A Sentence is Worth 128 Pseudo Tokens
A Semantic-Aware Contrastive Learning Framework for Sentence Embeddings
Tan, Haochen; Shao, Wei; Wu, Han; Yang, Ke; Song, Linqi

Published in:
Findings of the Association for Computational Linguistics

Published: 01/01/2022

Document Version:
Final Published version, also known as Publisher’s PDF, Publisher’s Final version or Version of Record

License:
CC BY

Publication record in CityU Scholars:
Go to record

Published version (DOI):
10.18653/v1/2022.findings-acl.22

Publication details:
A Sentence is Worth 128 Pseudo Tokens: A Semantic-Aware Contrastive Learning Framework for Sentence Embeddings

Haochen Tan¹², Wei Shao¹², Han Wu¹², Ke Yang³, Linqi Song¹²
¹City University of Hong Kong Shenzhen Research Institute
²Department of Computer Science, City University of Hong Kong
³School of Integrated Circuits, Peking University
haochetan2-c@my.cityu.edu.hk
linqi.song@cityu.edu.hk

Abstract

Contrastive learning has shown great potential in unsupervised sentence embedding tasks, e.g., SimCSE (Gao et al., 2021). However, we find that these existing solutions are heavily affected by superficial features like the length of sentences or syntactic structures. In this paper, we propose a semantics-aware contrastive learning framework for sentence embeddings, termed Pseudo-Token BERT (PT-BERT), which is able to exploit the pseudo-token space (i.e., latent semantic space) representation of a sentence while eliminating the impact of superficial features such as sentence length and syntax. Specifically, we introduce an additional pseudo token embedding layer independent of the BERT encoder to map each sentence into a sequence of pseudo tokens in a fixed length. Leveraging these pseudo sequences, we are able to construct same-length positive and negative pairs based on the attention mechanism to perform contrastive learning. In addition, we utilize both the gradient-updating and momentum-updating encoders to encode instances while dynamically maintaining an additional queue to store the representation of sentence embeddings, enhancing the encoder’s learning performance for negative examples. Experiments show that our model outperforms the state-of-the-art baselines on six standard semantic textual similarity (STS) tasks. Furthermore, experiments on alignments and uniformity losses, as well as hard examples with different sentence lengths and syntax, consistently verify the effectiveness of our method.

1 Introduction

Sentence embedding serves as an essential technique in a wide range of applications, including semantic search, text clustering, text classification, etc. (Kiros et al., 2015; Logeswaran and Lee, 2018; Conneau et al., 2017; Cer et al., 2018; Reimers and Gurevych, 2019; Gao et al., 2021). Contrastive learning works on learning representations such that similar examples stay close whereas dissimilar ones are far apart, and thus is suitable for sentence embeddings due to its natural availability of similar examples. Incorporating contrastive learning in sentence embeddings improves the efficiency of semantic information learning in an unsupervised manner (He et al., 2020; Chen et al., 2020) and has been shown to be effective on a variety of tasks (Reimers and Gurevych, 2019; Gao et al., 2021; Zhang et al., 2020).

In contrastive learning for sentence embeddings, a key challenge is constructing positive instances. Both discrete and continuous augmentation methods have been studied recently. Methods in Wu et al. (2018); Meng et al. (2021) perform discrete operations directly on the original sentences, such.
as word deletion and sentence shuffling, to get positive samples. However, these methods may lead to unacceptable semantic distortions or even complete misinterpretations of the original statement. In contrast, the SimCSE method (Gao et al., 2021) obtains two different embeddings in the continuous embedding space as a positive pair for one sentence through different dropout masks (Srivastava et al., 2014) in the neural network for representation learning. Nonetheless, this method overly relies on superficial features existing in the dataset like sentence lengths and syntactic structures and may pay less reflection on meaningful semantic information. As an illustrative example, the sentence-pair in Fig. 1 “A caterpillar was caught by me.” and “I caught a caterpillar.” appear to organize differently in expression but convey exactly the same semantics.

To overcome these drawbacks, in this paper, we propose a semantic-aware contrastive learning framework for sentence embeddings, termed Pseudo-Token BERT (PT-BERT), that is able to capture the pseudo-token space (i.e., latent semantic space) representation while ignoring effects of superficial features like sentence lengths and syntactic structures. Inspired by previous works on prompt learning and sentence selection (Li and Liang, 2021; Liu et al., 2021; Humeau et al., 2020), which create a pseudo-sequence and have it serve the downstream tasks, we present PT-BERT to train pseudo token representations and then to map sentences into pseudo token spaces based on an attention mechanism.

In particular, we train additional 128 pseudo token embeddings, together with sentence embeddings extracted from the BERT model (i.e., gradient-encoder), and then use the attention mechanism (Vaswani et al., 2017) to map the sentence embedding to the pseudo token space (i.e., semantic space). We use another BERT model (i.e., momentum-encoder (He et al., 2020)) to encode the original sentence, adopt a similar attention mechanism with the pseudo token embeddings, and finally output a continuously augmented version of the sentence embedding. We treat the representations of the original sentence encoded by the gradient-encoder and the momentum-encoder as a positive pair. In addition, the momentum-encoder also generates negative examples, dynamically maintains a queue to store these negative examples, and updates them over time. By projecting all sentences onto the same pseudo sentence, the model greatly reduces the dependence on sentence length and syntax when making judgments and makes the model more focused on the semantic level information.

In our experiments, we compare our results with the previous state-of-the-art work. We train PT-BERT on 10^6 randomly sampled sentences from English Wikipedia and evaluate on seven standard semantic textual similarity (STS) tasks (Agirre et al., 2012, 2013, 2014, 2015, 2016) (Marelli et al., 2014). Besides, we also compare our approach with a framework based on an advanced discrete augmentation we proposed. We obtain a new state-of-the-art on standard semantic textual similarity tasks with our PT-BERT, which achieves 77.74% of Spearman’s correlation. To show the effectiveness of pseudo tokens, we calculate the align-loss and uniformity loss (Wang and Isola, 2020) and verify our approach on a sub-dataset with hard examples sampled from STS-(2012-2016). We have released our source code¹ to facilitate future work.

2 Related Work

In this section, we discuss related studies with respect to the contrastive learning framework and sentence embedding.

2.1 Contrastive Learning for Sentence Embedding

Contrastive learning. Contrastive learning (Hadsell et al., 2006) has been used with much success in both natural language processing and computer vision (Yang et al., 2019; Klein and Nabi, 2020; Chen et al., 2020; He et al., 2020; Gao et al., 2021). In contrast to generative learning, contrastive learning requires learning to distinguish and match data at the abstract semantic level of the feature space. It focuses on learning common features between similar examples and distinguishing differences between non-similar examples. In order to compare the instances with more negative examples and less computation, memory bank (Wu et al., 2018) is proposed to enhance the performance under the contrastive learning framework. While with a large capacity to store more samples, the memory bank is not consistent enough, which could not update the negative examples during comparison. Momentum-Contrast (MoCo) (He et al., 2020) uses a queue to maintain the dictionary of samples.

¹https://github.com/Namco0816/PT-BERT
which allows the model to compare the query with more keys for each step and ensure the consistency of the framework. It updates the parameter of the dictionary in a momentum way.

**Discrete and continuous augmentation.** By equipping discrete augmentation that modifies sentences directly on token level with contrastive learning, significant success has been achieved in obtaining sentence embeddings. Such methods include word omission (Yang et al., 2019), entity replacement (Xiong et al., 2020), trigger words (Klein and Nabi, 2020) and traditional augmentations such as deletion, reorder and substitution (Wu et al., 2020; Meng et al., 2021). Examples with diverse expressions can be learned during training, making the model more robust to expressions of different sentence lengths and styles. However, these approaches are limited because there are huge difficulties in augmenting sentences precisely since a few changes can make the meaning completely different or even opposite.

Researchers have also explored the possibility of building sentences continuously, which instead applies operation in embedding space. CTT-BERT (Carlsson et al., 2021) encodes the same sentence with two different encoders. UnsupSimCSE (Gao et al., 2021) compares the representations of the same sentence with different dropout masks among the mini-batch. These approaches continuously augment sentences while retaining the original meaning. However, positive pairs seen by SimCSE always have the same length and structure, whereas negative samples are likely to act oppositely. As a result, sentence length and structure are highly correlated to the similarity score of examples. During training, the model has never seen positive samples with diverse expressions, so that in real test scenarios, the model would be more inclined to classify the synonymous pairs with different expressions as negatives, and those sentences with the same length and structures are more likely to be grouped as positive pairs. This may cause a biased encoder.

### 2.2 Pseudo Tokens

In the domain of prompt learning (Liu et al., 2021; Jiang et al., 2020; Li and Liang, 2021; Gao et al., 2020), the way to create prompt can be divided into two types, namely discrete and continuous ways. Discrete methods usually search the natural language template as the prompt (Davison et al., 2019; Petroni et al., 2019), while the continuous way always directly works on the embedding space with "pseudo tokens" (Liu et al., 2021; Li and Liang, 2021). In retrieval and dialogue tasks, the current approach adopts "pseudo tokens", namely "poly codes" (Humeau et al., 2020), to jointly encode the query and response precisely and ensure the inference time when compared with the Cross-Encoders and Bi-Encoders (Wolf et al., 2019; Mazaré et al., 2018; Dinan et al., 2019). The essence of these methods is to create a pseudo-sequence and have it serve the downstream tasks without the need for humans to understand the exact meaning. The parameters of these pseudo tokens are independent of the natural language embeddings, and can be tuned based on a specific downstream task. In the following sections, we will show the idea to weaken the model’s consideration of sentence length and structures by introducing additional pseudo token embeddings on top of the BERT encoder.

### 3 Methods

In this section, we introduce PT-BERT, which provides novel contributions on combining advantages of both discrete and continuous augmentations to advance the state-of-art of sentence embeddings. We first present the setup of problems with a thorough analysis on the bias introduced by the textual similarity theoretically and experimentally. Then we show the details of Pseudo-Token representation and our model’s architecture.

#### 3.1 Preliminary

Consider a sentence $s$, we say that the augmentation is continuous if $s$ is augmented by different encoding functions, $f(\cdot)$ and $f'(\cdot)$. Sentence em-

<table>
<thead>
<tr>
<th>Sub-dataset</th>
<th>original</th>
</tr>
</thead>
<tbody>
<tr>
<td>STS12</td>
<td>66.54</td>
</tr>
<tr>
<td>STS13</td>
<td>78.50</td>
</tr>
<tr>
<td>STS14</td>
<td>68.76</td>
</tr>
<tr>
<td>STS15</td>
<td>70.27</td>
</tr>
<tr>
<td>STS16</td>
<td>71.31</td>
</tr>
</tbody>
</table>

Table 1: SimCSE’s results on sub-dataset from STS12-16, comparing with original results.

<table>
<thead>
<tr>
<th>SimCSE32</th>
<th>SimCSE64</th>
<th>SimCSE128</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg.</td>
<td>76.25</td>
<td>75.20</td>
</tr>
</tbody>
</table>

Table 2: Different acceptable sequence length of SimCSE would affect the result on STS tasks.
Figure 2: The model is divided into two parts, the upper part (Encoder) updates the learnable parameters with gradient, while the bottom (Momentum Encoder) inherits parameters from the upper part with momentum-updating. We repeatedly input the same sequence of pseudo tokens while processing the original sentences. An additional BERT attention mapping the pooler-output of BERT to pseudo sequence representation, extending the sentence embedding to a fixed length and mapping the syntactic structure to the style of the pseudo sentence. The two attentions in the figure are the same and with identical parameters.

beddings $h = f(s)$ and $h' = f'(s)$ are obtained by these two functions. With a slight change of the encoding function (e.g., encoders with different dropout masks), $h'$ can be seen as a more precisely augmented version of $h$ compared with the discrete augmentation. Semantic information of $h'$ should be the same as $h$. Therefore, $h$ and $h'$ are a pair of positive examples and we could randomly sample a sentence to construct negative example pairs.

Previous state-of-the-art models (Gao et al., 2021) adopt the continuous strategy that augments sentences with dropout (Srivastava et al., 2014). It is obvious that all the positive examples in SimCSE have the same length and structure while negative examples act oppositely. In this way, SimCSE will inevitably take these two factors as hints during test. To further verify this conjecture, we sort out the positive pairs with a length difference of more than five words and negative pairs of less than two words from STS-(2012-2016).

Table 1 shows that the performance of SimCSE plummets on this dataset. Besides, we also find that SimCSE truncates all training corpus into 32 tokens, which shortens the discrepancy of the sentence’s length. After we scale the max length that SimCSE could accept from 32 to 64 and 128, the performance degrades significantly during the test even though the model is supposed to learn more from the complete version of sentences (See Table 2). The reason for this result may lie in the fact that, without truncation, all positive pairs still have the same length, whereas the difference in length between the negative and positive ones is enlarged. Therefore, the encoder will rely more on sentence length and make the wrong decision.

3.2 Pseudo-Token BERT

We realize it is vital to train an unbiased encoder that captures the semantics and also would not introduce intermediate errors. This motivates us to propose the PT-BERT, as evidence shows that the encoder may fail to make predictions when trained on a biased dataset with same-length positive pairs, by learning the spurious correlations that work only well on the training dataset (Arjovsky et al., 2019; Nam et al., 2020).

Pseudo-Token representations. The idea of PT-BERT is to reduce the model’s excessive dependence on textual similarity when making predic-
Discrete augmentation achieves this goal by providing both positive and negative examples with diverse expressions. Therefore the model does not jump to conclusions based on sentence length and syntactic structure during the test.

Note that we achieve this same purpose in a seemingly opposite way: mapping the representations of both positive and negative examples to a pseudo sentence with the same length and structure. We take an additional embedding layer outside the BERT encoder to represent a pseudo sentence \( \{0, 1, ..., m\} \) with fixed length \( m \) and syntax. This embedding layer is fully independent of the BERT encoder, including the parameters and corresponding vocabulary. Random initialization is applied to this layer, and each parameter will be updated during training. The size of this layer depends on the vocabulary of pseudo tokens (length of pseudo sentences). Besides, adopting the attention mechanism (Vaswani et al., 2017; Bahdanau et al., 2015; Gehring et al., 2017), we take the pseudo sentence embeddings as the query states of cross attention while key and value states are the sentence embeddings obtained from the BERT encoder. This allows the pseudo sentence to attend to the core part and ignore the redundant part of original sentence while keeping the fixed length and structure.

Fig. 2 illustrates the framework of PT-BERT. Denoting the pseudo sentence embedding as \( P \) and the sentence embedding encoded by BERT as \( Y \), we obtain the weighted pseudo sentence embedding of each sentence by mapping the sentence embedding to the pseudo tokens with attention:

\[
Z_i' = \text{Attention}(PWQ, Y_i W^K, Y_i W^V) \quad (1)
\]

\[
\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (2)
\]

where \( d_k \) is the dimension of the model, \( W^Q, W^K, W^V \) are the learnable parameters with \( \mathbb{R}^{d_k \times d_k} \), \( Y_i \) denotes the \( i \)-th sentence in the dataset. Then we obtain the final embedding \( h_i \) with the same attention layer by mapping pseudo sentences back to original sentence embeddings:

\[
h_i = \text{Attention}(Y_i W^Q, Z_i' W^K, Z_i' W^V). \quad (3)
\]

Finally, we compare the cosine similarities between the obtained embeddings of \( h \) and \( h' \) using Eq. 4, where \( h' \) are the samples encoded by the momentum-encoder and stored in a queue.

**Model architecture.** Instead of inputting the same sentence twice to the same encoder, we follow the architecture proposed in Momentum-Contrast (MoCo) (He et al., 2020) such that PT-BERT can efficiently learn from more negative examples. Samples in PT-BERT are encoded into vectors with two encoders: gradient-update encoder (the upper encoder in Fig. 2) and momentum-update encoder (the momentum encoder in Fig. 2). We dynamically maintain a queue to store the sentence representations from momentum-update encoder.

This mechanism allows us to store as much negative samples as possible without re-computation. Once the queue is full, we replace the "oldest" negative sample with a "fresh" one encoded by the momentum-encoder.

Similar to the works based on continuous augmentation, at the very beginning of the framework, PT-BERT takes input sentence \( s \) and obtains \( h_i \) and \( h_i' \) with two different encoder functions. We measure the loss function with:

\[
\ell_i = -\log \frac{e^{\text{sim}(h_i, h_i')/\tau}}{\sum_j e^{\text{sim}(h_i, h_j')/\tau}}, \quad (4)
\]

where \( h_i \) denotes the representations extracted from the gradient-update encoder, \( h_i' \) represents the sentence embedding in the queue, and \( M \) is the queue size. Our gradient-update and momentum-update encoder are based on the pre-trained language model with the same structure and dimensions as BERT-base-uncased (Devlin et al., 2019). The momentum encoder will update its parameters similar to MoCo:

\[
\theta_k \leftarrow \lambda \theta_k + (1 - \lambda) \theta_q, \quad (5)
\]

where \( \theta_k \) is the parameter of the momentum-contrast encoder that maintains the dictionary, \( \theta_q \) is the query encoder that updates the parameters with gradients, and \( \lambda \) is a hyperparameter used to control the updating process.

**Relationship with prompt learning.** Rather than directly perform soft prompting in the embedding space (Li and Liang, 2021; Qin and Eisner, 2021; Liu et al., 2021) of the model, our method follows the "plug and play" fashion that project the representations to pseudo sentences only during the period of training. During inference time, PT-BERT predicts the results only with its BERT backbone. Our original intention of designing this procedure is to make the model predict sentence
Table 3: Sentence embedding performance on STS tasks with Spearman’s correlation measured. We highlight the highest number for each methods. CLEAR (Wu et al., 2020) is trained on both English Wikipedia and Book Corpus with 500k steps with their own version of pre-trained models. Result of CT-BERT (Carlsson et al., 2021) is based on the settings of SimCSE (Gao et al., 2021)

<table>
<thead>
<tr>
<th>Model</th>
<th>STS12</th>
<th>STS13</th>
<th>STS14</th>
<th>STS15</th>
<th>STS16</th>
<th>STS-B</th>
<th>SICK-R</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrete Augmentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLEAR</td>
<td>49.00</td>
<td>48.90</td>
<td>57.40</td>
<td>63.60</td>
<td>65.60</td>
<td>72.50</td>
<td>75.60</td>
<td>61.80</td>
</tr>
<tr>
<td>MoCo</td>
<td>68.35</td>
<td>81.42</td>
<td>73.34</td>
<td>81.63</td>
<td>78.61</td>
<td>76.40</td>
<td>68.50</td>
<td>75.46</td>
</tr>
<tr>
<td>MoCo+reorder</td>
<td>66.14</td>
<td>80.06</td>
<td>73.14</td>
<td>81.35</td>
<td>76.01</td>
<td>73.99</td>
<td>65.76</td>
<td>73.78</td>
</tr>
<tr>
<td>MoCo+duplication</td>
<td>65.88</td>
<td>82.24</td>
<td>73.34</td>
<td>81.49</td>
<td>77.48</td>
<td>76.29</td>
<td>68.86</td>
<td>75.08</td>
</tr>
<tr>
<td>MoCo+deletion</td>
<td>67.86</td>
<td>81.43</td>
<td>72.8</td>
<td>81.48</td>
<td>77.84</td>
<td>76.91</td>
<td>69.46</td>
<td>75.40</td>
</tr>
<tr>
<td>MoCo+SRL</td>
<td>68.92</td>
<td>82.20</td>
<td>73.67</td>
<td>81.58</td>
<td>78.73</td>
<td>77.63</td>
<td>71.07</td>
<td>76.26</td>
</tr>
<tr>
<td>Continuous Augmentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CT-BERT</td>
<td>61.63</td>
<td>76.80</td>
<td>68.47</td>
<td>77.50</td>
<td>76.48</td>
<td>74.31</td>
<td>69.19</td>
<td>72.05</td>
</tr>
<tr>
<td>SimCSE-BERT&lt;sub&gt;base&lt;/sub&gt; + None</td>
<td>68.40</td>
<td>82.41</td>
<td>74.38</td>
<td>80.91</td>
<td>78.56</td>
<td>76.85</td>
<td>72.23</td>
<td>76.25</td>
</tr>
<tr>
<td>SimCSE-BERT&lt;sub&gt;base&lt;/sub&gt; + Crop</td>
<td>77.80</td>
<td>75.90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PT-BERT&lt;sub&gt;base&lt;/sub&gt;</td>
<td>71.20</td>
<td>83.76</td>
<td>76.34</td>
<td>82.63</td>
<td>78.90</td>
<td>79.42</td>
<td>71.94</td>
<td>77.74</td>
</tr>
</tbody>
</table>

Table 4: Results on STS-B development sets. Results of SimCSE (Gao et al., 2021) are reported from original paper.

4 Experiments

In this section, we perform the standard semantic textual similarity (STS) (Agris et al., 2012, 2013, 2014, 2015, 2016) tasks to test our model. For all tasks, we measure the Spearman’s correlation to compare our performance with the previous state-of-the-art SimCSE (Gao et al., 2021). In the following, we will describe the training procedure in detail.

4.1 Training Data and Settings

Datasets. Following SimCSE, We train our model on 1-million sentences randomly sampled from English Wikipedia, and evaluate the model every 125 steps to find the best checkpoints. Note that we do not fine-tune our model on any dataset, which indicates that our method is completely unsupervised.

Hardware and schedule. We train our model on the machine with one NVIDIA V100s GPU. Following the settings of SimCSE (Gao et al., 2021), it takes 50 minutes to run an epoch.

4.2 Implementations

We implement PT-BERT based on Huggingface transformers (Wolf et al., 2020) and initialize it with the released BERT<sub>base</sub> (Devlin et al., 2019). We initialize a new embedding for pseudo tokens with $128 \times 768$. During training, we create a pseudo sentence $\{0, 1, 2, ..., 127\}$ for every input and map the original sentence to this pseudo sentence by attention. With batches of 64 sentences and an additional dynamically maintained queue of 256 sentences, each sentence has one positive sample and 255 negative samples. Adam (Kingma and Ba,
The 2014 optimizer is used to update the model parameters. We also take the original dropout strategy of BERT with rate $p = 0.1$. We set the momentum for the momentum-encoder with $\lambda = 0.885$.

### 4.3 Evaluation Setup

We evaluate the fine-tuned BERT encoder on STS-B development sets every 125 steps to select the best checkpoints. We report all the checkpoints based on the evaluation results reported in Table 4. The training process is fully unsupervised since no training corpus from STS is used. During the evaluation, we also calculate the trends of alignment-loss and uniformity-loss. Losses were compared with SimCSE (Gao et al., 2021) under the same experimental settings. After training and evaluation, we test models on 7 STS tasks: STS 2012-2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016), STS Benchmark (Cer et al., 2017) and SICK-Relatedness (Marelli et al., 2014). We report the result of Spearman’s correlation for all the experiments.

### 4.4 Main Results and Analysis

We first compare PT-BERT with our baseline: MoCo framework + BERT encoder (MoCo-BERT). MoCo-BERT could be seen as a version of PT-BERT without pseudo token embeddings. Then we apply traditional discrete augmentations such as reorder, duplication, and deletion on this framework. We also compare our work with CLEAR (Wu et al., 2020) that substitutes and deletes the token spans. Besides, we argue that the performance of these methods is too weak. We additionally propose an advanced discrete augmentation approach that produces positive examples with the guidance of Semantic Role Labeling (SRL) (Gildea and Jurafsky, 2002; Palmer et al., 2010) information, instead of random deletion and reordering. SRL-guided augmentation could compensate the errors caused by these factors, acting as a combination of deletion, duplication, and reordering with better accuracy. SRL is broadly used to identify the predicate-argument structures of a sentence, it detects the arguments associated with the predicate or verb of a sentence and could indicate the main semantic information of who did what to whom. For the sentences with multiple predicates, we keep all the sets with order [ARG0, PRED, ARGM = NEG, ARG1] and concatenate them into a new sequence. For the sentences without recognized predicate-argument sets, we keep the original sentence as positive examples. In addition to the work based on discrete approaches, we also compare with SimCSE (Gao et al., 2021) which continuously augments sentences with dropout. In Table 3, PT-BERT with 128 pseudo tokens further pushed the state-of-the-art results to 77.74% and significantly outperformed SimCSE over six datasets.

In Fig 3, we observe that PT-BERT also achieves better alignment and uniformity against SimCSE, which indicates that pseudo tokens really help the learning of sentence representations. In detail, alignment and uniformity are proposed by (Wang and Isola, 2020) to evaluate the quality of representations in contrastive learning. The calculation of these two metrics are shown in the following formulas:

$$L_{\text{alignment}} = \mathbb{E}_{(x, x^+) \sim \rho_{\text{pos}}} ||f(x) - f(x^+)||^2,$$ (6)
Table 5: Evaluation results of ablation studies and hard sentence pairs.

<table>
<thead>
<tr>
<th>Method</th>
<th>STS12</th>
<th>STS13</th>
<th>STS14</th>
<th>STS15</th>
<th>STS16</th>
<th>STS-B</th>
<th>SICK-R</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Ablation studies on pseudo sequence length</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L-64</td>
<td>67.04</td>
<td>82.04</td>
<td>73.65</td>
<td>81.12</td>
<td>78.64</td>
<td>77.35</td>
<td>71.33</td>
<td>75.88</td>
</tr>
<tr>
<td>L-90</td>
<td>68.94</td>
<td>84.08</td>
<td>74.53</td>
<td>81.22</td>
<td>79.06</td>
<td>78.01</td>
<td>71.49</td>
<td>76.48</td>
</tr>
<tr>
<td>L-128(Ours)</td>
<td>71.20</td>
<td>83.76</td>
<td>76.34</td>
<td>82.63</td>
<td>78.90</td>
<td>79.42</td>
<td>71.94</td>
<td>77.74</td>
</tr>
<tr>
<td>L-256</td>
<td>67.09</td>
<td>82.25</td>
<td>72.63</td>
<td>81.48</td>
<td>78.55</td>
<td>77.30</td>
<td>69.53</td>
<td>75.55</td>
</tr>
<tr>
<td>L-360</td>
<td>68.90</td>
<td>82.21</td>
<td>73.77</td>
<td>81.31</td>
<td>77.50</td>
<td>77.22</td>
<td>69.32</td>
<td>75.75</td>
</tr>
<tr>
<td>(b) Ablation studies on queue size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q-192</td>
<td>70.29</td>
<td>83.78</td>
<td>75.98</td>
<td>82.13</td>
<td>78.48</td>
<td>78.91</td>
<td>72.53</td>
<td>77.44</td>
</tr>
<tr>
<td>Q-256(Ours)</td>
<td>71.20</td>
<td>83.76</td>
<td>76.34</td>
<td>82.63</td>
<td>78.90</td>
<td>79.42</td>
<td>71.94</td>
<td>77.74</td>
</tr>
<tr>
<td>Q-320</td>
<td>71.71</td>
<td>83.36</td>
<td>75.00</td>
<td>82.99</td>
<td>78.76</td>
<td>79.17</td>
<td>72.85</td>
<td>77.69</td>
</tr>
<tr>
<td>(c) Evaluations on hard sentence pairs with different length</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SimCSE</td>
<td>66.54</td>
<td>78.50</td>
<td>68.76</td>
<td>70.27</td>
<td>71.31</td>
<td>-</td>
<td>-</td>
<td>71.08</td>
</tr>
<tr>
<td>PT-BERT</td>
<td>72.02</td>
<td>80.24</td>
<td>72.92</td>
<td>74.50</td>
<td>72.50</td>
<td>-</td>
<td>-</td>
<td>74.44</td>
</tr>
</tbody>
</table>

5 Analysis

5.1 Ablation Studies

In this section, we first investigate the impact of different sizes of pseudo token embeddings. Then we would like to report the performance difference caused by queue size under the MoCo framework.

Pseudo Sentence Length Different lengths of pseudo tokens can affect the ability of the model to express the sentence representations. By mapping the original sentences to various lengths of pseudo tokens, the performance of PT-BERT could be different. In this section, we keep all the parts except the pseudo tokens and their embeddings unchanged. We scale the pseudo sequence length from 64 to 360. Table 5(a) shows a comparison between different lengths of pseudo sequence in PT-BERT. We find that during training, PT-BERT performs better when attending to pseudo sequences with 128 tokens. Too few pseudo tokens do not fully explain the semantics of the original sentence, while too many pseudo tokens increase the number of parameters and over-express the sentence.

Queue Size The introduction of more negative samples would make the model’s training more reliable. By training with different queue sizes, we report the result of PT-BERT with different performances due to the number of negative samples. In Table 5(b), queue size $q = 4$ performs best. However, the difference in performance between the three sets of experiments is not large, suggesting that the model can learn well as long as it can see enough negative samples.

5.2 Exploration on Hard Examples with Different Length

To prove the effectiveness of PT-BERT that could weaken the hints caused by textual similarity, we further test PT-BERT on the sub-dataset introduced in Sec. 3.1. We sorted out the positive pairs with

\[
L_{uniformity} = \log \frac{1}{(x,y) \sim \mathcal{P}_{data}} e^{-2||f(x) - f(y)||^2},
\]

where $(x, x^+)$ is the positive pair, $(x, y)$ is the pair consisting of any two different sentences in the whole sentence set, $f(x)$ is the normalized representation of $x$. We employ the final embedding $h$ to calculate these scores.

According to the above formulas, lower alignment loss means a shorter distance between the positive samples, and low uniformity loss implies the diversity of embeddings of all sentences. Both are our expectations for the representations based on contrastive learning. To evaluate our model’s performance on alignment and uniformity, we compare it with SimCSE on the STS-benchmark dataset (Cer et al., 2017), and the result is shown in Figure 3. The result demonstrates that PT-BERT outperforms SimCSE on these two metrics: our model has a lower alignment and uniformity than SimCSE in almost all the training steps, which indicates that the representations produced by our model are more in line with the goal of the contrastive learning.

$L_{uniformity} = \log \frac{1}{(x,y) \sim \mathcal{P}_{data}} e^{-2||f(x) - f(y)||^2},$ (7)

where $(x, x^+)$ is the positive pair, $(x, y)$ is the pair consisting of any two different sentences in the whole sentence set, $f(x)$ is the normalized representation of $x$. We employ the final embedding $h$ to calculate these scores.

According to the above formulas, lower alignment loss means a shorter distance between the positive samples, and low uniformity loss implies the diversity of embeddings of all sentences. Both are our expectations for the representations based on contrastive learning. To evaluate our model’s performance on alignment and uniformity, we compare it with SimCSE on the STS-benchmark dataset (Cer et al., 2017), and the result is shown in Figure 3. The result demonstrates that PT-BERT outperforms SimCSE on these two metrics: our model has a lower alignment and uniformity than SimCSE in almost all the training steps, which indicates that the representations produced by our model are more in line with the goal of the contrastive learning.
a length difference of more than five words and negative pairs of less than two words from STS-(2012-2016). PT-BERT significantly outperforms SimCSE with 3.36% Spearman’s correlation, indicating that PT-BERT could handle these hard examples better than SimCSE. This further proves that PT-BERT could debias the spurious correlation introduced by sentence length and syntax, and focus more on the semantics.

6 Conclusion
In this paper, we propose a semantic-aware contrastive learning framework for sentence embeddings, termed PT-BERT. Our proposed PT-BERT approach is able to weaken textual similarity information, such as sentence length and syntactic structures, by mapping the original sentence to a fixed pseudo sentence embedding. We provide analysis of these factors on methods based on continuous and discrete augmentation, showing that PT-BERT augments sentences more accurately than discrete methods while considering more semantics instead of textual similarity than continuous approaches. Lower uniformity loss and alignment loss prove the effectiveness of PT-BERT and further experiments also show that PT-BERT could handle hard examples better than existing approaches.

Providing a new perspective to the continuous data augmentation in sentence embeddings, we believe our proposed PT-BERT has great potential to be applied in broader downstream applications, such as text classification, text clustering, and sentiment analysis.

Acknowledgements
We would like to thanks the anonymous reviewers for their valuable and constructive comments. This work was supported in part by the Hong Kong RGC grant ECS 21212419, Technological Breakthrough Project of ECS, Technology and Innovation Commission of Shenzhen Municipality under Grants JSGG2020110216200001, the Hong Kong Laboratory for AI-Powered Financial Technologies, the CityU Teaching Development Grants under 6000755, and the UGC Special Virtual Teaching and Learning Grants under 6430300.

References


Yu Meng, Chenyan Xiong, Payal Bajaj, Saurabh Tiwary, Paul Bennett, Jiawei Han, and Xia Song. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining.


