Self-Guided Contrastive Learning for BERT Sentence Representations

Taeuk Kim†*, Kang Min Yoo‡, and Sang-goo Lee†
†Dept. of Computer Science and Engineering, Seoul National University, Seoul, Korea
‡NAVER AI Lab, Seongnam, Korea
{taeuk, sglee}@europa.snu.ac.kr, kangmin.yoo@navercorp.com

Abstract

Although BERT and its variants have reshaped the NLP landscape, it still remains unclear how best to derive sentence embeddings from such pre-trained Transformers. In this work, we propose a contrastive learning method that utilizes self-guidance for improving the quality of BERT sentence representations. Our method fine-tunes BERT in a self-supervised fashion, does not rely on data augmentation, and enables the usual [CLS] token embeddings to function as sentence vectors. Moreover, we redesign the contrastive learning objective (NT-Xent) and apply it to sentence representation learning. We demonstrate with extensive experiments that our approach is more effective than competitive baselines on diverse sentence-related tasks. We also show it is efficient at inference and robust to domain shifts.

1 Introduction

Pre-trained Transformer (Vaswani et al., 2017) language models such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) have been integral to achieving recent improvements in natural language understanding. However, it is not straightforward to directly utilize these models for sentence-level tasks, as they are basically pre-trained to focus on predicting (sub)word tokens given context. The most typical way of converting the models into sentence encoders is to fine-tune them with supervision from a downstream task. In the process, as initially proposed by Devlin et al. (2019), a pre-defined token’s (a.k.a. [CLS]) embedding from the last layer of the encoder is deemed as the representation of an input sequence. The most typical way of converting the models into sentence encoders is to fine-tune them with supervision from a downstream task. In the process, as initially proposed by Devlin et al. (2019), a pre-defined token’s (a.k.a. [CLS]) embedding from the last layer of the encoder is deemed as the representation of an input sequence. This simple but effective method is possible because, during supervised fine-tuning, the [CLS] embedding functions as the only communication gate between the pre-trained encoder and a task-specific layer, encouraging the [CLS] vector to capture the holistic information.

On the other hand, in cases where labeled datasets are unavailable, it is unclear what the best strategy is for deriving sentence embeddings from BERT. In practice, previous studies (Reimers and Gurevych, 2019; Li et al., 2020; Hu et al., 2020) reported that naively (i.e., without any processing) leveraging the [CLS] embedding as a sentence representation, as is the case of supervised fine-tuning, results in disappointing outcomes. Currently, the most common rule of thumb for building BERT sentence embeddings without supervision is to apply mean pooling on the last layer(s) of BERT.

In this paper, the term BERT has two meanings: Narrowly, the BERT model itself, and more broadly, pre-trained Transformer encoders that share the same spirit with BERT.
Yet, this approach can be still sub-optimal. In a preliminary experiment, we constructed sentence embeddings by employing various combinations of different BERT layers and pooling methods, and tested them on the Semantic Textual Similarity (STS) benchmark dataset (Cer et al., 2017).\(^2\) We discovered that BERT(-base)'s performance, measured in Spearman correlation (× 100), can range from as low as 16.71 ([CLS], the 10\(^{th}\) layer) to 63.19 (max pooling, the 2\(^{nd}\) layer) depending on the selected layer and pooling method (see Figure 1). This result suggests that the current practice of building BERT sentence vectors is not solid enough, and that there is room to bring out more of BERT’s expressiveness.

In this work, we propose a contrastive learning method that makes use of a newly proposed self-guidance mechanism to tackle the aforementioned problem. The core idea is to recycle intermediate BERT hidden representations as positive samples to which the final sentence embedding should be close. As our method does not require data augmentation, which is essential in most recent contrastive learning frameworks, it is much simpler and easier to use than existing methods (Fang and Xie, 2020; Xie et al., 2020). Moreover, we customize the NT-Xent loss (Chen et al., 2020), a contrastive learning objective widely used in computer vision, for better sentence representation learning with BERT. We demonstrate that our approach outperforms competitive baselines designed for building BERT sentence vectors (Li et al., 2020; Wang and Kuo, 2020) in various environments. With comprehensive analyses, we also show that our method is more computationally efficient than the baselines at inference in addition to being more robust to domain shifts.

2 Related Work

Contrastive Representation Learning. Contrastive learning has been long considered as effective in constructing meaningful representations. For instance, Mikolov et al. (2013) propose to learn word embeddings by framing words nearby a target word as positive samples while others as negative. Logeswaran and Lee (2018) generalize the approach of Mikolov et al. (2013) for sentence representation learning. More recently, several studies (Fang and Xie, 2020; Giorgi et al., 2020; Wu et al., 2020) suggest to utilize contrastive learning for training Transformer models, similar to our approach. However, they generally require data augmentation techniques, e.g., back-translation (Sennrich et al., 2016), or prior knowledge on training data such as order information, while our method does not. Furthermore, we focus on revising BERT for computing better sentence embeddings rather than training a language model from scratch.

On the other hand, contrastive learning has been also receiving much attention from the computer vision community (Chen et al. (2020); Chen and He (2020); He et al. (2020), inter alia). We improve the framework of Chen et al. (2020) by optimizing its learning objective for pre-trained Transformer-based sentence representation learning. For extensive surveys on contrastive learning, refer to Le-Khac et al. (2020) and Jaiswal et al. (2020).

Fine-tuning BERT with Supervision. It is not always trivial to fine-tune pre-trained Transformer models of gigantic size with success, especially when the number of target domain data is limited (Mosbach et al., 2020). To mitigate this training instability problem, several approaches (Aghajanyan et al., 2020; Jiang et al., 2020; Zhu et al., 2020) have been recently proposed. In particular, Gunel et al. (2021) propose to exploit contrastive learning as an auxiliary training objective during fine-tuning BERT with supervision from target tasks. In contrast, we deal with the problem of adjusting BERT when such supervision is not available.

Sentence Embeddings from BERT. Since BERT and its variants are originally designed to be fine-tuned on each downstream task to attain their optimal performance, it remains ambiguous how best to extract general sentence representations from them, which are broadly applicable across diverse sentence-related tasks. Following Conneau et al. (2017), Reimers and Gurevych (2019) (SBERT) propose to compute sentence embeddings by conducting mean pooling on the last layer of BERT and then fine-tuning the pooled vectors on the natural language inference (NLI) datasets (Bowman et al., 2015; Williams et al., 2018). Meanwhile, some other studies concentrate on more effectively leveraging the knowledge embedded in BERT to construct sentence embeddings without supervision. Specifically, Wang and Kuo (2020) propose a pooling method based on linear algebraic algorithms to draw sentence vectors from BERT’s intermediate layers. Li et al. (2020) suggest to learn a
mapping from the average of the embeddings obtained from the last two layers of BERT to a spherical Gaussian distribution using a flow model, and to leverage the redistributed embeddings in place of the original BERT representations. We follow the setting of Li et al. (2020) in that we only utilize plain text during training, however, unlike all the others that rely on a certain pooling method even after training, we directly refine BERT so that the typical \texttt{[CLS]} vector can function as a sentence embedding. Note also that there exists concurrent work (Carlsson et al., 2021; Gao et al., 2021; Wang et al., 2021) whose motivation is analogous to ours, attempting to improve BERT sentence embeddings in an unsupervised fashion.

3 Method

As BERT mostly requires some type of adaptation to be properly applied to a task of interest, it might not be desirable to derive sentence embeddings directly from BERT without fine-tuning. While Reimers and Gurevych (2019) attempt to alleviate this problem with typical supervised fine-tuning, we restrict ourselves to revising BERT in an unsupervised manner, meaning that our method only demands a bunch of raw sentences for training.

Among possible unsupervised learning strategies, we concentrate on contrastive learning which can inherently motivate BERT to be aware of similarities between different sentence embeddings. Considering that sentence vectors are widely used in computing the similarity of two sentences, the inductive bias introduced by contrastive learning can be helpful for BERT to work well on such tasks. The problem is that sentence-level contrastive learning usually requires data augmentation (Fang and Xie, 2020) or prior knowledge on training data, e.g., order information (Logeswaran and Lee, 2018), to make plausible positive/negative samples. We attempt to circumvent these constraints by utilizing the hidden representations of BERT, which are readily accessible, as samples in the embedding space.

3.1 Contrastive Learning with Self-Guidance

We aim at developing a contrastive learning method that is free from external procedure such as data augmentation. A possible solution is to leverage (virtual) adversarial training (Miyato et al., 2018) in the embedding space. However, there is no assurance that the semantics of a sentence embedding would remain unchanged when it is added with a random noise. As an alternative, we propose to utilize the hidden representations from BERT’s intermediate layers, which are conceptually guaranteed to represent corresponding sentences, as pivots that BERT sentence vectors should be close to or be away from. We call our method as self-guided contrastive learning since we exploit internal training signals made by BERT itself to fine-tune it.

We describe our training framework in Figure 2. First, we clone BERT into two copies, BERT\(_F\) (fixed) and BERT\(_T\) (tuned) respectively. BERT\(_F\) is fixed during training to provide a training signal while BERT\(_T\) is fine-tuned to construct better sentence embeddings. The reason why we differentiate BERT\(_F\) from BERT\(_T\) is that we want to prevent the training signal computed by BERT\(_F\) from being degenerated as the training procedure continues, which often happens when BERT\(_F\) = BERT\(_T\). This design decision also reflects our philosophy that our goal is to dynamically conflate the knowledge stored in BERT’s different layers to produce sentence embeddings, rather than introducing new information via extra training. Note that in our setting, the \texttt{[CLS]} vector from the last layer of BERT\(_T\), i.e., \(c_i\), is regarded as the final sentence embedding we aim to optimize/utilize during/after fine-tuning.

Second, given \(b\) sentences in a mini-batch, say \(s_1, s_2, \cdots, s_b\), we feed each sentence \(s_i\) into BERT\(_F\) and compute token-level hidden representations \(H_{i,k} \in \mathbb{R}^{\text{len}(s_i) \times d}\):

\[
[H_{i,0}; H_{i,1}; \cdots; H_{i,k}; \cdots; H_{i,l}] = \text{BERT}_F(s_i),
\]

![Figure 2: Self-guided contrastive learning framework.](image)
where $0 \leq k \leq l$ ($0$: the non-contextualized layer), $l$ is the number of hidden layers in BERT, $\text{len}(s_i)$ is the length of the tokenized sentence, and $d$ is the size of BERT’s hidden representations. Then, we apply a pooling function $p$ to $H_{i,k}$ for deriving diverse sentence-level views $h_{i,k} \in \mathbb{R}^d$ from all layers, i.e., $h_{i,k} = p(H_{i,k})$. Finally, we choose the final view to be utilized by applying a sampling function $\sigma$:

$$h_i = \sigma(\{h_{i,k} | 0 \leq k \leq l\}).$$

As we have no specific constraints in defining $p$ and $\sigma$, we employ max pooling as $p$ and a uniform sampler as $\sigma$ for simplicity, unless otherwise stated. This simple choice for the sampler implies that each $h_{i,k}$ has the same importance, which is persuasive considering it is known that different BERT layers are specialized at capturing disparate linguistic concepts (Jawahar et al., 2019).³

Third, we compute our sentence embedding $c_i$ for $s_i$ as follows:

$$c_i = \text{BERT}_T(s_i)_{[CLS]},$$

where $\text{BERT}(\cdot)_{[CLS]}$ corresponds to the $[CLS]$ vector obtained from the last layer of BERT. Next, we collect the set of the computed vectors into $X = \{x|x \in \{c_i\} \cup \{h_i\}\}$, and for all $x_m \in X$, we compute the NT-Xent loss (Chen et al., 2020):

$$L_m^\text{base} = - \log(\phi(x_m, \mu(x_m))/Z),$$

where $\phi(u, v) = \exp(g(f(u), f(v))/\tau)$ and $Z = \sum_{n=1, n \neq m}^{2b} \phi(x_m, x_n)$.

Note that $\tau$ is a temperature hyperparameter, $f$ is a projection head consisting of MLP layers,⁴ $g(u, v) = u \cdot v/\|u\|\|v\|$ is the cosine similarity function, and $\mu(\cdot)$ is the matching function defined as follows,

$$\mu(x) = \begin{cases} h_i & \text{if } x \text{ is equal to } c_i, \\
 c_i & \text{if } x \text{ is equal to } h_i. \end{cases}$$

Lastly, we sum all $L_m^\text{base}$ divided by $2b$, and add a regularizer $L^\text{reg} = \|\text{BERT}_F - \text{BERT}_T\|^2_2$ to prevent $\text{BERT}_T$ from being too distant from $\text{BERT}_F$.⁵

As a result, the final loss $L^\text{base}$ is:

$$L^\text{base} = \frac{1}{2b} \sum_{m=1}^{2b} L_m^\text{base} + \lambda \cdot L^\text{reg},$$

where the coefficient $\lambda$ is a hyperparameter.

To summarize, our method refines BERT so that the sentence embedding $c_i$ has a higher similarity with $h_i$, which is another representation for the sentence $s_i$, in the subspace projected by $f$ while being relatively dissimilar with $c_j$ when $i \neq j$. After training is completed, we remove all the components except $\text{BERT}_T$ and simply use $c_i$ as the final sentence representation.

### 3.2 Learning Objective Optimization

In Section 3.1, we relied on a simple variation of the general NT-Xent loss, which is composed of four factors. Given sentence $s_i$ and $s_j$ without loss of generality, the factors are as follows (Figure 3):

1. $c_i \rightarrow h_i$ (or $c_j \rightarrow h_j$): The main component that mirrors our core motivation that a BERT sentence vector ($c_i$) should be consistent with intermediate views ($h_i$) from BERT.
2. $c_i \leftarrow c_j$: A factor that forces sentence embeddings ($c_i, c_j$) to be distant from each other.
3. $c_i \leftarrow h_j$ (or $c_j \leftarrow h_i$): An element that makes $c_i$ being inconsistent with views for other sentences ($h_j$).
4. $h_i \leftarrow h_j$: A factor that causes a discrepancy between views of different sentences ($h_i, h_j$).

Even though all the four factors play a certain role, some components may be useless or even cause a negative influence on our goal. For instance, Chen and He (2020) have recently reported that in image representation learning, only (1) is vital while others are nonessential. Likewise, we customize the

![Figure 3: Four factors of the original NT-Xent loss. Green and yellow arrows represent the force of attraction and repulsion, respectively. Best viewed in color.](image-url)
training loss with three major modifications so that it can be more well-suited for our purpose.

First, as our aim is to improve $\mathbf{c}_i$ with the aid of $\mathbf{h}_i$, we re-define our loss focusing more on $\mathbf{c}_i$ rather than considering $\mathbf{c}_i$ and $\mathbf{h}_i$ as equivalent entities:

$$L_{i}^{opt1} = -\log(\hat{Z}^2) = -\log\left(\frac{\phi(\mathbf{c}_i, \mathbf{h}_i)}{\mathbf{h}_i} Z\right),$$

where $\hat{Z} = \sum_{b=1}^{B} \phi(\mathbf{c}_i, \mathbf{c}_j) + \sum_{b=1}^{B} \phi(\mathbf{c}_i, \mathbf{h}_j)$.

In other words, $\mathbf{h}_i$ only functions as points that $\mathbf{c}_i$ is encouraged to be close to or away from, and is not deemed as targets to be optimized. This revision naturally results in removing (4). Furthermore, we discover that (2) is also insignificant for improving performance, and thus derive $L_{i}^{opt2}$:

$$L_{i}^{opt2} = -\log(\phi(\mathbf{c}_i, \mathbf{h}_i) / \sum_{j=1}^{b} \phi(\mathbf{c}_i, \mathbf{h}_j)).$$

Lastly, we diversify signals from (1) and (3) by allowing multiple views $\{\mathbf{h}_{i,k}\}$ to guide $c_i$:

$$L_{i,k}^{opt3} = -\log \left( \frac{\phi(\mathbf{c}_i, \mathbf{h}_{i,k}) + \sum_{m=1}^{n} \phi(\mathbf{c}_i, \mathbf{h}_{i,m})}{\phi(\mathbf{c}_i, \mathbf{h}_{i,k}) + \sum_{m=1}^{n} \phi(\mathbf{c}_i, \mathbf{h}_{i,m})} \right).$$

We expect with this refinement that the learning objective can provide more precise and fruitful training signals by considering additional (and freely available) samples being provided with. The final form of our optimized loss is:

$$L_{i}^{opt} = \frac{1}{b(l+1)} \sum_{i=1}^{b} \sum_{k=0}^{l} L_{i,k}^{opt3} + \lambda \cdot L^{reg}.$$ 

In Section 5.1, we show the decisions made in this section contribute to improvements in performance.

4 Experiments

4.1 General Configurations

In terms of pre-trained encoders, we leverage BERT (Devlin et al., 2019) for English datasets and MBERT, which is a multilingual variant of BERT, for multilingual datasets. We also employ RoBERTa (Liu et al., 2019) and SBERT (Reimers and Gurevych, 2019) in some cases to evaluate the generalizability of tested methods. We use the suffixes `-base` and `-large` to distinguish small and large models. Every trainable model’s performance is reported as the average of 8 separate runs to reduce randomness. Hyperparameters are optimized on the STS-B validation set using BERT-base and utilized across different models. See Table 8 in Appendix A.1 for details. Our implementation is based on the HuggingFace’s Transformers (Wolf et al., 2019) and SBERT (Reimers and Gurevych, 2019) library, and publicly available at https://github.com/galsang/SG-BERT.

4.2 Semantic Textual Similarity Tasks

We first evaluate our method and baselines on Semantic Textual Similarity (STS) tasks. Given two sentences, we derive their similarity score by computing the cosine similarity of their embeddings.

Datasets and Metrics. Following the literature, we evaluate models on 7 datasets in total, that is, STS-B (Cer et al., 2017), SICK-R (Marelli et al., 2014), and STS12-16 (Agirre et al., 2012, 2013, 2014, 2015, 2016). These datasets contain pairs of two sentences, whose similarity scores are labeled from 0 to 5. The relevance between gold annotations and the scores predicted by sentence vectors is measured in Spearman correlation ($\times 100$).

Baselines and Model Specification. We first prepare two non-BERT approaches as baselines, i.e., Glove (Pennington et al., 2014) mean embeddings and Universal Sentence Encoder (USE; Cer et al. (2018)). In addition, various methods for BERT sentence embeddings that do not require supervision are also introduced as baselines:

- **CLS** token embedding: It regards the [CLS] vector from the last layer of BERT as a sentence representation.
- **Mean** pooling: This method conducts mean pooling on the last layer of BERT and use the output as a sentence embedding.
- **WK** pooling: This follows the method of Wang and Kuo (2020), which exploits QR decomposition and extra techniques to derive meaningful sentence vectors from BERT.
- **Flow**: This is BERT-flow proposed by Li et al. (2020), which is a flow-based model that maps the vectors made by taking mean pooling on the last two layers of BERT to a Gaussian space.\(^6\)
- **Contrastive (BT)**: Following Fang and Xie (2020), we revise BERT with contrastive learning. However, this method relies on back-translation to obtain positive samples, unlike ours. Details about this baseline are specified in Appendix A.2.

We make use of plain sentences from STS-B to fine-tune BERT using our approach, identical with Flow.\(^7\) We name the BERT instances trained with our self-guided method as **Contrastive (SG)** and

\(^6\)We restrictively utilize this model, as we find it difficult to exactly reproduce the model’s result with its official code.

\(^7\)For training, Li et al. (2020) utilize the concatenation of the STS-B training, validation, and test set (without gold annotations). We also follow the same setting for a fair comparison.
To our surprise, WK pooling’s performance is even better than back-translation. It is also worth mentioning that it performs generally better than competitive baselines when directly compared, forming the simple strategies. Nevertheless, its performance is shown to be worse than that of our self-guidance algorithm rather than back-translation. It is also worth mentioning that the optimized version of our method (SG-OPT) generally shows better performance than the basic one (SG), proving the efficacy of learning objective optimization (Section 3.2). To conclude, we demonstrate that our self-guided contrastive learning is effective in improving the quality of BERT sentence embeddings when tested on STS tasks.

### 4.3 Multilingual STS Tasks

We expand our experiments to multilingual settings by utilizing MBERT and cross-lingual zero-shot transfer. Specifically, we refine MBERT using only

| Models          | STS-B | SICK-R | STS12 | STS13 | STS14 | STS15 | STS16 | Avg. |
|-----------------|-------|--------|-------|-------|-------|-------|-------|------|
| Non-BERT Baselines |      |        |       |       |       |       |       |      |
| GloVe           | Mean  | 58.02  | 53.76 | 55.14 | 70.66 | 59.73 | 68.25 | 63.66 | 61.32 |
| USE             |       | 74.92  | 76.69 | 64.49 | 67.80 | 64.61 | 76.83 | 73.18 | 71.22 |
| **BERT-base**   |       |        |       |       |       |       |       |      |
| + No tuning     | Mean  | 47.29  | 58.22 | 30.87 | 59.89 | 47.73 | 60.29 | 63.73 | 52.57 |
| + No tuning     | Mean  | 71.17  | 74.54 | 16.01 | 21.80 | 15.96 | 33.59 | 34.07 | 25.58 |
| + Flow + Mean 2 | 71.35  | 64.95  | 0.86  | 64.95 | 67.92 | 63.67 | 77.73 | 65.95 | 68.77 |
| + Contrastive (BT) | ± 63.27 | ± 66.91 | ± 1.29 | ± 54.26 | ± 1.84 | ± 64.03 | ± 1.35 | ± 54.28 | ± 1.87 | ± 68.19 |
| + Contrastive (SG) | ± 75.08 | ± 68.19 | ± 0.86 | ± 63.60 | ± 0.98 | ± 76.48 | ± 0.69 | ± 67.57 | ± 0.57 | ± 79.42 |
| + Contrastive (SG-OPT) | ± 77.23 | ± 68.16 | ± 0.50 | ± 66.82 | ± 0.73 | ± 80.13 | ± 0.80 | ± 71.23 | ± 0.40 | ± 81.56 |

**Table 1:** Experimental results on STS tasks. Results for trained models are averaged over 8 runs (±: the standard deviation). The best figure in each model-wise part is in **bold** and the best in each column is underlined. Our method with self-guidance (SG, SG-OPT) generally outperforms competitive baselines. We borrow scores from previous work if we could not reproduce them. †: from Reimers and Gurevych (2019). ‡: from Li et al. (2020).

Contrastive (SG-OPT), which utilize $L_{\text{base}}$ and $L_{\text{opt}}$ in Section 3 respectively.

**Results.** We report the performance of different approaches on STS tasks in Table 1 and Table 11 (Appendix A.6). From the results, we confirm the fact that our methods (SG and SG-OPT) mostly outperform other baselines in a variety of experimental settings. As reported in earlier studies, the naïve [CLS] embedding and mean pooling are turned out to be inferior to sophisticated methods.

To our surprise, WK pooling’s performance is even lower than that of mean pooling in most cases, and the only exception is when WK pooling is applied to SBERT-base. Flow shows its strength outperforming the simple strategies. Nevertheless, its performance is shown to be worse than that of our methods (although some exceptions exist in the case of SBERT-large). Note that contrastive learning becomes much more competitive when it is combined with our self-guidance algorithm rather than back-translation. It is also worth mentioning that the optimized version of our method (SG-OPT) generally shows better performance than the basic one (SG), proving the efficacy of learning objective optimization (Section 3.2). To conclude, we demonstrate that our self-guided contrastive learning is effective in improving the quality of BERT sentence embeddings when tested on STS tasks.

**Table 2:** SemEval-2014 Task 10 Spanish task.

| Models          | Spanish |
|-----------------|---------|
| Baseline (Agirre et al., 2014) | 80.69   |
| **UMC-DLSI-run2 (Rank #1)** | 80.57   |

| MBERT          |       |
|----------------|-------|
| + CLS          | 12.60 |
| + Mean pooling | 81.14 |
| + WK pooling   | 79.78 |
| + Contrastive (BT) | 78.04 |
| + Contrastive (SG) | 82.09 |
| + Contrastive (SG-OPT) | 82.74 |

**Table 2:** SemEval-2014 Task 10 Spanish task.
Table 3: Results on SemEval-2017 Task 1: Track 1 (Arabic), Track 3 (Spanish), and Track 5 (English).

Table 4: Experimental results on SentEval.
Models & STS Tasks (Avg.)

| Models | STS Tasks (Avg.) |
|--------|------------------|
| BERT-base + SG-OPT ($L_{opt}^1$) | 74.62 |
| + $L_{opt}^2$ | 73.14 (-1.48) |
| + $L_{opt}^3$ | 72.61 (-2.01) |
| + SG ($L_{base}$) | 72.17 (-2.45) |
| BERT-base + SG-OPT ($τ = 0.01, λ = 0.1$) | 74.62 |
| + $τ = 0.1$ | 70.39 (-4.23) |
| + $τ = 0.001$ | 74.16 (-0.46) |
| + $λ = 0.0$ | 73.76 (-0.86) |
| + $λ = 1.0$ | 73.18 (-1.44) |
| - Projection head ($f$) | 72.78 (-1.84) |

Table 5: Ablation study.

| Layer | Elapsed Time |
|-------|--------------|
|       | Training (sec.) | Inference (sec.) |
| BERT-base + Mean pooling | - | 13.94 |
| + WK pooling | - | 197.03 (≈ 3.3 min.) |
| + Flow | 155.37 (≈ 2.6 min.) | 28.49 |
| + Contrastive (SG-OPT) | 455.02 (≈ 7.5 min.) | 10.51 |

Table 6: Computational efficiency tested on STS-B.

5.1 Ablation Study

We conduct an ablation study to justify the decisions made in optimizing our algorithm. To this end, we evaluate each possible variant on the test sets of STS tasks. From Table 5, we confirm that all our modifications to the NT-Xent loss contribute to improvements in performance. Moreover, we show that correct choices for hyperparameters are important for achieving the optimal performance, and that the projection head ($f$) plays a significant role as in Chen et al. (2020).

5.2 Robustness to Domain Shifts

Although our method in principle can accept any sentences in training, its performance might be varied with the training data it employs (especially depending on whether the training and test data share the same domain). To explore this issue, we apply SG-OPT on BERT-base by leveraging the mix of NLI datasets (Bowman et al., 2015; Williams et al., 2018) instead of STS-B, and observe the difference. From Figure 4, we confirm the fact that no matter which test set is utilized (STS-B or all the seven STS tasks), our method clearly outperforms Flow in every case, showing its relative robustness to domain shifts. SG-OPT only loses 1.83 (on the STS-B test set) and 1.63 (on average when applied to all the STS tasks) points respectively when trained with NLI rather than STS-B, while Flow suffers from the considerable losses of 12.16 and 4.19 for each case. Note, however, that follow-up experiments in more diverse conditions might be desired as future work, as the NLI dataset inherently shares some similarities with STS tasks.

5.3 Computational Efficiency

In this part, we compare the computational efficiency of our method to that of other baselines. For each algorithm, we measure the time elapsed during training (if required) and inference when tested on STS-B. All methods are run on the same machine (an Intel Xeon CPU E5-2620 v4 @ 2.10GHz and a Titan Xp GPU) using batch size 16. The experimental results specified in Table 6 show that although our method demands a moderate amount of time (< 8 min.) for training, it is the most efficient at inference, since our method is free from any post-processing such as pooling once training is completed.

5.4 Representation Visualization

We visualize a few variants of BERT sentence representations to grasp an intuition on why our method is effective in improving performance. Specifically, we sample 20 positive pairs (red, whose similarity scores are 5) and 20 negative pairs (blue, whose scores are 0) from the STS-B validation set. Then we compute their vectors and draw them on the 2D space with the aid of t-SNE. In Figure 5, we confirm that our SG-OPT encourages BERT sentence embeddings to be more well-aligned with their positive pairs while still being relatively far from their negative pairs. We also visualize embeddings from SBERT (Figure 6 in Appendix A.5), and identify that our approach and the supervised fine-tuning...
Table 7: Ensemble of the techniques for contrastive learning: back-translation (BT) and self-guidance (SG-OPT).

| Models                  | Pooling | STS-B | SICK-R | STS12 | STS13 | STS14 | STS15 | STS16 | Avg. |
|-------------------------|---------|-------|--------|-------|-------|-------|-------|-------|------|
| BERT-base               | CLS     | 63.27 | 66.91  | 54.26 | 54.28 | 68.19 | 68.79 | 68.49 | 62.63 |
| + Contrastive (BT)      | CLS     | 77.23 | 68.16  | 66.84 | 80.13 | 71.23 | 81.36 | 77.17 | 74.62 |
| + Contrastive (SG-OPT)  | CLS     | 77.99 | 68.75  | 68.49 | 80.00 | 71.34 | 81.71 | 77.43 | 75.10 |

Figure 5: Sentence representation visualization. (Top) Embeddings from the original BERT. (Bottom) Embeddings from the BERT instance fine-tuned with SG-OPT. Red numbers correspond to positive sentence pairs and blue to negative pairs.

used in SBERT provide a similar effect, making the resulting embeddings more suitable for calculating correct similarities between them.

6 Discussion

In this section, we discuss a few weaknesses of our method in its current form and look into some possible avenues for future work.

First, while defining the proposed method in Section 3, we have made decisions on some parts without much consideration about their optimality, prioritizing simplicity instead. For instance, although we proposed utilizing all the intermediate layers of BERT and max pooling in a normal setting (indeed, it worked pretty well for most cases), a specific subset of the layers or another pooling method might bring better performance in a particular environment, as we observed in Section 4.4 that we could achieve higher numbers by employing mean pooling and excluding lower layers in the case of SentEval (refer to Appendix A.3 for details). Therefore, in future work, it is encouraged to develop a systematic way of making more optimized design choices in specifying our method by considering the characteristics of target tasks.

Second, we expect that the effectiveness of contrastive learning in revising BERT can be improved further by properly combining different techniques developed for it. As an initial attempt towards this direction, we conduct an extra experiment where we test the ensemble of back-translation and our self-guidance algorithm by inserting the original sentence into BERT\(F\) and its back-translation into BERT\(F\) when running our framework. In Table 7, we show that the fusion of the two techniques generally results in better performance, shedding some light on our future research direction.

7 Conclusion

In this paper, we have proposed a contrastive learning method with self-guidance for improving BERT sentence embeddings. Through extensive experiments, we have demonstrated that our method can enjoy the benefit of contrastive learning without relying on external procedures such as data augmentation or back-translation, succeeding in generating higher-quality sentence representations compared to competitive baselines. Furthermore, our method is efficient at inference because it does not require any post-processing once its training is completed, and is relatively robust to domain shifts.

Acknowledgments

We would like to thank anonymous reviewers for their fruitful feedback. We are also grateful to Jung-Woo Ha, Sang-Woo Lee, Gyuwan Kim, and other members in NAVER AI Lab in addition to Reinald Kim Amplayo for their insightful comments.
References

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. 2020. Better fine-tuning by reducing representational collapse. *arXiv preprint arXiv:2008.03156*.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *SemEval*.

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

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity*. In *SEM*.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Fredrik Carlsson, Evangelia Gogoulou, Erik Ylipää, Amaru Cuba Gyllensten, and Magnus Sahlgren. 2021. Semantic re-tuning with contrastive tension. In *ICLR*.

Daniel Cer, Mona Diab, Eneko Agirre, and Weiwei Guo. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In *SemEval*.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In *SemEval*.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *ICLR*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *CVPR*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *ICML*.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. 2020. A survey on contrastive self-supervised learning. *arXiv preprint arXiv:2011.00362*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *ACL*.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *ACL*.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *EMNLP*.
A Appendices

A.1 Hyperparameters

| Hyperparameters       | Values                  |
|-----------------------|-------------------------|
| Random seed           | 1, 2, 3, 4, 1234, 2345, 3456, 7890 |
| Evaluation step       | 50                      |
| Epoch                 | 1                       |
| Batch size (b)        | 16                      |
| Optimizer             | AdamW ($\beta_1$, $\beta_2$=(0.9, 0.9)) |
| Learning rate         | 0.00005                 |
| Early stopping endurance | 10                      |
| $\tau$               | 0.01                    |
| $\lambda$            | 0.1                     |

Table 8: Hyperparameters for experiments.

A.2 Specification on Contrastive (BT)

This baseline is identical with our Contrastive (SG) model, except that it utilizes back-translation to generate positive samples. To be specific, English sentences in the training set are translated into German sentences using the WMT’19 English-German translator provided by Ng et al. (2019), and then the translated German sentences are back-translated into English with the aid of the WMT’19 German-English model also offered by Ng et al. (2019). We utilize beam search during decoding with the beam size 100, which is relatively large, since we want generated sentences to be more diverse while grammatically correct at the same time. Note that the contrastive (BT) model is trained with the NT-Xent loss (Chen et al., 2020), unlike CERT (Fang and Xie, 2020) which leverages the MoCo training objective (He et al., 2020).

A.3 SentEval Configurations

| Hyperparameters       | Values                  |
|-----------------------|-------------------------|
| Random seed           | 1, 2, 3, 4, 1234, 2345, 3456, 7890 |
| K-fold                | 10                      |
| Classifier (hidden dimension) | 50                     |
| Optimizer             | Adam                    |
| Batch size            | 64                      |
| Tenacity              | 5                       |
| Epoch                 | 4                       |

Table 9: SentEval hyperparameters.

In Table 9, we stipulate the hyperparameters of the SentEval toolkit used in our experiment. Additionally, we specify some minor modifications applied on our contrastive method (SG-OPT). First, we use the portion of the concatenation of SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018) datasets as the training data instead of STS-B. Second, we do not leverage the first several layers of PLMs when making positive samples, similar to Wang and Kuo (2020), and utilize mean pooling instead of max pooling.

A.4 GLUE Experiments

| Models     | QNLI | SST2 | COLA | MRPC | RTE  |
|------------|------|------|------|------|------|
| BERT-base  | 90.97| 91.08| 56.63| 87.09| 62.50|
| + SG-OPT   | 91.28| 91.68| 56.36| 86.96| 62.75|

Table 10: Experimental results on a portion of the GLUE validation set.

We here investigate the impact of our method on typical supervised fine-tuning of BERT models. Concretely, we compare the original BERT with one fine-tuned using our SG-OPT method on the GLUE (Wang et al., 2019) benchmark. Note that we use the first 10% of the GLUE validation set as the real validation set and the last 90% as the test set, as the benchmark does not officially provide its test data. We report experimental results tested on 5 sub-tasks in Table 10. The results show that our method brings performance improvements for 3 tasks (QNLI, SST2, and RTE). However, it seems that SG-OPT does not influence much on supervised fine-tuning results, considering that the absolute performance gap between the two models is not significant. We leave more analysis on this part as future work.

A.5 Representation Visualization (SBERT)

![Figure 6: Visualization of sentence vectors computed by SBERT-base.]

In Table 9, we stipulate the hyperparameters of the SentEval toolkit used in our experiment. Additionally, we specify some minor modifications applied on our contrastive method (SG-OPT). First, we use the portion of the concatenation of SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018) datasets as the training data instead of STS-B. Second, we do not leverage the first several layers of PLMs when making positive samples, similar to Wang and Kuo (2020), and utilize mean pooling instead of max pooling.

A.6 RoBERTa’s Performance on STS Tasks

In Table 11, we additionally report the performance of sentence embeddings extracted from RoBERTa using different methods. Our methods, SG and SG-OPT, demonstrate their competitive performance.
| Models          | Pooling | SICK-R | STS12 | STS13 | STS14 | STS15 | STS16 | Avg  |
|-----------------|---------|--------|-------|-------|-------|-------|-------|------|
| RoBERTa-base    | + No tuning | CLS 45.41 | 61.89 | 16.67 | 45.57 | 30.36 | 55.08 | 56.98 | 44.57 |
|                 | + No tuning | Mean 54.53 | 62.03 | 32.11 | 56.33 | 45.22 | 61.34 | 61.98 | 53.36 |
|                 | + No tuning | WK 35.75 | 54.69 | 20.31 | 36.51 | 32.41 | 48.12 | 46.32 | 39.16 |
|                 | + Contrastive (BT) | CLS 77.93±1.08 | 71.97±1.00 | 62.34±2.41 | 78.60±1.74 | 58.65±1.48 | 79.31±0.65 | 77.49±1.29 | 74.04±1.16 |
|                 | + Contrastive (SG) | CLS 78.38±0.43 | 69.74±1.00 | 62.85±2.88 | 78.37±1.55 | 68.28±0.89 | 80.42±0.65 | 77.69±0.76 | 73.67±0.62 |
|                 | + Contrastive (SG-OPT) | CLS 77.60±0.30 | 68.42±0.71 | 62.57±1.12 | 78.96±0.67 | 69.24±0.44 | 79.99±0.44 | 77.17±0.24 | 73.42±0.31 |

| RoBERTa-large   | + No tuning | CLS 12.52 | 40.63 | 19.25 | 22.97 | 14.93 | 33.41 | 38.01 | 25.96 |
|                 | + No tuning | Mean 47.07 | 58.38 | 33.63 | 57.22 | 45.67 | 63.00 | 61.18 | 52.31 |
|                 | + No tuning | WK 30.29 | 28.25 | 23.17 | 30.92 | 23.36 | 40.07 | 43.32 | 31.34 |
|                 | + Contrastive (BT) | CLS 77.05±1.22 | 67.83±1.34 | 57.60±3.57 | 72.14±1.16 | 62.25±2.10 | 71.49±3.24 | 71.75±1.73 | 68.59±1.53 |
|                 | + Contrastive (SG) | CLS 76.15±0.54 | 66.07±0.82 | 64.77±2.52 | 71.96±1.53 | 64.54±1.04 | 78.06±0.52 | 75.14±0.94 | 70.95±1.13 |
|                 | + Contrastive (SG-OPT) | CLS 78.14±0.72 | 67.97±1.09 | 64.29±1.54 | 76.36±1.47 | 68.48±1.58 | 80.10±1.03 | 76.60±0.98 | 73.13±1.20 |

Table 11: Performance of RoBERTa on STS tasks when combined with different sentence embedding methods. We could not report the performance of Li et al. (2020) (Flow) as their official code do not support RoBERTa.

overall. Note that contrastive learning with back-translation (BT) also shows its remarkable performance in the case of RoBERTa-base.