Grouped Contrastive Learning of Self-supervised Sentence Representation

Qian Wang , Weiqi Zhang , Tianyi Lei , Dezhong Peng *

Posted Date: 26 July 2023

doi: 10.20944/preprints202307.1752.v1

Keywords: Contrastive Learning; Self-attention; Data Augmentation; Grouped Representation; Unsupervised Learning

Article

Grouped Contrastive Learning of Self-Supervised Sentence Representation

Qian Wang [1],Weiqi Zhang [1], Tianyi Lei [1], Dezhong Peng [1,*]

[1] College of Computer Science, Sichuan University, Chengdu
* Correspondence: pengdz@scu.edu.cn

Abstract: This paper proposes a Grouped Contrastive Learning of self-supervised Sentence Representation (GCLSR), which can learn an effective and meaningful representation of sentences. Previous works maximize the similarity between two vectors to be the objective of contrastive learning, suffering from the high-dimensionality of the vectors. In addition, most previous works have adopted discrete data augmentation to obtain positive samples and directly employed contrastive framework of computer vision to perform contrastive training, which could hamper contrastive training because text data is discrete and sparse compared with image data. To address those issues, we propose a grouped contrastive learning framework, i.e., GCLSR, which divides the high-dimensional feature vector into several groups and respectively computes the groups' contrastive losses to make use of more local information, eventually obtaining a more fine-grained sentence representation. In addition, in GCLSR, we design a new self-attention mechanism and a continuous as well as partial word vector augmentation (PWVA). For the discrete and sparse text data, the usage of self-attention could help model focus the informative words by measuring the importance of every word in a sentence. By using the PWVA, GCLSR can obtain high-quality positive samples used for contrastive learning. Experimental results demonstrate that our proposed GCLSR achieves an encouraging result on the challenging datasets of the standard semantic textual similarity (STS) task and transfer task.

Keywords: contrastive learning; self-attention; data augmentation; grouped representation; unsupervised Learning

1. Introduction

Representation learning of sentences plays a key role in most tasks in natural language processing (NLP). Good sentence representation not only improves the performance of downstream tasks but also benefits the steady training of model [1–5].

Recently, researchers have achieved great advances in sentence representation based on contrastive learning with pre-trained language models [6–10]. On the one hand, the large-scale pre-trained language models (PLMs), typified by BERT [11], are trained with unlabeled data, improving the state-of-the-art results in most downstream tasks. Therefore, PLMs are applied to various real scenarios, such as Text Generation [8], Name Entity Recognition [12], Question Answering [13], Translation [13]. On the other hand, unsupervised representation learning based on contrastive learning advances the development of computer vision [14–17]. Therefore, many researchers combine PLMs and contrastive learning to finish the task of sentence representation [18,19]. For example, Wu *et al.* [20] adopt back-translate as the data augmentation to produce positive samples used for contrastive learning and PLMs as the backbone to obtain semantic feature of sentences, achieving a promising result on sentence representation. Gao *et al.* [21] respectively take the standard dropout mask of Transformer, cosine similarity as the data augmentation and the contrastive objective function to finish the contrastive training, producing a meaningful sentence representation.

However, there are issues with implementing contrastive learning in sentence representation. **a)** An appropriate data augmentation is needed to produce positive samples used for contrastive learning. In contrastive training, the semantic gap between the positive example pair should be narrow.

 doi:10.20944/preprints202307.1752.v1

Therefore, improper data augmentation may change semantic information of sentences, resulting in a difficulty of advancing performance. **b)** The information of text data is sparse and discrete. Unlike image data, the information between the adjacent pixels is continuous while the information of text data is discrete, indicating that the model could not learn the distinguishing features by contrastive learning. **c)** Similarity computing between high-dimensional vectors could lose local information of vectors. Generally, the objective of contrastive learning is to minimize the similarity of high-dimensional vectors, which could not make use of local information of vectors well and affect performance.

To solve the issues above, we propose the Grouped Contrastive Learning of self-supervised Sentence (GCLSR). GCLSR adopts continuous data and partial data augmentation to obtain high-quality positive samples used for contrastive learning. Due to the discrete and sparse text data, GCLSR designs a self-attention mechanism to focus on the informative words by measuring the importance of every word in a sentence. To address high-dimensional feature vectors, GCLSR proposes grouped contrastive learning to disentangle more local information of feature vectors.

The contributions of this paper are summarized as follows:

- We propose a new data augmentation named partial word vector augmentation (PWVA) to obtain positive samples used for contrastive learning. PWVA performs data augmentation on partial word vectors of word embedding space of a sentence. In this way, the positive sample pairs can retain more original semantic information, which could enhance and facilitate contrastive learning.
- We design a new computation method of self-attention to help model focus on the informative words of a sentence. Experimental results show that the usage of self-attention can enhance the representation for discrete and sparse text data.
- We design a new paradigm of contrastive learning——the grouped contrastive learning for sentence representation (GCLSR), which can make use of more local information of high-dimensional feature vectors.
- We conduct extensive experiments on the different datasets to evaluate GCLSR. Experimental results demonstrate that our proposed GCLSR achieves a promising result on sentence representation. In addition, we carry out experiments to further study the GCLSR, for example, ablation study and the possible implementation schemes based on our method.

The rest of this paper is organized as follows: The related works on representation learning based on contrastive learning, text data augmentation, and self-attention will be shown in Section 2. Then, we will introduce our approach GCLSR and evaluate it in Sections 3 and 4 respectively. Section 5 will investigate GCLSR further. Finally, the part of the conclusion and future work will be shown in Section 6.

2. Related Work

2.1. Representation learning Based on Contrastive Learning

Contrastive learning currently achieves surprising results in representation learning [14,17,22]. Generally, the implementation of contrastive learning is based on Siamese networks: an effective framework to learn the discriminative information between the positive and negative sample pairs.

In the computer vision domain, contrastive learning achieves significant improvement on the representation of image. SimCLR [14] uses an encoder and projection head as the contrastive framework, which advances the state-of-the-art result on image representation. BYOL [17] designs a Momentum encoder to avoid collapsing on contrastive training, which obtains an encouraging result. SimSiam [16] adopts the structure of BYOL but it discards the Momentum encoder. Surprisingly, using only stop-gradient can achieve competitive performance.

Due to the success of contrastive learning in computer vision, in the NLP, researchers have employed contrastive learning to obtain the representation of sentences. CERT [23] augments a

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

sentence by back-translation and performs contrastive training using the contrastive framework of MoCo. CMV-BERT [24] adopts different tokenizers to conduct sentence augmentation and performs contrastive training on the framework of the SimSiam. ConSERT [25] performs data augmentation (such as Token Shuffling, Adversarial Attack, Cutoff and Dropout) on word embedding layer of BERT to obtain positive samples and achieve encouraging results. More details about contrastive learning in sentence representation are shown in Table 1.

Table 1. Some contrastive learning methods in sentence representation. P: the computing of loss just includes positive samples. P+N: the computing of loss includes positive and negative samples, i.e., $\ell = -log\frac{exp(sim(z_i,z_j)/\tau)}{\sum_{k=1}^{2N} exp(sim(z_i,z_k)/\tau)}(k \neq i)$, where z_i, z_j,and z_k are the positive samples while the z_k represents the negative samples. τ is a temperature parameter. $sim(\cdot)$ denotes the similarity. Discrete/Continuous and Full: data augmentation is discrete/Continuous and performed on every word of a sentence.

Model	Backbone	Data Augmentation	Loss	Framework
CERT [23]	Pre-trained BERT	Back-translation (Discrete and Full)	P+N	Based on MoCo
CMV-BERT [24]	ALBERT(3 layers)	Multi-tokenizers (Discrete and Full)	P	Based on Simsiam
CLEAR [20]	Transformer	Substitution (Discrete and Full)	P+N	Based on SimCLR
ConSERT [25]	Pre-trained BERT	Dropout (Continuous and Full)	P+N	Based on SimCLR
SimCSE [21]	Pre-trained BERT	Dropout (Continuous and Full)	P+N	Based on SimCLR

While great successes are achieved by contrastive learning in sentence representation, deficiencies still are existed in aforementioned methods, which hampers the improvement of performance. The details are shown below: (1) Most methods directly utilize the framework of computer vision as the pipeline of contrastive learning. Therefore, it could hamper the contrastive training because text data is discrete and sparse compared with image data. (2) The well-performed pre-trained models (such as BERT) are adopted as the backbone network of contrastive learning. Consequently, it could not evaluate the performance of a lightweight model on sentence representation using contrastive learning. After all, the pre-trained model can work well in NLP tasks. (3) Improper data augmentation could change the original semantics of a sentence. Most methods use discrete data augmentation to produce positive samples to perform contrastive training, which could deteriorate the original semantic totally. Different from the aforementioned methods, we design dedicated text data augmentation PWVA and contrastive framework for contrastive learning and use lightweight model Textcnn to explore the effectiveness of contrastive learning on sentence representation. The details of our proposed model (GCLSR) are shown in Figure 1.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

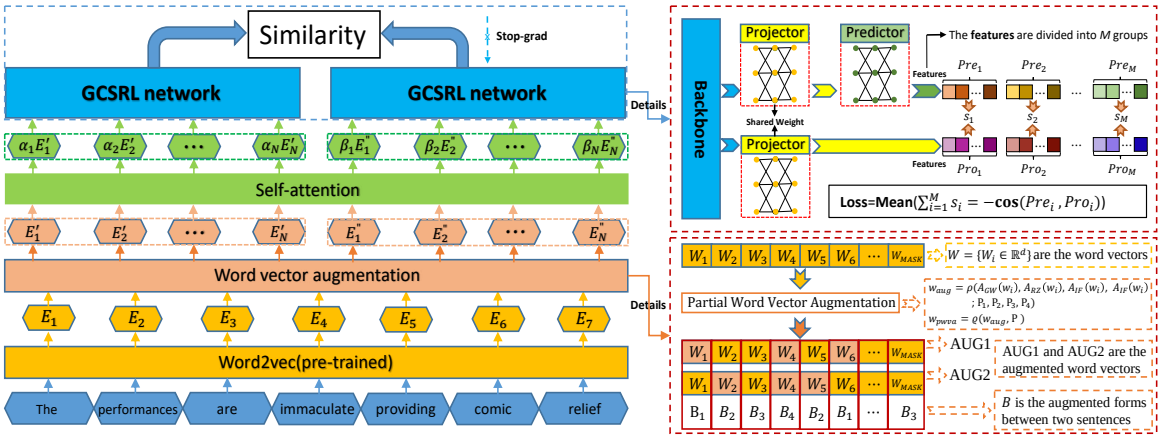


Figure 1. The GCLSR architecture.The GCLSR contains three parts: partial word vector augmentation, self-attention and GCLSR network. The upper right plot is the details of the GCLSR network. The lower right plot is the visualization of PWVA. Specifically, a sentence W is performed the data augmentation twice on word vector space with PWVA, obtaining positive sample pairs AUG1 and AUG2. In AUG1 and AUG2, the orange boxes denote the augmented word vectors while the yellow boxes mean that the word vector does not be augmented. Therefore, there are four combination forms, i.e., B1, B2, B3, and B4, between AUG1 and AUG2. Specifically, the B1 represents the form that the word vector W1 of AUG1 is augmented while AUG2's W1 does not be augmented. The "partial" is reflected that there are some word vectors are not augmented in AUG1 and AUG2, which can retain more original semantics used for contrastive learning.

2.2. Text Data Augmentation

Data augmentation is an effective strategy to improve performance and steadiness of training. Wei *et al.* [26] proposes a popular data augmentation named EDA for text classification and achieves promising results. Wang *et al.* [27] search for the k-nearest-neighbor words on word embedding space to obtain positive samples. Guo *et al.* [28] conducts a linear interpolation between the pair of embedding vectors to perform data augmentation.

While many data augmentation obtain encouraging results in NLP, there is no dedicated one for contrastive training. Generally, the positive samples used for contrastive training are produced by data augmentation. Therefore, it could be improper if the methods above directly are applied to data augmentation for contrastive training. The reasons are shown below: (1) The semantic gap between the augmented sentences should be narrow. In contrast, the contrastive model could be collapsing easily. (2) The data augmentation can be performed on the partial data. Partial data augmentation can retain more original semantics, which could help model learn distinguishing features easily. (3) The augmentation of text is as continuous as possible. Most methods above are discrete, such as EDA, Back-translation, which may make the contrastive training unsteady and hurt the generalization.

2.3. Self-attention in Language Model

Great progress has been made in the development of attention, since Bahdanau *et al.* [29] adopt attention to enhance the performance of NLP tasks. Devlin *et al.* [11] design an encoder with attention to process sentences and achieve great performance on the various tasks of NLP. However, it needs additional operations (such as the position-wise Feed-Forward Networks and LayerNorm) to ensure steady training, resulting in a difficulty of applying to a practical lightweight computational platform. Therefore, we design a self-attention mechanism with low computing consumption to compute the importance of words in a sentence. In this way, the usage of self-attention on contrastive learning can help a model focus on the informative words of a sentence to solve the issue——text data is discrete and sparse. The details of our proposed method of self-attention are shown in Section 3.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

3. Methodology

As discussed in Section 2, the keys of contrastive learning lie in performing data augmentation to produce positive samples and designing a contrastive framework. In this paper, we propose a grouped contrastive learning for sentence representation (GCLSR). GCLSR contains three parts: **a)** partial word vector augmentation (introduced in Section 3.1), **b)** self-attention of data (introduced in Section 3.2) and **c)** Grouped contrastive learning (introduced in Section 3.3). Figure 1 illustrates the overall architecture and training pipeline of GCLSR.

3.1. Partial Word Vector Augmentation

As discussed above, the aim of performing data augmentation in contrastive learning is obtaining positive samples. However, most existing methods are discrete and performed on full words of a sentence, which could deteriorate original semantic information for discrete and sparse text data. Therefore, we design a continuous and Partial Word Vector Augmentation (PWVA) for contrastive learning. Furthermore, a word vector is a vector with fixed dimensionality and every element in word vector is a real value. Therefore, a word vector can be treated as 1D discrete signal. In this way, word vectors can be processed by strategies of digital signal processing. Our proposed PWVA is based on this insight to implement data augmentation. Specifically, we perform the PWVA by two choices of probability. Let $W = \{w_i \in \mathbb{R}^d\}_{i=1}^N$ be the N word vectors with d dimensionality. The first probability choice of PWVA can be written by:

$$w_{aug} = \rho(A_{gwn}(w_i), A_{rzs}(w_i), A_{ifft}(w_i), A_{rbn}(w_i); p1, p2, p3, p4), \tag{1}$$

where $\rho(\cdot)$ is a function to select augmented methods from GWN, RZS, IFFT and RBN with probabilities $p1$, $p2$, $p3$, $p4$, respectively. The w_{aug} is the augmented word vectors. The second choice of PWVA can be written as follows:

$$w_{pwva} = \varrho(w_{aug}, w_i; p), \tag{2}$$

where ϱ is a function to select final PWVA's output w_{pwva} from w_{aug} and w_i with probability p. The visualization of PWVA is shown in the lower right plot of Figure 1. In addition, four data augmentation methods used in Eq.(1) are represented as follows.

- **Gaussian White Noise (GWN)**

  We add the Gaussian white noise (shown in Figure 2(a)) to the word vectors, which can make the model more robust [30]. The Gaussian white noise can be written as follows:

  $$A_{gwn}(w_i) = w_i + \lambda \cdot \mathcal{N}(0, 1), \tag{3}$$

  where λ and $\mathcal{N}(0, 1)$ are the trade-off parameter and standard normal distribution respectively.
- **Random Zero Setting (RZS)**

  We make use of the random zero setting $A_{rzs}(w_i) = Dropout(w_i)$ (shown in Figure 2(b)) to let some values of word vector be zero to prevent data dependence and improve the generalization ability.
- **Inverse Fast Fourier Transformation (IFFT)**

  Word vectors of a sentence include statistical information (such as word frequencies, and contextual information). Therefore, we perform the FFT on word vectors to extract features in the frequency domain, and then the IFFT (shown in Figure 2(c)) to transfer it into the time domain. The word vectors after performing IFFT have slight differences compared with the original word vectors, advancing the fault tolerance of the data boundary. The IFFT can be represented as follows:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

$$A_{ifft}(w_i) = Real(IFFT(FFT(w_i))), \tag{4}$$

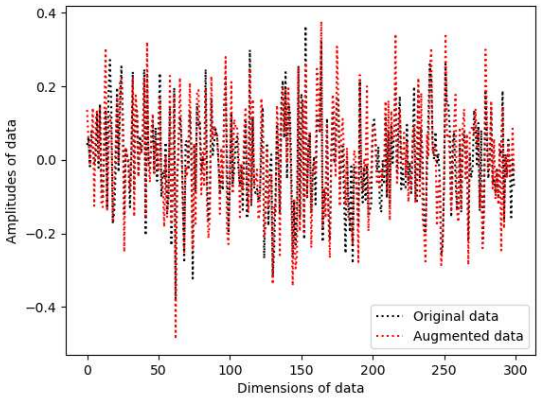
where $Real(\cdot)$ denotes the real part.

- **Random Background Noise (RBN)**

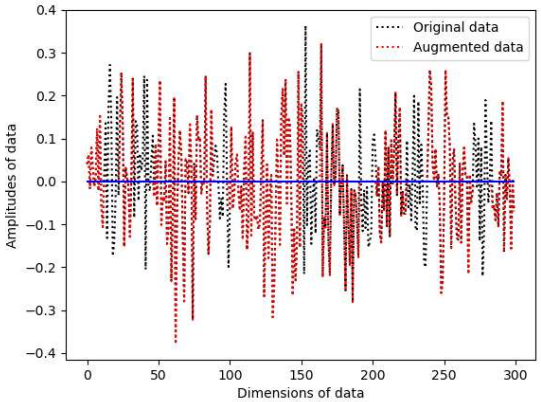
  Random background noise can not be learned by a model [31]. Therefore, we add the random background noise (shown in Figure 2(d)) into the word vectors to prevent collapse and improve the training stability. The RBN can be written as follows:

$$A_{rbn}(w_i) = w_i + uniform(0, 0.1), \tag{5}$$

where $uniform(\cdot)$ is the uniform distribution.



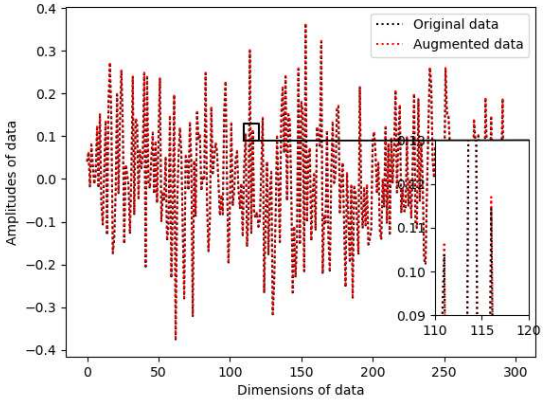
(a) **Gaussian White Noise:** Amplitudes of noise added to the word vectors follow the standard normal distribution.



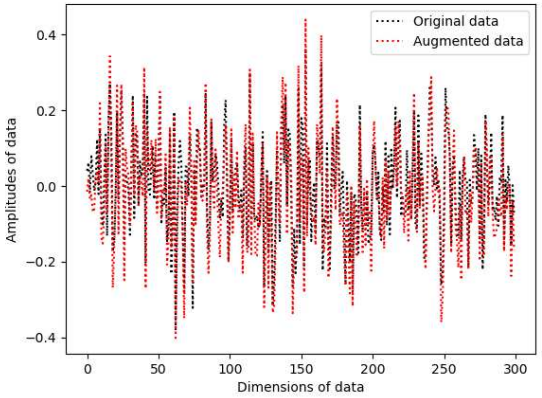
(b) **Random Zero Setting:** The data corresponding to black curve is set to zero.

Figure 2. *Cont.*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1



(a) **Inverse Fast Fourier Transformation:** Differences can be observed from the data boundary by the right zoom window.



(b) **Random Background Noise:** Amplitudes of noise added to the word vectors follow uniform distribution.

Figure 2. Four word vectors augmentation methods of the **PWVA**. Note that we perform data augmentation on word vectors of a sentence rather than original sentence.

In summary, continuous and partial PWVA can enhance and facilitate contrative learning. The characteristic of “continuous" of PWVA can ensure that there is no semantic gap in word vectors while the “partial" can retain more original semantics of word vectors. In this way, a model can easily learn more distinguishing features by learning the differences between the original word vectors and corresponding augmented word vectors. In contrast, all existing methods obtain distinguishing features between augmented data, resulting in a difficulty of contrastive training. This insight is the main contribution of PWVA. The ablation study will be shown in Section 5.

3.2. Self-attention of the Word Vectors

Self-attention is applied to many scenarios and achieves great success. Inspired by the work [11], we design a dedicated self-attention to help the model focus on informative word vectors from the discrete and sparse text data. Note that the word vectors are produced by the pre-trained word2vec [32] before carrying out data augmentations. Hence, the word vectors already include some semantic information. Furthermore, the self-attention added to the word vectors can make the model focus on the features useful for distinguishing semantic information. The details are shown in Figure 3. Note that our method of self-attention is different from BERT’s. The main differences are shown as follows: (1) We fill the value $1e-9$ after computing the scores for padding tokens. (2) We first compute the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

importance of the words to a sentence, and then make it multiplied by the original word vector. The differences are shown in the last three rows of the Algorithm 1.

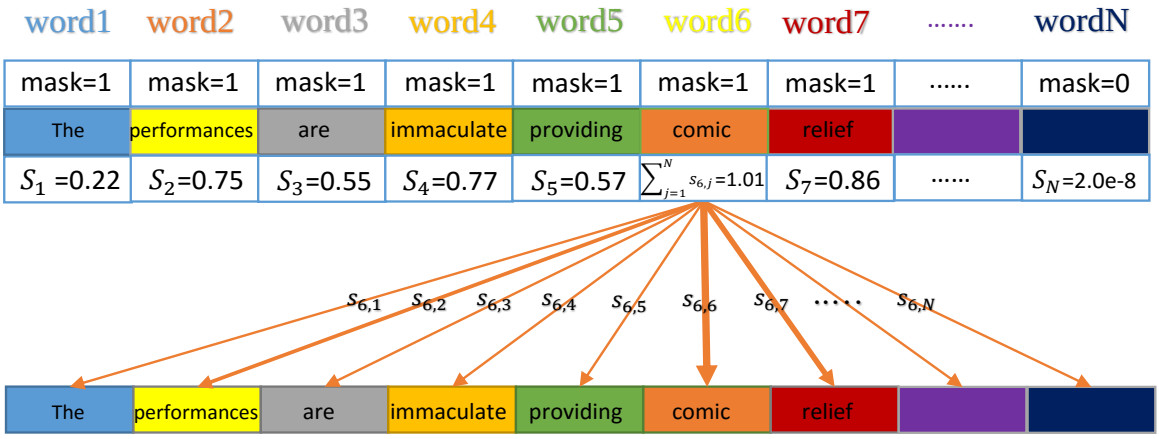


Figure 3. The self-attention of word vectors. Let $N =$ be the number of words in a sentence. $MASK$, $s_{i,j} = x_i * x_j$ are the mask matrix (that the value equals 0 if the word is the Padding token) and attention of the word x_i to x_j in a sentence respectively. Specially, $S = \sum_{j=1}^{N} s_{i,j}$ $(i = 1, 2, \cdots, N)$ can represent the importance of the word x_i to a sentence. We can observe from Figure 3 that the first few words with a larger value of "importance" are these words: "comic", "relief", "performances" and "immaculate", which can help the model to watch the crucial information of a sentence.

Algorithm 1: self-attention of word vectors, PyTorch-like

```
# N=word_num, query=key=value
def data_self-attention(query,key,value,mask):
    mask = mask.unsqueeze(2)
    mask = torch.matmul(mask, mask.transpose(-2, -1))
    d_k = query.size(-1)
    #compute the attentions
    scores = torch.matmul(query, key.transpose(-2, -1))
    scores=scores/math.sqrt(d_k ) #[bath_size, N, N]
    if mask is not None:
        # fill the scores with 1e-9 if mask=0
        scores =scores.masked_fill(mask == 0, 1e-9)
    #compute the importance of a word to a sentence
    scores=torch.sum(scores,dim=2,keepdim=True)
    return scores*value
```

3.3. Grouped Contrastive Learning

Generally, the pipelines of contrastive learning are that one first performs data augmentation to produce positive samples, then obtains the features computed by the backbone, and finally computes the contrastive loss [16]. Unfortunately, computing a contrastive loss between high-dimensional vectors could not make use of the local information of vectors well. To solve this issue, we propose the GCLSR to mitigate the aforementioned drawback during contrastive training. As shown in Figure 1, the GCLSR consists of two branches. The first branch includes the Backbone, Projector[14]] and Predictor[17] while the other is the backbone, and projector. Specially, in order to make use of local information of features, we first divide the features of the Projector and Predictor into M groups with D dimensionality. The grouped features of the Projector and Predictor can be denoted as $Fea_{Pro} = \{Pro_i \in \mathbb{R}^D\}_{i=1}^{M}$ and $Fea_{Pre} = \{Pre_i \in \mathbb{R}^D\}_{i=1}^{M}$ respectively. Finally, we use the negative mean of the cosine similarity as the contrastive loss [17]:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

$$\ell = -Mean(\sum_{i=1}^{M}(\frac{Pre_i \cdot Pro_i}{\|Pre_i \cdot Pro_i\|_2})) \tag{6}$$

where $\|\cdot\|$ is l_2-norm. In addition, we adopt the symmetrical loss to improve performance.

$$\ell_{sym} = -\frac{1}{2}Mean(\sum_{i=1}^{M}(\frac{Pre_i \cdot Pro_i}{\|Pre_i \cdot Pro_i\|_2} + \frac{Pro_i \cdot Pre_i}{\|Pre_i \cdot Pro_i\|_2}) \tag{7}$$

4. Experiments

We evaluate our proposed method GCLSR on 7 semantic textual similarity (STS) tasks. In addition, we also conduct experiments on 7 transfer tasks. Note that we use a lightweight model, i.e., Textcnn, as the backbone network to explore the effectiveness of contrastive learning on sentence representation. Therefore, we do not take the state-of-the-art results as comparison objects. In addition, STS experiments as well as transfer tasks are fully unsupervised and no STS data sets (train, test, validation) are used during the training phase.

4.1. Implementation Settings

Unless specified, we use the following settings for contrastive self-supervised pre-training:

- **Backbone**. We use textcnn [33] as the default backbone. Specifically, the filter region size is [1,1,1,6,15,20]. The number of filters is 300. Note that we do not use a Fully Connected (FC) layer and dropout at the end of the backbone, because it makes the results worse.
- **Projector**. The projection layers include 3 FC layers. Every output of FC layer has a batch normalization [34] and ReLU (except for the last FC layer). The dimensions of the hidden and output layers are 4096.
- **Predictor**. The prediction layers have 2 FC layers. The hidden layers have a Batch Normalization (BN) and ReLU while the output layers don't have BN and ReLU. The dimensions of hidden as well as output layers are 1024 and 4096 respectively, producing a bottleneck structure and improving the robustness of model [16].
- **Optimizer**. The SGD is used for the optimizer. The Learning Rate (LR) is $base\_lr * BatchSize/128$ (the $base\_lr$ is 0.03). The LR has a cosine decay schedule [35]. The Weight Decay is 0.001. We also use the Warm-up (5 epochs). Besides, the Momentum is 0.9 before Warm-up epochs and 0.8 after Warm-up epochs respectively, which makes the model more robust (more details are shown in Section 5).

4.2. Semantic Textual Similarity Task

We train our self-supervised model GCLSR with pre-trained word2vec on 10^4 sampled sentences from English Wikipedia randomly [21]. The stop epochs are 20 and the best checkpoint on validation datasets is used for testing. Finally, we use SentEval toolkit[36] to evaluate our proposed method on 7 STS tasks, i.e., STS 2012-2016[37–41], STS Benchmark[42] and SICK-Relatedness[43]. Following [16], (a) we only use Spearman correlation to measure the quality of sentence representation for all STS tasks. [16] argue that Spearman correlation suits the need of evaluating better; (b) No additional networks are applied on the top of sentence representation. In other words, we directly compute the Spearman correlation with cosine similarities; (c) Given that STS data of every year include several sub-datasets, we concatenate all sub-datasets to compute the Spearman correlation. The operation "concatenate" incorporating different subsets is more proper compared with other methods in practical application.

The results of the evaluation are shown in Table 2. We observe that all variations we proposed can work well and are better than Word2vec embeddings (avg.). Specifically, we improve the averaged Spearman's correlation from 44.16 to 66.75 compared with Word2vec embeddings (avg.), pre-trained language model BERT (from 56.57 to 66.75), RoBERT (from 56.57 to 66.75), and CLEAR

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

(from 61.8 to 66.75, i.e., 8.01). Furthermore, our proposed data augmentation PWVA outperforms the compared method EDA and Back-translation on STS tasks on the proposed contrastive learning framework GCLSR$_{base}$. The applications of the self-attention (GCLSR$_{base}$+PWVA+self-att.) and grouping (GCLSR$_{base}$+ PWVA+self-att.+GP) can also improve the performance. The experimental results show that our proposed method GCLSR$_{base}$+PWVA+self-att.+GP can also achieve a better result compared with GCLSR$_{base}$+EDA+self-att.+GP and GCLSR$_{base}$+TransL.+self-att.+GP. In addition, we observe that good representation in the STS tasks does not necessarily obtain better performance in the transfer tasks. Therefore, the STS results should be taken as the main comparison.

Table 2. The evaluation of sentence representation on STS tasks. All results are computed with the Spearman's correlation. *: results from[21]; **: results from[20]; The remaining results are evaluated by ourselves. GCLSR$_{base}$ means that the model only consists of a backbone, Projector and Predictor. The model receives two same word vectors as the input, i.e., no data augmentation is used. EDA is a text data augmentation method proposed by [26]. TransL denotes the data augmentation back translation. Self-att as well as GP denote self-attention mechanism and feature grouping respectively.

Model	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
Word2vec embeddings(avg.)	33.75	43.20	36.95	55.23	54.85	36.24	48.90	44.16
BERT$_{base}$(first-last avg.)*	39.70	59.38	49.67	66.03	66.19	53.87	62.06	56.70
RoBERT$_{base}$(first-last avg.)*	40.88	58.74	49.07	65.63	61.48	58.55	61.63	56.57
CLEAR**	49.00	48.90	57.40	63.60	65.60	**75.60**	**72.50**	61.80
BERT$_{base}$-flow*	**58.40**	**67.10**	60.85	**75.16**	71.22	68.66	64.47	**66.55**
BERT$_{base}$-whitening*	57.83	66.90	**60.90**	75.08	**71.31**	68.24	63.73	66.28
GCLSR$_{base}$	57.47	68.70	64.03	72.84	67.90	65.64	59.55	65.16
GCLSR$_{base}$+EDA	57.33	68.09	63.65	72.01	66.63	65.34	59.71	64.68
GCLSR$_{base}$+TransL	**60.53**	66.09	63.50	72.83	67.39	66.09	59.78	65.17
GCLSR$_{base}$+PWVA	58.92	67.94	64.41	73.54	68.72	66.16	**59.85**	65.65
GCLSR$_{base}$+PWVA+self-att.	57.62	71.00	65.83	75.51	69.81	67.41	59.41	66.66
GCLSR$_{base}$+EDA+self-att.+GP	58.13	68.85	64.14	73.40	66.92	65.31	59.61	65.19
GCLSR$_{base}$+TransL.+self-att.+GP	58.80	68.00	64.70	73.74	68.50	67.24	59.67	65.81
GCLSR$_{base}$+PWVA+self-att.+GP	57.81	**71.01**	**65.83**	**75.62**	**70.01**	**67.58**	59.34	**66.75**

4.3. Transfer Task

Our proposed model is tested on the following tasks:MR[44], CR[45], SUBJ[46], MPQA[47], SST-2[48], TREC[49] and MRPC[50]. Note that we only add a linear layer on the frozen backbone to evaluate sentence representation. The pre-trained stage is as same as STS tasks. The evaluation results are shown in Table 3. We find that the overall tendency of results is as same as STS tasks. However, there are two abnormalities needed to be explained. (1) The pre-trained model BERT$_{base}$ obtains a better result compared with our proposed model on transfer tasks. Firstly, we only choose 10^4 sentence of the wiki to perform the pre-training. Secondly, the number of parameters of our model is far less than BERT$_{base}$. Therefore, we just take a little time to perform pre-training (about one hour on 1 Tesla v100 GPU). (2) The application of Self-attention and grouping harm the performance slightly compared with GCLSR$_{base}$+PWVA. A possible explanation is that the implementation of PWVA is on word vectors, which could change original semantic information of vectors. In addition, we do not perform the joint training with tasks, which makes the model could not digest the learned contrastive features.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

Table 3. The results of transfer tasks. All results are computed with the Spearman's correlation. *: results from[21]; **: results from[20]; The remaining results are evaluated by ourselves.

Model	MR	CR	SUBJ	MPQA	SST-2	TREC	MRPC	Avg.
Word2vec embeddings(avg.)	75.91	77.56	89.31	87.18	80.89	77.40	**72.17**	80.06
$BERT_{base}$(first-last avg.)*	**78.66**	**86.25**	**94.37**	**88.66**	**84.40**	**92.80**	69.54	**84.94**
$GCLSR_{base}$	76.78	79.02	90.21	88.35	81.77	83.00	73.28	81.77
$GCLSR_{base}$+EDA	76.56	79.55	90.54	88.54	81.66	84.40	72.99	82.03
$GCLSR_{base}$+TransL	76.61	79.52	90.41	88.74	81.27	84.00	73.04	81.94
$GCLSR_{base}$+PWVA	76.76	**80.08**	90.66	88.59	81.60	85.20	73.57	**83.35**
$GCLSR_{base}$+PWVA+self-att.	76.97	78.81	**90.98**	88.36	80.51	84.60	**73.74**	82.00
$GCLSR_{base}$+EDA+self-att.+GP	76.90	79.58	90.57	88.50	**81.82**	**85.60**	72.75	82.25
$GCLSR_{base}$+TransL.+self-att.+GP	76.98	79.32	90.41	88.54	81.16	84.00	73.28	81.96
$GCLSR_{base}$+PWVA+self-att.+GP	**77.69**	79.87	90.89	**88.99**	81.44	83.80	73.39	82.30

5. Further Investigation of GCLSR

We propose a new paradigm for contrastive learning GCLSR consisting of three major crucial components, i.e., (a) data augmentation, (b) self-attention, (c) grouped contrastive learning, to study the effectiveness of contrastive learning on the sentence representation. Experimental results show that our proposed GCLSR achieves a promising result. However, some experiment settings of GCLSR influence the performance of sentence representation, such as Warm-up, Weight Decay, etc. Therefore, we conduct ablation experiments to study it further. All experiments are conducted in STS 2012-2015.

5.1. Effect of the Batch Size

Given that big batch size could impact the performance shown in previous works [14], we conduct an ablation experiment to study it. Table 4 shows the comparison results of batch size from 64 to 4096. We use the same linear scaling rule–$base\_lr * Batch\_size * 128$(the $base\_lr$ is 0.3) for all experiments.

Table 4. The effect of batch size.

Batch size/STS	STS12	STS13	STS14	STS15	Avg.
64	**56.35**	72.56	66.59	74.73	67.56
128	56.20	**72.77**	66.72	75.03	**67.68**
256	55.94	72.66	66.67	75.22	67.62
512(ours)	55.79	72.72	**66.73**	**75.40**	67.66
1024	55.73	72.44	66.45	**75.40**	67.51

Table 4 reports the results of batch size from 64 to 1024. Different from previous conclusions, our model is insensitive to batch size. On the contrary, the performance is worse when the batch size increase to 1024 compared with the batch size of 512. In addition, a small batch size of 64 also achieves competitive performance. A reasonable explanation is that the computing of contrastive loss does not include negative examples.

5.2. Effect of the Weight Decay

We find that the value of Weight Decay will influence performance dramatically. We conjecture that the perturbation of model weight can influence contrastive self-supervised training. Therefore, we perform an experiment to investigate it. The results are shown in Table 5.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

Table 5. The effect of the weight decay.

Weight decay/STS	STS12	STS13	STS14	STS15	Avg.
0.0001	55.04	70.79	65.55	73.36	66.19
0.001(ours)	**55.79**	**72.72**	**66.73**	**75.40**	**67.66**
0.01	55.22	70.48	65.43	74.12	66.31
0.1	54.94	70.26	64.75	72.58	65.63

The experimental results show that improper Weight Decay could make the model stop training early, resulting in underfitting and poor performance.

5.3. Effect of the LR of Predictor

As mentioned by [16], the Predictor with a constant LR (without decay) can obtain good image representation. Therefore, we design an experiment to verify whether the same settings can get good sentence representation. The results are shown in Table 6.

Table 6. The effect of the LR of Predictor. Decay: the LR of Predictor reduces with a cosine decay

LR/STS	STS12	STS13	STS14	STS15	Avg.
Decay	54.64	70.58	65.33	72.92	65.87
0.08	55.22	70.17	65.17	73.10	65.92
0.2	**56.24**	72.22	66.53	75.25	67.56
0.5	56.14	72.59	66.73	75.36	**67.71**
1(ours)	55.79	**72.72**	**66.73**	**75.40**	67.66

Experimental results show that Predictor with a constant LR can obtain better sentence representation compared with a decay LR. Specifically, as shown in Table 6 and Figure 4, the model will stop training (stop at 9th epochs) when the LR of the Predictor is small or reduced by a linear scaling rule. Besides, the model needs a bigger learning rate (LR=1) compared with vision tasks (LR=0.1) to get better results. A possible explanation is that the Predictor can adapt the latest representation. Therefore, it is not necessary to force Predictor to converge before the model is trained sufficiently [16].

5.4. Effect of the SGD Momentum

In general, an optimizer with Momentum can accelerate training because the update of the next step is based on the former steps. In other words, the gradient has a certain initial velocity (the network can remember the direction of gradient descent), which makes the network get rid of the local optima. In our proposed methods, the Momentum is set to 0.9 before Warm-up epochs and 0.8 after Warm-up epochs respectively. More details are shown in Table 7.

Table 7. The effect of the SGD Momentum (Mot). (0.9,0.8) means that the Momentum is 0.9 before Warm-up and 0.8 after Warm-up.

Momentum/STS	STS12	STS13	STS14	STS15	Avg.
0.8	55.47	72.45	66.50	74.84	67.32
0.9	55.27	71.96	66.27	75.18	67.17
0.99	54.76	70.86	65.69	73.93	66.24
(0.9,0.8)(ours)	**55.79**	**72.72**	**66.73**	**75.40**	**67.66**

We observe that small Momentum will take more time to train the model and not necessarily achieve the best performance. While a big Momentum can save the training time, the model can miss the optima resulting from a big step updating in the vicinity of the optimal point. Therefore, we set the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

Momentum to (0.9, 0.8) to accelerate the training before Warm-up epochs, and slow the updating step after Warm-up epochs, achieving better performance.

5.5. Effect of the Warm-up

In the training phase, the LR is linear scaling, i.e., the LR linearly increases to the maximum and reduces to the minimum, which can make a model more robust. Given that the parameters of a model are randomly initialized, it is inappropriate to employ a large LR in the first few updates of training because the noise of data may influence the performance. The comparison results are shown in Table 8.

Table 8. The effect of the Warm-up. 1, 2, 3 ,4 and 5 represent epochs that the LR starts to reduce.

Warm-up/STS	STS12	STS13	STS14	STS15	Avg.
1	55.62	72.52	66.54	74.75	67.36
2	55.45	72.38	66.42	74.90	67.29
3	55.62	72.54	66.56	75.09	67.45
4	55.64	72.55	66.60	75.07	67.47
5(ours)	**55.79**	**72.72**	**66.73**	**75.40**	**67.66**

Overall, the performances between the different Warm-up epochs are comparable. However, a small Warm-up can make the model stop early, especially for data with much noise.

5.6. Effect of the Region Size of Textcnn

The region size is a crucial parameter of textcnn. Therefore, we design different region sizes to investigate their impacts. The results are shown in Table 9.

Table 9. The effect of the Region Size of textcnn.

Region size/STS	STS12	STS13	STS14	STS15	Avg.
(1,2,3,4,5,6)	55.36	67.92	63.33	73.55	65.04
(2,3,4,5,6)	53.85	62.36	59.82	70.80	61.71
(1,1,1,2,3,4)	55.52	71.72	65.79	74.90	66.98
(1,1,1,4,5,6)	56.71	71.35	65.53	75.19	67.20
(1,1,1,1,1,1)	**57.81**	71.01	65.83	**75.62**	67.57
(1,1,1,6,15,20)(ours)	55.79	**72.72**	**66.73**	75.40	**67.66**
(1,1,1,20,30,40)	57.44	70.82	66.00	75.35	67.40

The experimental results show that the performance can be influenced by Region Size dramatically. Specifically, Region Size 1 is crucial for obtaining good results, observed from Region Size (1,2,3,4,5,6) and (2,3,4,5,6). A possible explanation is that Region Size 1 can enhance the representation of every word itself in a sentence without noise from other words. Furthermore, we increase the Region Size to study it. The results show that though big Region Size can obtain a better result compared with a small Region Size (1,1,1,2,3,4) on STS tasks, worse performance is obtained in transfer tasks (big Region Size will reduce by 0.3 percentage points). We argue that a big Region Size could obtain more context information, but at the same time, much noise is also added into the representation.

5.7. Effect of the Data Augmentation

Data augmentation can affect the quality of positive samples used for contrastive learning, which directly influences the performance and robustness of the model. Consequently, we propose two hypotheses for discrete text data: (1) partial data augmentation can retain more semantic information. (2) continuous data augmentation can guarantee that there is no semantic gap in augmented data. Next, we conduct an experiment to verify it. The results are shown in Table 10 and Figure 4.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 July 2023 doi:10.20944/preprints202307.1752.v1

Table 10. The effect of the data augmentation. No Aug.: no word vectors are performed the data augmentation. Full Aug.: all of word vectors of a sentence are augmented with our proposed four data augmentation strategies. Partial Aug.: word vectors are augmented by our proposed PWVA.

Augmentation/STS	STS12	STS13	STS14	STS15	Avg.
No Aug.	54.65	70.63	65.39	72.96	65.91
Full Aug.	55.31	70.51	65.81	73.49	66.28
Partial Aug.(ours)	**55.79**	**72.72**	**66.73**	**75.40**	**67.66**

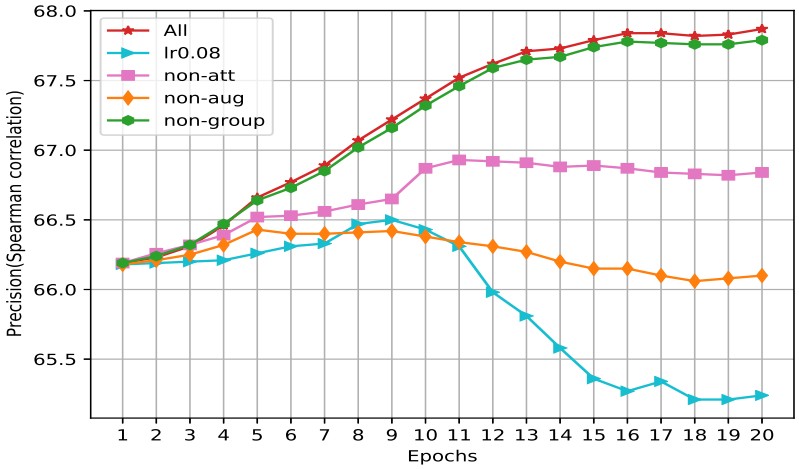


Figure 4. Comparison experiment results. “All" means that all methods we proposed are used. “lr0.08" represents that the learning rate of Predictor is 0.08. “non-att": the self-attention we proposed is not used in whole training. “non-aug": no data augmentations are applied on training, i.e., the two channels of network receive the same input. “non-group": feature grouping is not adopted to make use of local information of features.

As shown in experimental results, the continuous and partial PWVA exactly improves the performance compared with No Aug (from 65.91 to 67.66). and Full Aug. (from 66.28 to 67.66), which verifies our two hypotheses about data augmentation used in contrastive learning. In addition, the model can work well without data augmentation. A possible explanation is that unrecognizable words’ random initialization can be regarded as a method of data augmentation, resulting in an improvement of stability and robustness.

5.8. Effect of the Size of Groups

Grouping the features of the Projector and Predictor can solve the issue of information loss caused by contrastive loss computing between high-dimensional vectors. Therefore, we conduct an experiment to study the effect of the size of the feature grouping. The results are shown in Table 11.

Table 11. The effect of the size of feature grouping. No grouping: feature grouping is not performed.

Grouping size/STS	STS12	STS13	STS14	STS15	Avg.
No grouping	55.77	72.69	66.67	75.33	67.62
4	55.77	72.62	66.61	**75.50**	67.63
8	55.79	72.63	66.65	75.35	67.61
16(ours)	**55.79**	**72.72**	**66.73**	75.40	**67.66**
32	55.73	72.45	66.57	75.39	67.54
128	55.75	72.63	66.59	75.26	67.56

 doi:10.20944/preprints202307.1752.v1

Generally speaking, different grouping sizes can achieve comparable performance on STS tasks. Though the performance gap between feature grouping and no grouping is small, as observed from Figure 4, the stability and robustness of the model with feature grouping are better compared with no grouping. It verifies that the usage of local information by feature grouping can help the model mine more information for contrastive learning to advance the performance of sentence representation slightly (from 66.66 to 66.75).

6. Conclusion and Future Work

Previous works use big pre-trained language models to perform sentence representation (such as BERT, and RoBERT), which could not evaluate the performance of a lightweight model on sentence representation using contrastive learning. In this paper, we propose a lightweight model GCLSR to investigate the effectiveness of contrastive learning for sentence representation. GCLSR consists of continuous and partial data augmentation PWVA, self-attention, and grouped contrastive learning. GCLSR can obtain more original semantics from PWVA to produce high-quality positive samples. Self-attention can help GCLSR focus on informative words. Grouped contrastive learning can make use of more local information of features. The experimental results show that our proposed method GCLSR can produce a meaningful sentence representation. In the future, we intend to combine contrastive learning with self-attention further. In addition, we will perform our proposed method with a big pre-trained language model (such as BERT, and GPT) to get better results in sentence representation.

Author Contributions: Conceptualization, Qian Wang; methodology, Qian Wang; software, Qian Wang and Weiqi Zhang; validation, Tianyi Lei; formal analysis, Weiqi Zhang; investigation, Qian Wang; resources, Qian Wang; data curation, Tianyi Lei; writing—original draft preparation, Qian Wang; writing—review and editing, Qian Wang; visualization, Weiqi Zhang; Dezhong Peng, X.X.; project administration, Dezhong Peng; funding acquisition, Dezhong Peng.

Funding: This research is financially supported by the National Natural Science Foundation of China (Grant No. U19A2078).

Data Availability Statement: Our code and training data of GCLSR are available on https://github.com/qianandfei/GCLSR.

Acknowledgments: The authors acknowledge the financial support provided by the Sichuan Science and Technology Planning Project (Grant Nos. 2021YFG0301, 2021YFG0317, 2022YFQ0014, 2022Y-FH0021, 2023YFG0033, 2023ZHCG0016).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhao, D.; Wang, J.; Lin, H.; Chu, Y.; Wang, Y.; Zhang, Y.; Yang, Z. Sentence representation with manifold learning for biomedical texts. *Knowledge-Based Systems* **2021**, *218*, 106869.
2. Li, B.; Zhou, H.; He, J.; Wang, M.; Yang, Y.; Li, L. On the sentence embeddings from pre-trained language models. *arXiv preprint arXiv:2011.05864* **2020**.
3. Logeswaran, L.; Lee, H. An efficient framework for learning sentence representations. *arXiv preprint arXiv:1803.02893* **2018**.
4. Kim, T.; Yoo, K.M.; Lee, S.g. Self-Guided Contrastive Learning for BERT Sentence Representations. *arXiv preprint arXiv:2106.07345* **2021**.
5. Zhang, D.; Li, S.W.; Xiao, W.; Zhu, H.; Nallapati, R.; Arnold, A.O.; Xiang, B. Pairwise supervised contrastive learning of sentence representations. *arXiv preprint arXiv:2109.05424* **2021**.
6. Ethayarajh, K. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. *arXiv preprint arXiv:1909.00512* **2019**.
7. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* **2019**.
8. Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; Hon, H.W. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197* **2019**.

9. Wu, L.; Hu, J.; Teng, F.; Li, T.; Du, S. Text semantic matching with an enhanced sample building method based on contrastive learning. *International Journal of Machine Learning and Cybernetics* **2023**, pp. 1–8.
10. Ma, X.; Li, H.; Shi, J.; Zhang, Y.; Long, Z. Importance-aware contrastive learning via semantically augmented instances for unsupervised sentence embeddings. *International Journal of Machine Learning and Cybernetics* **2023**, pp. 1–12.
11. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* **2018**.
12. Liu, P.; Guo, Y.; Wang, F.; Li, G. Chinese named entity recognition: The state of the art. *Neurocomputing* **2022**, *473*, 37–53.
13. Yu, P.; Weizhong, Q. Three-stage question answering model based on BERT. *Journal of Computer Applications* **2022**, *42*, 64.
14. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. International conference on machine learning. PMLR, 2020, pp. 1597–1607.
15. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum contrast for unsupervised visual representation learning. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9729–9738.
16. Chen, X.; He, K. Exploring simple siamese representation learning. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 15750–15758.
17. Grill, J.B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P.H.; Buchatskaya, E.; Doersch, C.; Pires, B.A.; Guo, Z.D.; Azar, M.G.; others. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733* **2020**.
18. Reimers, N.; Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* **2019**.
19. Giorgi, J.M.; Nitski, O.; Bader, G.D.; Wang, B. Declutr: Deep contrastive learning for unsupervised textual representations. *arXiv preprint arXiv:2006.03659* **2020**.
20. Wu, Z.; Wang, S.; Gu, J.; Khabsa, M.; Sun, F.; Ma, H. Clear: Contrastive learning for sentence representation. *arXiv preprint arXiv:2012.15466* **2020**.
21. Gao, T.; Yao, X.; Chen, D. SimCSE: Simple Contrastive Learning of Sentence Embeddings. *arXiv preprint arXiv:2104.08821* **2021**.
22. Wang, Q.; Zhang, W.; Lei, T.; Cao, Y.; Peng, D.; Wang, X. CLSEP: Contrastive learning of sentence embedding with prompt. *Knowledge-Based Systems* **2023**, p. 110381.
23. Fang, H.; Wang, S.; Zhou, M.; Ding, J.; Xie, P. Cert: Contrastive self-supervised learning for language understanding. *arXiv preprint arXiv:2005.12766* **2020**.
24. Zhu, W.; Cheung, D. CMV-BERT: Contrastive multi-vocab pretraining of BERT. *arXiv preprint arXiv:2012.14763* **2020**.
25. Yan, Y.; Li, R.; Wang, S.; Zhang, F.; Wu, W.; Xu, W. ConSERT: A Contrastive Framework for Self-Supervised Sentence Representation Transfer. *arXiv preprint arXiv:2105.11741* **2021**.
26. Wei, J.; Zou, K. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196* **2019**.
27. Wang, W.Y.; Yang, D. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 2557–2563.
28. Guo, H.; Mao, Y.; Zhang, R. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941* **2019**.
29. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* **2014**.
30. Uchaikin, V.V.; Zolotarev, V.M. Chance and Stability: Stable Distributions and their Applications **2011**.
31. Géron, A. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow **2022**.
32. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* **2013**.
33. Kim, Y. Convolutional Neural Networks for Sentence Classification. *Eprint Arxiv* **2014**.

34. Ioffe, S.; Normalization, C.S.B. Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv preprint arXiv:1502.03167*.
35. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983* **2016**.
36. Conneau, A.; Kiela, D. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449* **2018**.
37. Agirre, E.; Cer, D.; Diab, M.; Gonzalez-Agirre, A. Semeval-2012 task 6: A pilot on semantic textual similarity. * SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012), 2012, pp. 385–393.
38. Agirre, E.; Cer, D.; Diab, M.; Gonzalez-Agirre, A.; Guo, W. * SEM 2013 shared task: Semantic textual similarity. Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity, 2013, pp. 32–43.
39. Agirre, E.; Banea, C.; Cardie, C.; Cer, D.; Diab, M.; Gonzalez-Agirre, A.; Guo, W.; Mihalcea, R.; Rigau, G.; Wiebe, J. Semeval-2014 task 10: Multilingual semantic textual similarity. Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014), 2014, pp. 81–91.
40. Agirre, E.; Banea, C.; Cardie, C.; Cer, D.; Diab, M.; Gonzalez-Agirre, A.; Guo, W.; Lopez-Gazpio, I.; Maritxalar, M.; Mihalcea, R.; others. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015), 2015, pp. 252–263.
41. Agirre, E.; Banea, C.; Cer, D.; Diab, M.; Gonzalez Agirre, A.; Mihalcea, R.; Rigau Claramunt, G.; Wiebe, J. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511. ACL (Association for Computational Linguistics), 2016.
42. Cer, D.; Diab, M.; Agirre, E.; Lopez-Gazpio, I.; Specia, L. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055* **2017**.
43. Marelli, M.; Menini, S.; Baroni, M.; Bentivogli, L.; Bernardi, R.; Zamparelli, R.; others. A SICK cure for the evaluation of compositional distributional semantic models. Lrec. Reykjavik, 2014, pp. 216–223.
44. Pang, B.; Lee, L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv preprint cs/0506075* **2005**.
45. Hu, M.; Liu, B. Mining and summarizing customer reviews. Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 2004, pp. 168–177.
46. Pang, B.; Lee, L. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *arXiv preprint cs/0409058* **2004**.
47. Wiebe, J.; Wilson, T.; Cardie, C. Annotating expressions of opinions and emotions in language. *Language resources and evaluation* **2005**, *39*, 165–210.
48. Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.D.; Ng, A.Y.; Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. Proceedings of the 2013 conference on empirical methods in natural language processing, 2013, pp. 1631–1642.
49. Voorhees, E.M.; Tice, D.M. Building a question answering test collection. Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, 2000, pp. 200–207.
50. Dolan, W.B.; Brockett, C. Automatically constructing a corpus of sentential paraphrases. Proceedings of the Third International Workshop on Paraphrasing (IWP2005), 2005.