = \frac{1}{N}\sum_i r(i), \; F = \frac{1}{N}\sum_i f(i) \tag{8}$$

D. BASELINES

For short texts, we primarily compared SCL with SCCL (i.e., state-of-the-art model for short texts) and its baseline SIF + Aut.,Self-Train. We also included the other methods reported by Hadifar et al. [2] for comparisons. For long texts, we primarily compared SCL with SS-SB-MT [1] (i.e., state-of-the-art model for long texts) and its baseline G-BAT [27]. We also included the other methods for comparisons such as GAE, AE and VAGE reported by [1]. We present the main baselines including SCCL, SIF + Aut.,Self-Train., SS-SB-MT and G-BAT.

1) SCCL

This method contains two part: (1) Instance-wise Contrastive Learning (2) clustering learning: pushing the cluster assignment probability towards the target distribution by optimizing the KL divergence. This method concurrently optimizes a top-down clustering loss and a bottom-up instance-wise contrastive loss by utilizing a BERT pretrained model for the task of short text clustering.

2) SIF + AUT.,SELF-TRAIN

The model includes three steps: (1) short texts are embedded using Smooth Inverse Frequency (SIF) embeddings; (2)

[3]https://translate.google.com/?hl=zh-CN

TABLE 3. SCL performance comparisons for short-text datasets: $*$ denotes the results reported by [2]; for SCCL, r are our reproduced *average* results of 10 experiments; the **bold** results are the best scores; BERT-emb.=BERT-embedding; S. BERT=Sentence BERT.

Method		SearchSnippets		Stackoverflow	
		ACC	NMI	ACC	NMI
TF		24.7^{*}	9.0^{*}	13.5^{*}	7.8^{*}
TF-IDF		33.8^{*}	21.4^{*}	20.3^{*}	15.6^{*}
Skip-Thought		33.6^{*}	13.8^{*}	9.3^{*}	2.7^{*}
SIF		53.4^{*}	36.9^{*}	30.5^{*}	28.9^{*}
STC^2		77.0^{*}	62.9^{*}	51.1^{*}	49.0^{*}
SIF + Aut.,Self-Train.		$77.1^{*,r}$	$56.7^{*,r}$	$59.8^{*,r}$	$54.8^{*,r}$
SCCL		**80.2**^{r}	61.8^{r}	60.5^{r}	52.7^{r}
LSTM	SCL (BT)	70.9	54.1	68.4	61.9
	SCL (RM)	75.1	58.7	61.0	58.0
BERT	BERT-emb.	48.7	45.4	61.9	43.7
	S. BERT	60.0	41.8	40.9	31.9
	SCL (BT)	78.2	63.6	**77.7**	**72.5**
	SCL (RM)	77.2	**63.8**	67.5	57.4

during a pre-training phase, a deep autoencoder is applied to encode and reconstruct the short-text SIF embeddings; (3) in a self-training phase, we use soft cluster assignments as an auxiliary target distribution, and jointly fine-tune the encoder weights and the clustering assignments. Note that this method was not tested with long texts in the original work [2].

3) SS-SB-MT

This method builds a keyword correlation graph (KCG) for a text using node features (embeddings from a SBERT), word co-occurrence edges, sentence similarity edges and sentence position edges. Then the constructed graphs of the texts are fed into a Multi-Task GAE (MTGAE). Clustering is based on the latent representations from MTGAE. The name SS-SB-MT comes from **S**entence **S**imilarity, **SB**ERT and **MT**GAE. Note that this method is not suitable for short texts, as the KCG is hard to construct when the number of word co-occurrence edges is too small.

4) G-BAT

The proposed bidirectional adversarial training (BAT) consists of three components: (1) the Encoder E takes the V-dimensional document representation d_r sampled from text corpus C as input and transforms it into the corresponding K-dimensional topic distribution θ_r; (2) the Generator G takes a random topic distribution θ_f drawn from a Dirichlet prior as input and generates a V-dimensional fake word distribution d_f; (3) the Discriminator D takes the real distribution pair $p_r = [\theta_r; d_r]$ and fake distribution pair $p_f = [\theta_f; d_f]$ as input and discriminates the real distribution pairs from the fake ones. The outputs of the discriminator are used as supervision signals to learn E, G and D during adversarial training.

We compared FCL with a supervised method that takes training data and cross-entropy loss to tune BERT [32] and to train the LSTM model. For 20Newsgroup, we use its original training data and test data, which occupy approximately 70% and 30% of the total data, respectively. For the datasets which are not originally divided into training and test parts, we randomly select 70% of the items in each dataset to tune the models, and the other 30% for clustering evaluation.

TABLE 4. SCL performance comparisons for long-text datasets: $\star$, †, ‡, and $*$ denote the results reported by [1], [25], [27], and [34] respectively; for SIF + Aut.,Self-Train., r are our reproduced results; for SS-SB-MT, $r1$ are our reproduced results; $r2$ are the results reproduced and reported by [18]; the **bold** results are the best scores; BERT-emb.=BERT-embedding; S. BERT=Sentence BERT.

Method		20Newsgroup		Reuters	
		ACC	AMI	ACC	AMI
NMF		$31.9^{\star}$	$45.3^{\star}$	$49.6^{\star}$	$43.8^{\star}$
TF-IDF		$33.7^{\star}$	$41.7^{\star}$	$35.0^{\star}$	$45.6^{\star}$
FCH		$33.8^{\dagger}$			
LDA		$37.2^{\star}$	$28.8^{\star}$	$54.9^{\star}$	$50.3^{\star}$
G-BAT		$41.3^{\ddagger}$			
GAE		$41.4^{\star}$	$49.3^{\star}$	$52.3^{\star}$	$48.4^{\star}$
AE		$41.7^{\star}$	$48.7^{\star}$	$53.7^{\star}$	$55.0^{\star}$
VGAE		$43.1^{\star}$	$48.1^{\star}$	$53.1^{\star}$	$53.3^{\star}$
DCN		44.0^{*}			
SIF + Aut.,Self-Train.		29.0^{r}	28.5^{r}	34.0^{r}	33.7^{r}
SS-SB-MT		$47.4^{\star}$	**53.0***	$56.3^{\star}$	**58.4***
SS-SB-MT		10.4^{r1}	5.3^{r1}	34.8^{r1}	12.3^{r1}
SS-SB-MT		10.8^{r2}	5.5^{r2}	35.2^{r2}	12.5^{r2}
SCCL		43.5	44.6	47.4	47.6
LSTM	SCL (BT)	30.2	32.1	55.0	46.0
	SCL (RM)	23.5	24.3	50.3	33.3
BERT	BERT-emb.	28.5	27.7	33.1	37.4
	S. BERT	33.3	28.7	27.7	24.9
	SCL (BT)	**50.1**	47.1	61.7	45.4
	SCL (RM)	44.1	42.2	**64.4**	46.3

E. RESULTS

1) SCL PERFORMANCE

a: SHORT TEXT

As an essential comparison, in addition to using CL, SCCL [11] also incorporates the clustering task into the loss function, thereby replacing the traditional clustering process. Our reproduction of SCCL suggests that introducing clustering into the loss will cause serious overfitting, and the clustering accuracy is unstable (the difference between different experiments can reach 7.1 scores). In Table 3 (results for short texts), we first reproduce SCCL using the source code provided by Zhang. et al [11], and the results are several points lower than the reported ones. This is mainly due to unstable results of multiple experiments. Then, as for SIF + Aut.,Self-Train., our reproduced results are exactly the same as the reported ones. When we choose BERT as the text encoder, for SearchSnippets, SCL (BT) underperforms SCCL by 2.0 points, while outperforms SIF + Aut.,Self-Train. by 1.1 points in terms of ACC. SCL (RM) outperforms SCCL and SIF + Aut.,Self-Train. by 2.0 and 7.1 points in terms of NMI, respectively. In contrast, for Stackoverflow, our results show larger improvements. More specifically, SCL (BT) outperforms SCCL and SIF + Aut.,Self-Train.

by 17.2 and 17.9 points in ACC, and by 19.8 and 17.7 points in NMI, respectively.

To verify that SCL fine tuning of BERT has independent effectiveness, we also report the clustering metrics using original BERT embeddings and Sentence BERT (pretrained by a unsupervised similarity task). LM-based encoder outperforms very traditional TF and TF-IDF due to latent representations in vector space. Furthermore, SCL enhances this representation ability to meet the intent of a clustering task.

When we use LSTM as the encoder, for SearchSnippnets, SCL (RM) underperforms SCCL and SIF + Aut.,Self-Train. by 5.1 and 2.0 points in terms of ACC. In terms of NMI, it underperfroms SCCL by 3.1 points, however outperforms SIF + Aut.,Self-Train. by 2.0 points. For Stackoverflow, LSTM shows substantial improvements. SCL (BT) outperforms SCCL by 13.5 and 9.2 points in ACC and in NMI, respectively. In contrast, SCL (BT) outperforms SIF + Aut.,Self-Train. by 8.6 and 7.1 points in ACC and NMI, respectively. Note that compared to SCCL and SIF + Aut.,Self-Train., LSTM is much lighter and easier to deploy. The results also suggest that SCL framework enhances the performance of the classic models, by taking an advantage from the self-supervised learning approach.

b: LONG TEXT.

As shown in Table 4 (results for long texts), the performance of SS-SB-MT that we reproduced is far below what was reported in the original work [1]. As Chiu et al. [1] provided neither code nor the entire parameters, although we exactly follow the description in the paper, we cannot guarantee that every parameter is set correctly. However, note that our reproduced results are similar to the ones reported by Alireza [18]. We also test SCCL and its baseline SIF + Aut.,Self-Train. for long texts, and their performance heavily deteriorates.

When the encoder is BERT, for 20Newsgroup, SCL (BT) outperforms SS-SB-MT and G-BAT by 2.7 points and 8.8 points in ACC, respectively; meanwhile, for Reuters, both SCL (BT) and SCL (RM) outperform SS-SB-MT and G-BAT in ACC. Furthermore, SCL (RM) achieves the best ACC. However, SS-SB-MT outperforms SCL in AMI for 20Newsgroup and Reuters. This may be because SS-SB-MT uses topics to build graphs for the texts, and the clustering results are more interpretable.

To highlight the advantage of our method, We specifically compare the SCCL for long-text clustering with SCL using BERT. For 20Newsgroup, SCCL underperforms SCL (BT) by 6.6 and 2.5 points in ACC and AMI, respectively. For Reuters, SCCL underperforms SCL (RM) by 17.0 points, however outperforms SCL (RM) by 1.3 points in AMI. Considering the performance in both short-text and long-text clustering, SCCL lacks either stability and generation ability. It suggests that clustering learning has a side effect on performance. Additionally, original BERT embeddings and Sentence BERT show less effectiveness for long-text than short-text clustering, This confirms the role of SCL fine tuning in improving performance.

TABLE 5. BCubed F1 Scores under BERT-SCL for the datasets with unknown *n* from gold clusters; the bold results are the best scores.

Method		n	SearchS.	Stack.	20N.G.	Reut.
BERT	SCL (BT)	5	44.8	27.2	25.7	**48.9**
		10	**46.2**	43.0	34.3	47.4
		15	37.6	55.9	**36.0**	40.4
		20	31.0	64.5	34.7	33.4
	SCL (RM)	5	49.1	26.2	23.2	**53.5**
		10	**49.3**	38.8	30.2	51.5
		15	40.6	47.1	**31.5**	45.6
		20	34.0	**51.3**	30.9	35.0

TABLE 6. BCubed F1 Scores under LSTM-SCL for the datasets with unknown *n* from gold clusters; the bold results are the best scores.

Method		n	SearchS.	Stack.	20N.G.	Reut.
LSTM	SCL (BT)	5	51.2	22.5	15.1	42.4
		10	**53.1**	34.8	19.3	**46.2**
		15	44.6	45.7	**19.7**	39.1
		20	37.1	**50.8**	19.0	36.1
	SCL (RM)	5	54.6	22.1	16.8	37.8
		10	**58.5**	34.3	**17.5**	**37.9**
		15	43.8	43.3	17.4	34.1
		20	37.2	**47.1**	16.9	33.2

When the encoder is LSTM, for 20Newsgroup, SCL (BT) outperforms SCL (RM), and SCL (BT) outperforms SIF + Aut.,Self-Train. by 1.18 points and 3.6 points in ACC and AMI, respectively. For Reuters, SCL (BT) underperforms SS-SB-MT by 1.3 and 12.4 points in ACC and AMI, respectively. The reason may be that LSTM has light structure and no pre-trained knowledge (it is also an advantage from a deployment perspective, even if it hurts performance) to effectively learn the representations of long texts.

We further compare BCubed F1 Score for SCL (BT) and SCL (RM) in Table 5 for BERT under different numbers of clusters, $\hat{n}$. For SearchSnippets and Stackoverflow, the best scores are obtained when $\hat{n}$ is close to the n of the gold clusters (8 and 20, respectively). However, for 20Newsgroup and Reuters, we obtained the best scores when $\hat{n}$ are 15 and 5, respectively. Then we change the encoder into LSTM and report and the BCubed F1 Score in Table 6. It can be observed that except 20Newgroup, the LSTM helps to obtain the best score when $\hat{n}$ is close to or equals the n of the gold clusters. Results in Table 5 and Table 6 indicate that both BERT and LSTM are capable of distinguishing categories, as the encoders of SCL.

As mentioned earlier, we introduce UDA into FCL to enhance the model ability to distinguish categories by adding KL divergence. BT/RM (data augmentation for positive and negative sampling) in SCL and UDA have different optimization objectives. Nevertheless, we still explore the performance of SCL combined with UDA, and the results are shown in the Table 7. Compared with FCL+UDA, SCL+UDA has little improvement in performance.

TABLE 7. SCL ablation study: the **bold** results are the best scores.

	Method	SearchSnippets		Stackoverflow		20Newsgroup		Reuters	
		ACC	NMI	ACC	NMI	ACC	AMI	ACC	AMI
BERT	SCL(BT)	78.2	63.5	77.6	72.4	49.8	46.3	61.5	44.9
	SCL(BT) + UDA	78.2	**63.6**	**77.7**	**72.5**	**50.1**	**47.1**	**61.7**	**45.4**
	SCL(RM)	**77.3**	63.8	67.4	57.3	44.0	42.1	64.1	46.2
	SCL(RM) + UDA	77.2	63.8	**67.5**	**57.4**	**44.1**	**42.2**	**64.4**	**46.3**
LSTM	SCL(BT)	**71.1**	**54.3**	68.3	61.8	30.0	32.0	54.8	45.8
	SCL(BT) + UDA	70.9	54.1	**68.4**	**61.9**	**30.2**	**32.1**	**55.0**	**46.0**
	SCL(RM)	75.0	58.5	60.8	57.7	23.1	24.0	55.1	33.1
	SCL(RM)+ UDA	**75.1**	**58.7**	**61.0**	**58.0**	**23.5**	**24.3**	**50.3**	**33.3**

TABLE 8. FCL performance comparisons: the **bold** results are the best scores.

	Method	SearchSnippets			Stackoverflow			20Newsgroup			Reuters		
		ACC	AMI	ARI	ACC	AMI	ARI	ACC	AMI	ARI	ACC	AMI	ARI
BERT	1%FCL	63.8	46.8	46.6	68.1	59.3	59.5	33.3	34.5	34.5	70.9	61.8	66.4
	1%FCL + UDA	64.2	47.1	47.3	67.3	58.7	58.9	34.4	35.6	35.3	64.3	64.1	64.9
	10%FCL	89.6	75.9	77.4	86.0	79.2	79.1	**66.0**	**59.5**	**59.5**	71.3	**62.0**	67.9
	10%FCL + UDA	**90.7**	**78.1**	**79.7**	**87.2**	**80.6**	**80.7**	62.1	54.6	54.0	**78.1**	58.5	58.5
	1% Supervised (ref.)	63.6	46.5	47.0	67.3	58.0	58.5	34.7	34.6	35.2	76.2	56.5	56.7
	10% Supervised (ref.)	90.7	77.5	77.5	84.4	75.5	75.0	62.2	53.0	53.9	83.2	62.8	62.2
	Supervised (ref.)	96.0	88.7	90.5	89.0	81.0	78.2	69.1	58.8	49.7	88.5	70.0	80.7
LSTM	1%FCL	55.7	33.9	34.2	52.0	38.1	38.9	24.0	25.2	25.9	33.0	22.7	23.8
	1%FCL +UDA	56.3	34.3	34.7	52.2	38.2	38.9	24.1	25.3	26.1	33.2	22.9	24.1
	10%FCL	83.8	65.6	65.8	72.0	70.3	70.8	37.0	33.2	33.9	62.4	44.8	45.6
	10%FCL + UDA	84.0	65.7	65.6	71.5	70.3	70.4	37.6	33.1	33.5	63.1	45.6	46.0
	1% Supervised (ref.)	27.4	9.6	9.9	15.5	9.1	9.7	16.6	17.0	17.8	36.1	31.9	32.6
	10% Supervised (ref.)	55.5	41.7	41.9	63.1	55.1	55.6	27.8	28.5	28.7	57.5	48.1	48.5
	Supervised (ref.)	91.3	57.6	57.7	81.1	55.1	55.6	60.5	21.2	21.5	86.9	41.5	42.1

2) FCL PERFORMANCE

Comparing FCL to SCL and other unsupervised methods may not be fair, because FCL is a semi-supervised model and uses weak labels. Therefore, we compare the FCL results to the aforementioned supervised method by employing two encoders (see Section IV-D). In addition to the training data percentage, we emphasize that even if FCL and supervised learning use the same proportion of training data, the advantage of FCL is also reflected in the no requirement for full category labeling. In Table 8, given the same proportion of training data, the performance gap between FCL and supervised learning is small. Even if we compare 10% FCL and 100% supervised learning, using BERT model, FCL underperforms the supervised method by 6.4 points and 3.0 points for SearchSnippets and Stackoverflow in terms of ACC, respectively. When UDA is added in addition (FCL + UDA), we can obtain higher ACC and AMI scores for both datasets. FCL underperforms the supervised method by 3.1 points and 17.2 points in ACC for 20Newsgroup and Reuters, respectively. However, for FCL + UDA, this difference reduces to 10.4 points for Reuters. That is, FCL performs very close to supervised learning, and the additional UDA shows sound effects for the short texts clustering.

When using LSTM, FCL underperforms the corresponding supervised method by 7.5 and 9.1 points in terms of ACC for SearchSnippets and Stackoverflow, respectively. This performance gap is larger than the one between BERT-FCL and supervised BERT. The performance gap is larger for long texts, suggesting a lack of pre-trained knowledge in LSTM again. It is likely that LSTM is not suitable for long text clustering also because the supervised learning method can achieve decent scores in ACC but poor scores in AMI and ARI. In contrast, LSTM-FCL increases the scores in AMI and ARI. As for UDA, it does not appear to bring any benefit over LSTM.

V. DISCUSSION

A. TUNING

1) TUNING BATCH SIZE

The batch size in our experiment was small (i.e., $2m = 160$ for short texts and $2m = 40$ for long texts) due to resource (i.e., GPUs) constraints. However, according to [4] who used

TABLE 9. Clustering ACC for 20Newsgroup under τ variations.

Method $\tau =$	0.25	0.5	0.75	1
SCL (BT)	45.3	50.1	49.6	44.6
10%FCL + UDA	61.0	62.1	56.4	50.5

TPUs, we expect that we can improve performance given the larger batch size.

2) TUNING τ

To examine the effect of the choice of the temperature parameter τ on clustering accuracy, we report on the results of an additional experiment using different τ values. Note that the purpose of this experiment is to investigate the best possible performance; as we are exploiting gold clusters to compute cluster accuracy, tuning τ in this manner is not practical. From the clustering accuracies for the 20Newsgroup under τ variations shown in Table 9, it can be seen that both 10% FCL with UDA and SCL with BT can achieve the best performance when τ is 0.5, and the accuracy decreases when either $\tau < 0.5$ or $\tau > 0.5$. Therefore, we suggest setting $\tau = 0.5$ as the default for the clustering tasks.

B. FAIRNESS OF THE COMPARISONS

As SCL is an unsupervised method, we compared it with state-of-the-art unsupervised methods. However, we acknowledge that SCL exploits BT and RM as an additional knowledge source. The main disadvantage of SCL is that, in a mini-batch, two positive pairs may be generated from two texts in the same class. However, we treat these two pairs as negative samples for each other, which limits the performance of SCL. As for FCL, we compared it with a supervised method rather than other few-shot learning methods. Note that while FCL is semi-supervised, it is different from traditional few-shot learning methods. More specifically, although we use the texts for all classes when constructing a mini-batch, the only prior knowledge needed is whether the two texts are similar or not. That is, we can transfer an n-clustering problem using n classes of labels to one using binary discriminative labels.

VI. CONCLUSION

In this paper, we proposed SCL and FCL with UDA for text clustering by using two kinds of neural structures. SCL is label-free, whereas FCL requires weak labels and outperforms SCL. We tuned and trained LSTM and BERT by our learning methods and performed clustering according to the learned latent representations. In SCL, we introduced two data augmentation methods, BT and RM. In FCL, we optionally use the UDA. Our main findings from the experiments are as follows.

- BERT-SCL outperforms the state-of-the-art in terms of ACC over Stackoverflow (short-text), 20Newsgroup and Reuters (long-text), whereas existing methods are usually valid only for short texts or only for long texts. In particular, although BERT-SCL underperforms the state-of-the-art in terms of ACC (2.0 points) over SearchSnippnets (short-text), the stability and generation ability of our methods are stronger than that of SCCL.
- LSTM-SCL performs reasonably for short-text clustering, however, does not perform well for long-text clustering.
- BERT-FCL shows that FCL can obtain performance that is very close to a supervised method, and even comparing 10%FCL with 100% supervised learning; FCL with UDA improves the performance for short texts further.
- LSTM-FCL shows a performance improvement in terms AMI and ARI, compared to that of supervised models, which suggests the advantage of FCL.

To the best of our knowledge, we are the first to successfully apply CL in an unsupervised manner to perform clustering tasks. To ensure the reproducibility of our work, we have made our code and experiment results available.

ACKNOWLEDGMENT

The authors would like to thank Dr. Zhaohao Zeng, Sijie Tao, and Junjie Wang for their feedback on their article and Jing Shen for the provision of the GPU. Haoxiang Shi would like to thank OpenAI for releasing the ChatGPT model, which has been instrumental in proofreading their work.

REFERENCES

[1] B. Chiu, S. K. Sahu, D. Thomas, N. Sengupta, and M. Mahdy, "Autoencoding keyword correlation graph for document clustering," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2020, pp. 3974–3981.

[2] A. Hadifar, L. Sterckx, T. Demeester, and C. Develder, "A self-training approach for short text clustering," in *Proc. 4th Workshop Represent. Learn. NLP (RepL4NLP)*, 2019, pp. 194–199.

[3] C. C. Aggarwal and C. Zhai, "A survey of text clustering algorithms," in *Mining Text Data*, 2012, pp. 77–128.

[4] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, Nov. 2020, pp. 1597–1607.

[5] Q. Xie, Z. Dai, E. Hovy, T. Luong, and Q. Le, "Unsupervised data augmentation for consistency training," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 6256–6268.

[6] A. Graves, "Long short-term memory," in *Supervised Sequence Labelling With Recurrent Neural Networks*. Amsterdam, The Netherlands: Elsevier, 2012, pp. 37–45.

[7] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi, "Learning to classify short and sparse text & web with hidden topics from large-scale data collections," in *Proc. 17th Int. Conf. World Wide Web*, Apr. 2008, pp. 91–100.

[8] J. Xu, B. Xu, P. Wang, S. Zheng, G. Tian, J. Zhao, and B. Xu, "Self-taught convolutional neural networks for short text clustering," *Neural Netw.*, vol. 88, pp. 22–31, Apr. 2017.

[9] K. Lang, "Newsweeder: Learning to filter netnews," in *Machine Learning Proceedings*. San Mateo, CA, USA: Morgan Kaufmann, 1995, pp. 331–339.

[10] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *J. Mach. Learn. Res.*, vol. 5, pp. 361–397, Dec. 2004.

[11] D. Zhang, F. Nan, X. Wei, S. Li, H. Zhu, K. McKeown, R. Nallapati, A. Arnold, and B. Xiang, "Supporting clustering with contrastive learning," 2021, *arXiv:2103.12953*.

[12] H. Shi, C. Wang, and T. Sakai, "A simple and effective usage of self-supervised contrastive learning for text clustering," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2021, pp. 315–320.

[13] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jun. 2006, pp. 1735–1742.
[14] N. Inoue and K. Goto, "Semi-supervised contrastive learning with generalized contrastive loss and its application to speaker recognition," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2020, pp. 1641–1646.
[15] K. Hassani and A. H. Khasahmadi, "Contrastive multi-view representation learning on graphs," in *Proc. Int. Conf. Mach. Learn.*, Nov. 2020, pp. 4116–4126.
[16] R. C. Staudemeyer and E. R. Morris, "Understanding LSTM—A tutorial into long short-term memory recurrent neural networks," 2019, *arXiv:1909.09586*.
[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6000–6010.
[18] T. Alireza. *[Re] Autoencoding Keyword Correlation Graph for Document Clustering*. Accessed: Mar. 2020. [Online]. Available: https://github.com/1997alireza/Autoencoding-Graph-for-Document-Clustering/blob/main/docs/Report.pdf
[19] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.
[20] B. Gunel, J. Du, A. Conneau, and V. Stoyanov, "Supervised contrastive learning for pre-trained language model fine-tuning," 2020, *arXiv:2011.01403*.
[21] H. Fang, S. Wang, M. Zhou, J. Ding, and P. Xie, "CERT: Contrastive self-supervised learning for language understanding," 2020, *arXiv:2005.12766*.
[22] T. Li, X. Chen, S. Zhang, Z. Dong, and K. Keutzer, "Cross-domain sentiment classification with contrastive learning and mutual information maximization," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 8203–8207.
[23] H. Wu, T. Ma, L. Wu, T. Manyumwa, and S. Ji, "Unsupervised reference-free summary quality evaluation via contrastive learning," 2020, *arXiv:2010.01781*.
[24] L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. Bennett, J. Ahmed, and A. Overwijk, "Approximate nearest neighbor negative contrastive learning for dense text retrieval," 2020, *arXiv:2007.00808*.
[25] T. Yuan, W. Deng, J. Hu, Z. An, and Y. Tang, "Unsupervised adaptive hashing based on feature clustering," *Neurocomputing*, vol. 323, pp. 373–382, Jan. 2019.
[26] X. Li, H. Yin, K. Zhou, and X. Zhou, "Semi-supervised clustering with deep metric learning and graph embedding," *World Wide Web*, vol. 23, no. 2, pp. 781–798, Mar. 2020.
[27] R. Wang, X. Hu, D. Zhou, Y. He, Y. Xiong, C. Ye, and H. Xu, "Neural topic modeling with bidirectional adversarial training," 2020, *arXiv:2004.12331*.
[28] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.
[29] A. Shukla, G. S. Cheema, and S. Anand, "Semi-supervised clustering with neural networks," in *Proc. IEEE 6th Int. Conf. Multimedia Big Data (BigMM)*, Sep. 2020, pp. 152–161.
[30] L. A. Vilhagra, E. R. Fernandes, and B. M. Nogueira, "TextCSN: A semi-supervised approach for text clustering using pairwise constraints and convolutional Siamese network," in *Proc. 35th Annu. ACM Symp. Appl. Comput.*, Mar. 2020, pp. 1135–1142.
[31] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Is a correction for chance necessary?" in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, Jun. 2009, pp. 1073–1080.
[32] Z. Wang, H. Mi, and A. Ittycheriah, "Semi-supervised clustering for short text via deep representation learning," 2016, *arXiv:1602.06797*.
[33] Y. Qian, Q. Zheng, T. Sakai, J. Ye, and J. Liu, "Dynamic author name disambiguation for growing digital libraries," *Inf. Retr. J.*, vol. 18, no. 5, pp. 379–412, Oct. 2015.
[34] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong, "Towards K-means-friendly spaces: Simultaneous deep learning and clustering," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2017, pp. 3861–3870.
[35] F. Role, S. Morbieu, and M. Nadif, "CoClust: A Python package for co-clustering," *J. Stat. Softw.*, vol. 88, no. 7, pp. 1–29, 2019.

HAOXIANG SHI is currently pursuing the Ph.D. degree in computer science with Waseda University, Tokyo, Japan.

His research interests include natural language processing and dense retrieval, including parameter efficiency, contrastive learning, and data augmentation.

TETSUYA SAKAI is currently a Professor with the Department of Computer Science and Engineering, Waseda University, Japan. He is also a General Research Advisor with Naver Corporation, South Korea, and a Visiting Professor with the National Institute of Informatics, Japan. His research interests include information retrieval, natural language processing, and human–computer interaction (HCI).

He is also an ACM Distinguished Member, a Senior Associate Editor of *ACM TOIS*, and a fellow of IPSJ. In 2023, he was inducted into the SIGIR Academy.