ESimCSE: Enhanced Sample Building Method for Contrastive Learning of Unsupervised Sentence Embedding

Xing Wu\textsuperscript{1,2,3}, Chaochen Gao\textsuperscript{1,2}, Liangjun Zang\textsuperscript{1}, Jizhong Han\textsuperscript{1}, Zhongyuan Wang\textsuperscript{3}, Songlin Hu\textsuperscript{1,2}\textsuperscript{†}

\textsuperscript{1}Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
\textsuperscript{2}School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
\textsuperscript{3}Kuaishou Technology, Beijing, China
{wuxing,gaochaochen,zangliangjun,hanjizhong,husonglin}@iie.ac.cn
wangzhongyuan@kuaishou.com

Abstract

SimCSE\textsuperscript{1} adopts dropout as data augmentation and encodes an input sentence twice into two corresponding embeddings to build a positive pair. Since SimCSE is a Transformer-based encoder that directly encodes the length information of sentences through positional embeddings, the two embeddings in a positive pair contain the same length information. Thus, a model trained with these positive pairs is biased, tending to consider that sentences of the same or similar length are more similar in semantics. To alleviate it, we apply a simple but effective repetition operation to modify the input sentence. Then we pass the input sentence and its modified counterpart to the pre-trained Transformer encoder, respectively, to get the positive pair. Additionally, we draw inspiration from the computer vision community and introduce momentum contrast to enlarge the number of negative pairs without additional calculations. The proposed modifications are applied to positive and negative pairs separately, and build a new sentence embedding method, termed Enhanced SimCSE (ESimCSE). We evaluate the proposed ESimCSE on several benchmark datasets w.r.t the semantic text similarity (STS) task. Experimental results show that ESimCSE outperforms SimCSE by an average Spearman correlation of 2.02% on BERT-base. Our code are available at https://github.com/caskcsg/ESimCSE.

1 Introduction

Recently, researchers have proposed using contrastive learning to learn better unsupervised sentence embeddings (Wu et al.; Zhang et al., b; Liu et al., 2021; Gao et al., 2021; Yan et al., 2021). Contrastive learning aims to learn effective sentence embeddings based on the assumption that effective sentence embeddings should bring similar sentences closer while pushing away dissimilar ones. It generally uses various data augmentation methods (Shleifer, 2019; Wei and Zou, 2019; Wu et al., 2019) to generate different views for each sentence randomly, and assumes a sentence is semantically more similar to its augmented counterpart than any other sentence. Among these methods, the most representative one is SimCSE (Gao et al., 2021), which performs on par with previously supervised counterparts. SimCSE implicitly hypothesizes dropout acts as a minimal data augmentation method. Specifically, SimCSE composes \( N \) sentences in a batch and feeds each sentence to the pre-trained BERT twice with two independently sampled dropout masks. Then the embeddings derived from the same sentence constitute a “positive pair”, while those derived from two different sentences constitute a “negative pair”.

Using dropout as a minimal data augmentation method is simple and effective, but there is a weak point. SimCSE models are built on Transformer blocks, which will encode a sentence’s length information through positional embeddings. In a positive pair, two embeddings are derived from the same sentence to contain the same length information. In contrast, in a negative pair, two embeddings in a negative pair are derived from two different sentences and generally contain different length information. Therefore, positive and negative pairs are different in their length information, acting as a feature to distinguish them. The semantic simi-

<table>
<thead>
<tr>
<th>Length Diff</th>
<th>Avg. Similarity Diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>( &gt; 3 )</td>
<td>16.34</td>
</tr>
<tr>
<td>( \leq 3 )</td>
<td>18.18 (+1.84)</td>
</tr>
</tbody>
</table>

Table 1: The average similarity difference between the model (SimCSE-BERT) predictions and the normalized ground truths.

The first two authors contribute equally.
Corresponding author.
\textsuperscript{1}We focus on unsupervised sentence embedding, so SimCSE in this article refers to unsupervised SimCSE.
Table 2: An example of semantic similarity after different methods change a sentence’s length.

<table>
<thead>
<tr>
<th>Method</th>
<th>Text</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>original sentence</td>
<td>I like this apple because it looks so fresh and it should be delicious.</td>
<td>1.0</td>
</tr>
<tr>
<td>random insertion</td>
<td>I don’t like this apple because <strong>but</strong> it looks so <strong>not</strong> fresh and it should be <strong>dog</strong> delicious.</td>
<td>0.69</td>
</tr>
<tr>
<td>random deletion</td>
<td>I like this <strong>apple</strong> because it looks so <strong>fresh</strong> and it should be delicious.</td>
<td>0.32</td>
</tr>
<tr>
<td>word repetition</td>
<td>I like <strong>like</strong> this apple because it looks so <strong>so</strong> fresh and <strong>and</strong> it should be delicious.</td>
<td>0.99</td>
</tr>
<tr>
<td>word repetition</td>
<td>I <strong>I</strong> like this apple <strong>apple</strong> because it looks <strong>looks</strong> so fresh <strong>fresh</strong> and it should be delicious <strong>delicious</strong>.</td>
<td>0.98</td>
</tr>
</tbody>
</table>

Larity model trained with these pairs can be biased, which probably considers that two sentences of the same or similar lengths are more similar in semantics. To confirm it, we evaluate on seven standard semantic textual similarity datasets with the SimCSE-BERT base model published by (Gao et al., 2021). We partition each STS test set into two groups based on whether the sentence pairs’ length difference is ≤ 3. We calculate the similarity differences between the model predictions and the normalized ground truths for each group. As shown in Table 1, the average similarity difference of seven datasets is higher when the length difference is ≤ 3, which verifies our assumption. Comparison details on each dataset can refer to Table 7.

To alleviate this problem, we propose a simple but effective enhancement method to SimCSE. For each positive pair, we expect to change the length of a sentence without changing its semantic meaning. Existing methods to change the length of a sentence generally use random insertion and random deletion. However, inserting randomly selected words into a sentence may introduce extra noise, which will probably distort the meaning of the sentence; deleting keywords from a sentence will also change its semantics substantially. Such operations are detrimental to SimCSE learning, which is also discussed in a contemporaneous work (Chuang et al., 2022). Therefore, we propose a safer method, termed “word repetition”, which randomly duplicates some words in a sentence. For example, as shown in Table 2, either random insertion or random deletion may generate a sentence that deviates far from the meaning of the original sentence. On the contrary, the method of “word repetition” maintains the meaning of the original sentence quite well.

Apart from the optimization above for positive pairs construction, we further explore how to optimize the construction of negative pairs. Since contrastive learning is carried out between positive pairs and negative pairs, theoretically, more negative pairs can lead to a better comparison between the pairs (Chen et al.). And thus, a potential optimization direction is to leverage more negative pairs, encouraging the model towards more refined learning. However, according to (Gao et al., 2021), larger batch size is not always a better choice. For example, for the SimCSE-BERT base model, the optimal batch size is 64, and other settings of the batch size will lower the performance. Therefore, we tend to figure out how to expand the negative pairs more effectively. In the community of computer vision, to alleviate the GPU memory limitation when expanding the batch size, a feasible way is to introduce the momentum contrast (He et al.), which is also applied to natural language understanding (Fang et al.). Momentum contrast allows us to reuse the encoded embeddings from the immediate preceding mini-batches to expand the negative pairs by maintaining a queue. It always enqueues the sentence embeddings of the current mini-batches and meanwhile dequeues the “oldest” ones. As the enqueued sentence embeddings come from the preceding mini-batches, we keep a momentum updated encoder by taking the moving average of its parameters and use the momentum encoder to generate enqueued sentence embeddings. Note that, we turn off dropout when using the momentum encoder, which can narrow the gap between training and prediction.

The above two optimizations are proposed separately for building positive and negative pairs. We finally combine both with SimCSE, termed Enhanced SimCSE (ESimCSE). We illustrate the schematic diagram of ESimCSE in Figure 1. The proposed ESimCSE is evaluated on the semantic
text similarity (STS) task with 7 STS-B test sets. Experimental results show that ESimCSE can improve the similarity measuring performance in different model settings over the previous state-of-the-art SimCSE. Specifically, ESimCSE gains an average increase of Spearman’s correlation over SimCSE by +2.02% on BERT\textsubscript{base}.

Our contributions can be summarized as follows:

- We observe that SimCSE constructs each positive pair with two sentences of the same length, which can bias the learning process. We propose a simple but effective “word repetition” method to alleviate the problem.
- We propose to use the momentum contrast method to increase the number of negative pairs involved in the loss calculation, which encourages the model towards more refined learning.
- We conduct extensive experiments on several benchmark datasets w.r.t semantic text similarity task. The experimental results well demonstrate that both proposed optimizations bring improvements to SimCSE.

2 Background: SimCSE

Given a set of paired sentences \( \{ x_i, x_i^+ \}_{i=1}^m \), where \( x_i \) and \( x_i^+ \) are semantically related and will be referred to positive pairs. The core idea of SimCSE is to use identical sentences to build the positive pairs, i.e., \( x_i^+ = x_i \). Note that in Transformer, there is a dropout mask placed on fully-connected layers and attention probabilities. And thus, the key ingredient is to feed the same input \( x_i \) to the encoder twice by applying different dropout masks \( z_i \) and \( z_i^+ \) and output two separate sentence embeddings to build a positive pair as follows:

\[
h_i = f_\theta(x_i, z_i), \quad h_i^+ = f_\theta(x_i, z_i^+) \tag{1}
\]

With \( h_i \) and \( h_i^+ \) for each sentence in a mini-batch with batch size \( N \), the contrastive learning objective w.r.t \( x_i \) is formulated as follows,

\[
\ell_i = - \log \frac{e^{\text{sim}(h_i, h_i^+)/\tau}}{\sum_{j=1}^N e^{\text{sim}(h_i, h_j^+)/\tau}} \tag{2}
\]

where \( \tau \) is a temperature hyperparameter and \( \text{sim}(h_i, h_j^+) \) is the similarity metric, which is typically the cosine similarity function.

3 Proposed Enhanced SimCSE

In this section, we first introduce the word repetition method to construct better positive pairs. Then we introduce the momentum contrast method to expand negative pairs.

3.1 Word Repetition

The word repetition mechanism randomly duplicates some words/sub-words in a sentence. Here we take sub-word repetition as an example. Given a sentence \( s \), after processing by a sub-word tokenizer, we get a sub-word sequence \( x = \)
\{x_1, x_2, \ldots, x_N\}, N being the length of sequence. We define the number of repeated tokens as

\[
dup\_len \in [0, \text{max}(2, \text{int}(\text{dup\_rate} \times N))] (3)
\]

where \(\text{dup\_rate}\) is the maximal repetition rate, which is a hyperparameter. Then \(\text{dup\_len}\) is a randomly sampled number in the set defined above, which will introduce more diversity when extending the sequence length. After \(\text{dup\_len}\) is determined, we use uniform distribution to randomly select \(\text{dup\_len}\) sub-words that need to be repeated from the sequence, which composes the \(\text{dup\_set}\) as follows,

\[
\text{dup\_set} = \text{uniform}([1, N], \text{num} = \text{dup\_len}) (4)
\]

For example, if the 1st sub-word is in \(\text{dup\_set}\), then sequence \(x\) becomes \(x^+ = \{x_1, x_1, x_2, \ldots, x_N\}\). And different from SimCSE which passes \(x\) to the pre-trained BERT twice, E-SimCSE passes \(x\) and \(x^+\) independently.

### 3.2 Momentum Contrast

The momentum contrast allows us to reuse the encoded sentence embeddings from the immediate preceding mini-batches by maintaining a queue of a fixed size. Specifically, the embeddings in the queue are progressively replaced. When the output sentence embeddings of the current mini-batch is enqueued, the “oldest” ones in the queue are removed if the queue is full. Note that we use a momentum-updated encoder to encode the enqueued sentence embeddings. Formally, denoting the parameters of the encoder as \(\theta_e\) and those of the momentum-updated encoder as \(\theta_m\), we update \(\theta_m\) in the following way,

\[
\theta_m \leftarrow \lambda \theta_m + (1 - \lambda) \theta_e (5)
\]

where \(\lambda \in [0, 1]\) is a momentum coefficient parameter. Note that only the parameters \(\theta_e\) are updated by back-propagation. And here we introduce \(\theta_m\) to generate sentence embeddings for the queue, because the momentum update can make \(\theta_m\) evolve more smoothly than \(\theta_e\). As a result, though the embeddings in the queue are encoded by different encoders (in different “steps” during training), the difference among these encoders can be made small.

With sentence embeddings in the queue, the loss function of ESimCSE is further modified as follows,

\[
\ell_i = -\log \frac{e^{\text{sim}(h_i, h_i^+)}/\tau}{\sum_{j=1}^N e^{\text{sim}(h_i, h_j^+)/\tau} + \sum_{m=1}^M e^{\text{sim}(h_i, h_m^+)/\tau}} (6)
\]

where \(h_i^+\) denotes a sentence embedding in the momentum-updated queue, and \(M\) is the size of the queue.

### 4 Experiment

#### 4.1 Experiment Setup

Our experimental language is English. For a fair comparison, our experimental setup mainly follows SimCSE. We use 1-million sentences randomly drawn from English Wikipedia for training\(^2\). The semantic textual similarity task measures the capability of sentence embeddings, and we conduct our experiments on seven standard semantic textual similarity (STS) datasets. STS12-STS16 datasets (Agirre et al., d,e,b,a,c) do not have train or development sets, and thus we evaluate the models on the development set of STS-B (Cer et al.) to search for better settings of the hyper-parameters. The SentEval toolkit\(^3\) is used for evaluation, and Spearman correlation coefficient \(^4\) is used to report the model performance. All the experiments are conducted on Nvidia 3090 GPUs.

#### 4.2 Training Details

We start from pre-trained checkpoints of BERT(uncased) or RoBERTa(cased) using both the base and the large versions, and we add an MLP layer on top of the [CLS] representation to get the sentence embedding. We implement ESimCSE based on Huggingface’s transformers package\(^5\). We train our models for one epoch using the Adam optimizer with the batch size = 64 and the temperature \(\tau = 0.05\) in Eq. (3). The learning rate is set as 3e-5 for ESimCSE-BERT\_base model and 1e-5 for other models. The dropout rate is \(p = 0.1\) for base models, \(p = 0.15\) for large models. For the momentum contrast, we empirically choose a relatively large momentum \(\lambda\)

\(^2\)https://huggingface.co/datasets/princeton-nlp/datasets-for-simcse/resolve/main/wiki1m_for_simcse.txt
\(^3\)https://github.com/facebookresearch/SentEval
\(^4\)https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient
\(^5\)https://github.com/huggingface/transformers, version 4.2.1.
### 4.3 Main Results

Table 3 shows the models’ performance on seven semantic textual similarity (STS) test sets. We mainly select SimCSE for comparison, since it is the current state-of-the-art and shares the same setting as our approach. In addition, we also use IS-BERT (Zhang et al., a), CT-BERT (Carlsson et al., 2021), ConSERT (Yan et al., 2021), SG-OPT (Kim et al., 2021), BERT-flow (Li et al., 2020), Mirror-BERT (Liu et al., 2021) as baselines. It can be seen that ESimCSE improves the measurement of semantic textual similarity in different settings over SimCSE. Specifically, ESimCSE outperforms SimCSE by +2.02% on BERT\textsubscript{base}, +0.90% on BERT\textsubscript{large}, +0.87% on RoBERTa\textsubscript{base}, +0.55% on RoBERTa\textsubscript{large}, respectively.

### 5 Ablation Study

This section investigates how different settings affect ESimCSE’s performance. All results are compared on BERT\textsubscript{base} scale models and are evaluated on the development set of STS-B unless otherwise specified.

#### 5.1 The Importance of Word Repetition and Momentum Contrast

We explore how much improvement it can bring to SimCSE when only using word repetition or momentum contrast. As shown in Table 4, either word

<table>
<thead>
<tr>
<th>Model</th>
<th>STS-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimCSE</td>
<td>82.45</td>
</tr>
<tr>
<td>+ word repetition</td>
<td>84.09 (+1.64)</td>
</tr>
<tr>
<td>+ momentum contrast</td>
<td>83.98 (+1.53)</td>
</tr>
<tr>
<td>ESimCSE</td>
<td><strong>84.85 (+2.40)</strong></td>
</tr>
</tbody>
</table>

Table 4: Improvement on STS-B development sets that word repetition or momentum contrast brings to SimCSE. 

---

repetition or momentum contrast can bring substantial improvements to SimCSE. It means that both proposed methods to enhance the positive pairs and negative pairs are effective. Better yet, these two modifications can be superimposed (ESimCSE) to get further improvements.

5.2 Effect of Sentence-Length-Extension Method

In addition to sub-word repetition, we also explore three other methods to increase sentence length:

- **Word Repetition** is similar to sub-word repetition, except that the repetition operation occurs before tokenization. For example, given a word “microbiology”, word repetition will produce “microbiology microbiology”, while sub-word repetition will produce “micro #biology” or “micro #biology #biology”.

- **Inserting Stop-words** inserts a random stop-word after the selected word instead of repeating the selected word.

- **Inserting [MASK]** inserts a [MASK] token after the selected word. We can regard [MASK] as a dynamic context-compatible word placeholder.

- **Inserting Masked Prediction** inserts a [MASK] token after the selected word and uses the masked language model to predict the top-1 substitution. The substitution is used to replace the inserted [MASK] token.

Inserting masked prediction also brings a good improvement, but this method requires a pre-trained masked language model to predict replacements, bringing high additional computational overhead.

5.3 Batching Sentences of Similar Length in Training

Apart from sentence-length-extension methods, we explore whether batching sentences of similar length in training will alleviate the bias towards identical sequence length in inference. We divide training sentences into buckets by length and batch them within each bucket. We explore two different settings:

- We divide the training set into two coarse-grained buckets based on whether the sentence length is greater than \( buc\_len \), where \( buc\_len \in [3, 8] \);

- We divide the training set by sentence length into 6 fine-grained buckets: \( \{ \leq 3, 4, 5, 6, 7, \geq 8 \} \), which we use \( buc\_len = 3 \sim 8 \) for short.

We list the experimental results in Table 6. Dividing the training set into buckets does not bring significant improvements and even decreases in some settings. We believe that after being divided into buckets, shuffle can only be performed within a bucket, leading to an insufficient comparison in contrastive learning. In contrast, the effect of word repetition is much better.

5.4 The Relationship between The Similarity and Length Difference

We further explore the relationship between the similarity and length difference of sentence pairs on ESimCE, compared with that of SimCSE in the Introduction. As STS12-STS16 datasets do not have train or development sets, and thus we evaluate the models on the test set of each dataset. We partition each STS test set into two groups based

<table>
<thead>
<tr>
<th>buc_len</th>
<th>wr</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>STS-B</td>
<td>84.09</td>
<td>81.92</td>
<td>82.00</td>
<td>82.66</td>
</tr>
<tr>
<td>buc_len</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>3 \sim 8</td>
</tr>
<tr>
<td>STS-B</td>
<td>82.00</td>
<td>82.13</td>
<td>83.00</td>
<td>82.18</td>
</tr>
</tbody>
</table>

Table 6: Effects of different bucket lengths buc\_len. “wr” means using word repetition method instead of bucketing sentences. “3 \sim 8” means fine-grained buckets setting: \( \{ \leq 3, 4, 5, 6, 7, \geq 8 \} \).
Table 7: The difference between the model predicted cosine similarity and the true label on each dataset’s test set. “LD” is short for length difference.

<table>
<thead>
<tr>
<th>Model</th>
<th>LD</th>
<th>STS12</th>
<th>STS13</th>
<th>STS14</th>
<th>SICK15</th>
<th>STS16</th>
<th>STS-B</th>
<th>SICK-R</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimCSE</td>
<td>&gt; 3</td>
<td>8.93</td>
<td>15.74</td>
<td>11.90</td>
<td>19.68</td>
<td>28.91</td>
<td>21.33</td>
<td>7.86</td>
<td>16.34</td>
</tr>
<tr>
<td></td>
<td>≤ 3</td>
<td>9.29</td>
<td>22.81</td>
<td>19.53</td>
<td>19.92</td>
<td>24.08</td>
<td>22.12</td>
<td>9.53</td>
<td>18.18</td>
</tr>
<tr>
<td>ESimCSE</td>
<td>&gt; 3</td>
<td>13.48</td>
<td>23.73</td>
<td>17.14</td>
<td>25.98</td>
<td>34.71</td>
<td>26.22</td>
<td>10.44</td>
<td>21.67</td>
</tr>
<tr>
<td></td>
<td>≤ 3</td>
<td>12.52</td>
<td>28.56</td>
<td>24.13</td>
<td>24.17</td>
<td>29.32</td>
<td>25.63</td>
<td>12.35</td>
<td>22.38</td>
</tr>
</tbody>
</table>

Table 8: Effect of repeated words on the average similarity of two sets

<table>
<thead>
<tr>
<th>Model</th>
<th>Sim &lt;q,s1&gt;</th>
<th>Sim &lt;q,s2&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimCSE</td>
<td>26.39</td>
<td>27.07(+0.68)</td>
</tr>
<tr>
<td>ESimCSE</td>
<td>36.82</td>
<td>36.87(+0.05)</td>
</tr>
</tbody>
</table>

Table 9: Effects of repetition rate dup_rate.

<table>
<thead>
<tr>
<th>dup_rate</th>
<th>STS-B</th>
<th>STS-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.08</td>
<td>83.5</td>
<td>83.62</td>
</tr>
<tr>
<td>0.12</td>
<td>82.96</td>
<td>84.85</td>
</tr>
<tr>
<td>0.16</td>
<td>84.01</td>
<td>83.84</td>
</tr>
</tbody>
</table>

5.5 Will Word Repetition Bring New Bias?

We further explore whether word repetition will mislead the model to be more inclined to consider sentences with repeated overlaps are more similar. We conduct a detection experiment on wiki data with the following settings:

1. We randomly select a sentence as a query, such as q = “I like this apple because it looks very fresh”
2. We use the query to randomly recall a candidate sentence with 13%-17% overlap tokens, such as s1 = “This is a very tall tree and it looks like a giant”
3. We apply the word-repetition operation on the overlap tokens in the candidate sentence and produce a word-repeated sentence, such as s2 = “This this is a very very tall tree and it looks looks like a giant.”
4. We calculate the similarity of <q, s1> and <q, s2> and compare them.

We experiment on 100 different query sentences and calculate their average similarity. As shown in Table 8, compared to the 0.68 increase of the SimCSE, ESimCSE-BERT only increased by 0.05. Therefore, word repetition does not bring a new bias to the learning process.

5.6 Effect of Hyperparameters

Repetition Rate To quantitatively study the effect of repetition rate on the model performance, we slowly increase the repetition rate parameter dup_rate from 0.08 to 0.36, with each increase by 0.04. As shown in Table 9, when dup_rate = 0.32, ESimCSE achieves the best performance, a larger or smaller dup_rate will cause performance degradation, which is consistent with our intuition.

Momentum Queue Size The size of the momentum contrast queue determines the number of negative pairs involved in the loss calculation. We experiment with the queue size equals to different multiples of the batch size. The experimental results are listed in Table 10. The optimal result is reached when the queue size was 2.5 times the batch size. A smaller queue size will reduce the effect. This is intuitive because more negative pairs participate in the loss calculation to compare positive pairs more fully. But a too large queue size also reduces the effect. We guess that is because the negative pairs in the momentum contrast are generated by the past “steps” during training, and a larger queue will use the outputs of more outdated encoder models which are quite different from the current one. And thus that will reduce the reliability of the loss calculation.
Table 10: Effects of queue size of momentum contrast.

<table>
<thead>
<tr>
<th>Queue Size</th>
<th>STS-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \times \text{batch}_\text{size}$</td>
<td>83.83</td>
</tr>
<tr>
<td>$1.5 \times \text{batch}_\text{size}$</td>
<td>83.81</td>
</tr>
<tr>
<td>$2 \times \text{batch}_\text{size}$</td>
<td>83.03</td>
</tr>
<tr>
<td>$2.5 \times \text{batch}_\text{size}$</td>
<td>84.85</td>
</tr>
<tr>
<td>$3 \times \text{batch}_\text{size}$</td>
<td>82.66</td>
</tr>
</tbody>
</table>

5.7 Performance on Transfer Tasks

Following (Gao et al., 2021), we further evaluate ESimCSE on transfer tasks, to see the transferability of the sentence embeddings from ESimCSE. The transfer tasks include: MR (movie review) (Pang and Lee, 2005), CR (product review) (Hu and Liu, 2004), SUBJ (subjectivity status) (Pang and Lee, 2004), MPQA (opinion-polarity) (Wiebe et al., 2005), SST-2 (binary sentiment analysis) (Socher et al., 2013), TREC (question-type classification) (Voorhees and Tice, 2000) and MRPC (paraphrase detection) (Dolan and Brockett, 2005). For more details, one can refer to SentEval. As shown in Table 11, compared with the performance of SimCSE, ESimCSE slightly increases the transferability of embedding. As our optimizations are focused on semantic textual similarity tasks, the ability of ESimcse on transfer tasks remains stable relative to SimCSE.

6 Related Work

Unsupervised sentence representation learning has been widely studied. (Socher et al.; Hill et al.; Le and Mikolov) propose to learn sentence representation according to the internal structure of each sentence. (Kiros et al.; Logeswaran and Lee) predict the surrounding sentences of a given sentence based on the distribution hypothesis. (Pagliardini et al.) propose Sent2Vec, a simple unsupervised model allowing to compose sentence embeddings using word vectors along with n-gram embeddings. Recently, contrastive learning has been explored in unsupervised sentence representation learning and has become a promising trend (Zhang et al., b; Wu et al.; Meng et al.; Liu et al., 2021; Gao et al., 2021; Yan et al., 2021; Chuang et al., 2022). Those contrastive learning based methods for sentence embeddings are generally based on the assumption that a good semantic representation should be able to bring similar sentences closer while pushing away dissimilar ones. Therefore, those methods use various data augmentation methods to randomly generate two different views for each sentence and design an effective loss function to make them closer in the semantic representation space. Among these contrastive methods, the most related ones to our work are unsup-ConSERT (Yan et al., 2021) and unsup-SimSCE (Gao et al., 2021). ConSERT explores various effective data augmentation strategies (e.g., adversarial attack, token shuffling, Cutoff, dropout) to generate different views for contrastive learning and analyze their effects on unsupervised sentence representation transfer. Unsup-SimSCE, the current state-of-the-art unsupervised method uses only standard dropout as minimal data augmentation, and feed an identical sentence to a pre-trained model twice with independently sampled dropout masks to generate two distinct sentence embeddings as a positive pair. Unsup-SimSCE is very simple but works surprisingly well, performing on par with previously supervised counterparts. However, we find that SimCSE constructs each positive pair with two sentences of the same length, which can mislead the learning of sentence embeddings. So we propose a simple but effective method timed “word repetition” to alleviate it. We also propose to use the momentum contrast method to increase the number of negative pairs involved in the loss calculation, which encourages the model towards more refined learning.

7 Conclusion and Future Work

In this paper, we propose optimizations to construct positive and negative pairs for SimCSE and combine them with SimCSE, which is termed ESimCSE. Through extensive experiments, the proposed ESimCSE achieves considerable improvements on standard semantic text similarity tasks over SimCSE.

In the future, we will focus on designing a more refined objective function to improve the discrimination between different negative pairs. Also we will make attempt to optimize the performance on both semantic textual similarity tasks and transfer tasks.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weixi Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada
Table 11: Results on transfer tasks of different sentence embedding models, in terms of accuracy. ♣: results from (Gao et al., 2021).

<table>
<thead>
<tr>
<th>Model</th>
<th>MR</th>
<th>CR</th>
<th>SUBJ</th>
<th>MPQA</th>
<th>SST</th>
<th>TREC</th>
<th>MRPC</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimCSE</td>
<td>81.18</td>
<td>86.46</td>
<td>94.45</td>
<td>88.88</td>
<td>85.50</td>
<td>89.80</td>
<td>74.43</td>
<td>85.81</td>
</tr>
<tr>
<td>ESimCSE</td>
<td>81.32</td>
<td>86.22</td>
<td>94.74</td>
<td>88.74</td>
<td>85.50</td>
<td>91.00</td>
<td>74.90</td>
<td>86.06</td>
</tr>
</tbody>
</table>


Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janrye Wiebe. b. Semeval-2014 task 10: Multilingual semantic textual similarity.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau Claramunt, and Janrye Wiebe. c. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation.


Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. e. * sem 2013 shared task: Semantic textual similarity.


Yu Meng, Chenyan Xiong, Payal Bajaj, Saurabh Tiyari, Paul Bennett, Jiawei Han, and Xia Song. Coco-lm: Correcting and contrasting text sequences for language model pretraining. arXiv preprint arXiv:2102.08473.


Richard Socher, Eric Huang, Jeffrey Penning, Christopher D Manning, and Andrew Ng. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. Advances in neural information processing systems.


