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. 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
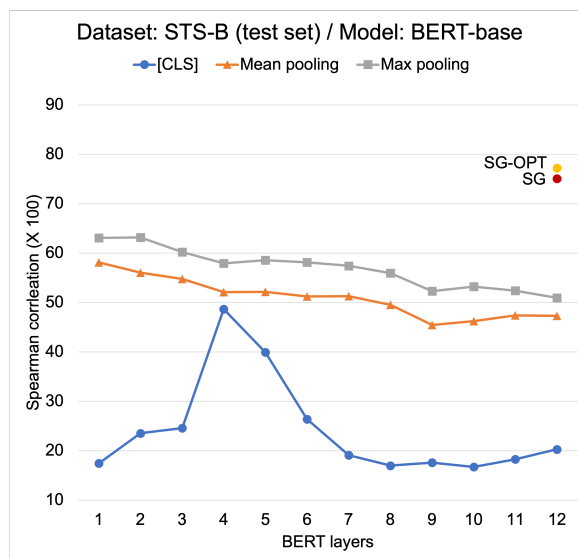

Figure 1: BERT(-base)'s layer-wise performance with different pooling methods on the STS-B test set. We observe that the performance can be dramatically varied according to the selected layer and pooling strategy. Our self-guided training (SG / SG-OPT) assures much improved results compared to those of the baselines.

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.[1] In practice, previous studies (Reimers and Gurevych, 2019; Li et al., 2020; Hu et al., 2020) reported that naïvely (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.

---

[*]This work has been mainly conducted when TK was a research intern at NAVER AI Lab.

[1]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 ($\times$ 100), can range from as low as 16.71 (`[CLS]`, the 10th layer) to 63.19 (max pooling, the 2nd 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

---

[2]In the experiment, we employ the settings identical with ones used in Chapter 4. Refer to Chapter 4 for more details.

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 [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
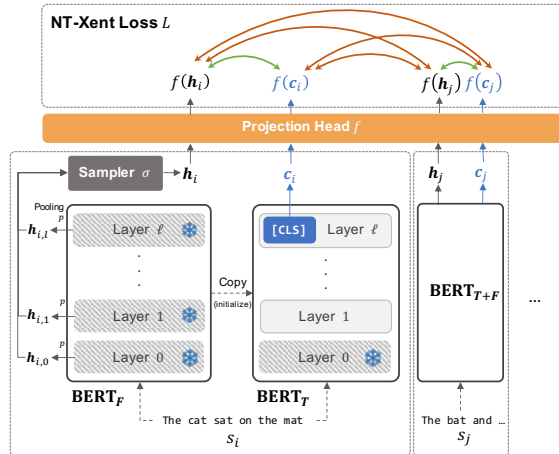

Figure 2: Self-guided contrastive learning framework. We clone BERT into two copies at the beginning of training. $BERT_T$ (except Layer 0) is then fine-tuned to optimize the sentence vector c_i while $BERT_F$ is fixed.

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 [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 $H_{i,k} \in \mathbb{R}^{len(s_i) \times d}$:

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

where $0 \leq k \leq l$ (0: the non-contextualized layer), l is the number of hidden layers in BERT, $len(s_i)$ 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 $\mathbf{h}_{i,k} \in \mathbb{R}^d$ from all layers, i.e., $\mathbf{h}_{i,k} = p(H_{i,k})$. Finally, we choose the final view to be utilized by applying a sampling function σ:

$$\mathbf{h}_i = \sigma(\{\mathbf{h}_{i,k} | 0 \leq k \leq l\}).$$

As we have no specific constraints in defining p and σ, we employ max pooling as p and a uniform sampler as σ for simplicity, unless otherwise stated. This simple choice for the sampler implies that each $\mathbf{h}_{i,k}$ 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 $\mathbf{c}_i$ for s_i as follows:

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

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

$$L_m^{base} = -\log\left(\phi(\mathbf{x}_m, \mu(\mathbf{x}_m))/Z\right),$$
$$\text{where } \phi(\mathbf{u}, \mathbf{v}) = \exp(g(f(\mathbf{u}), f(\mathbf{v}))/\tau)$$
$$\text{and } Z = \sum_{n=1, n \neq m}^{2b} \phi(\mathbf{x}_m, \mathbf{x}_n).$$

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

$$\mu(\mathbf{x}) = \begin{cases} \mathbf{h}_i & \text{if } \mathbf{x} \text{ is equal to } \mathbf{c}_i. \\ \mathbf{c}_i & \text{if } \mathbf{x} \text{ is equal to } \mathbf{h}_i. \end{cases}$$

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

---

[3] We can also potentially make use of another sampler functions to inject our bias or prior knowledge on target tasks.

[4] We employ a two-layered MLP whose hidden size is 4096. Each linear layer in the MLP is followed by a GELU function.

[5] To be specific, L^{reg} is the square of the L2 norm of the difference between BERT_F and BERT_T. As shown in Figure 2, we also freeze the 0th layer of BERT_T for stable learning.
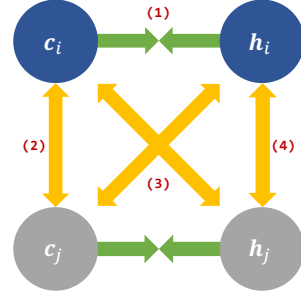

Figure 3: Four factors of the original NT-Xent loss. Green and yellow arrows represent the force of attraction and repulsion, respectively. Best viewed in color.

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

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

where the coefficient λ is a hyperparameter.

To summarize, our method refines BERT so that the sentence embedding $\mathbf{c}_i$ has a higher similarity with $\mathbf{h}_i$, which is another representation for the sentence s_i, in the subspace projected by f while being relatively dissimilar with $\mathbf{c}_{j,j\neq i}$ and $\mathbf{h}_{j,j\neq i}$. After training is completed, we remove all the components except BERT_T and simply use $\mathbf{c}_i$ 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) $\mathbf{c}_i \rightarrow\leftarrow \mathbf{h}_i$ (or $\mathbf{c}_j \rightarrow\leftarrow \mathbf{h}_j$): The main component that mirrors our core motivation that a BERT sentence vector ($\mathbf{c}_i$) should be consistent with intermediate views ($\mathbf{h}_i$) from BERT.

(2) $\mathbf{c}_i \longleftrightarrow \mathbf{c}_j$: A factor that forces sentence embeddings ($\mathbf{c}_i$, $\mathbf{c}_j$) to be distant from each other.

(3) $\mathbf{c}_i \longleftrightarrow \mathbf{h}_j$ (or $\mathbf{c}_j \longleftrightarrow \mathbf{h}_i$): An element that makes $\mathbf{c}_i$ being inconsistent with views for other sentences ($\mathbf{h}_j$).

(4) $\mathbf{h}_i \longleftrightarrow \mathbf{h}_j$: A factor that causes a discrepancy between views of different sentences ($\mathbf{h}_i$, $\mathbf{h}_j$).

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

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:

$$L_i^{opt1} = -\log\left(\phi(\mathbf{c}_i, \mathbf{h}_i)/\hat{Z}\right),$$

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

$$L_{i,k}^{opt3} = -\log \frac{\phi(\mathbf{c}_i, \mathbf{h}_{i,k})}{\phi(\mathbf{c}_i, \mathbf{h}_{i,k}) + \sum_{m=1, m\neq i}^{b} \sum_{n=0}^{l} \phi(\mathbf{c}_i, \mathbf{h}_{m,n})}.$$

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:

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

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.

Models	Pooling	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	<u>76.69</u>	64.49	67.80	64.61	76.83	73.18	71.22
BERT-base									
+ No tuning	CLS	20.30	42.42	21.54	32.11	21.28	37.89	44.24	31.40
+ No tuning	Mean	47.29	58.22	30.87	59.89	47.73	60.29	63.73	52.57
+ No tuning	WK	16.07	41.54	16.01	21.80	15.96	33.59	34.07	25.58
+ Flow	Mean-2	$71.35_{\pm0.27}$	$64.95_{\pm0.16}$	$64.32_{\pm0.17}$	$69.72_{\pm0.25}$	$63.67_{\pm0.06}$	$77.77_{\pm0.15}$	$69.59_{\pm0.28}$	$68.77_{\pm0.07}$
+ Contrastive (BT)	CLS	$63.27_{\pm1.48}$	$66.91_{\pm1.29}$	$54.26_{\pm1.84}$	$64.03_{\pm2.35}$	$54.28_{\pm1.87}$	$68.19_{\pm0.95}$	$67.50_{\pm0.96}$	$62.63_{\pm1.28}$
+ **Contrastive (SG)**	CLS	$75.08_{\pm0.73}$	$\mathbf{68.19}_{\pm0.36}$	$63.60_{\pm0.98}$	$76.48_{\pm0.69}$	$67.57_{\pm0.57}$	$79.42_{\pm0.49}$	$74.85_{\pm0.54}$	$72.17_{\pm0.44}$
+ **Contrastive (SG-OPT)**	CLS	$\mathbf{77.23}_{\pm0.43}$	$68.16_{\pm0.50}$	$\mathbf{66.84}_{\pm0.73}$	$\mathbf{80.13}_{\pm0.51}$	$\mathbf{71.23}_{\pm0.40}$	$\mathbf{81.56}_{\pm0.28}$	$\mathbf{77.17}_{\pm0.22}$	$\mathbf{74.62}_{\pm0.25}$
BERT-large									
+ No tuning	CLS	26.75	43.44	27.44	30.76	22.59	29.98	42.74	31.96
+ No tuning	Mean	47.00	53.85	27.67	55.79	44.49	51.67	61.88	48.91
+ No tuning	WK	35.75	38.39	12.65	26.41	23.74	29.34	34.42	28.67
+ Flow	Mean-2	$72.72_{\pm0.36}$	$63.77_{\pm0.18}$	$62.82_{\pm0.17}$	$71.24_{\pm0.22}$	$65.39_{\pm0.15}$	$78.98_{\pm0.21}$	$73.23_{\pm0.24}$	$70.07_{\pm0.81}$
+ Contrastive (BT)	CLS	$63.84_{\pm1.05}$	$66.53_{\pm2.62}$	$52.04_{\pm1.75}$	$62.59_{\pm1.84}$	$54.25_{\pm1.45}$	$71.07_{\pm1.11}$	$66.71_{\pm1.08}$	$62.43_{\pm1.07}$
+ **Contrastive (SG)**	CLS	$75.22_{\pm0.57}$	$69.63_{\pm0.95}$	$64.37_{\pm0.72}$	$77.59_{\pm1.01}$	$68.27_{\pm0.40}$	$80.08_{\pm0.28}$	$74.53_{\pm0.43}$	$72.81_{\pm0.31}$
+ **Contrastive (SG-OPT)**	CLS	$\mathbf{76.16}_{\pm0.42}$	$\mathbf{70.20}_{\pm0.65}$	$\mathbf{67.02}_{\pm0.72}$	$\mathbf{79.42}_{\pm0.80}$	$\mathbf{70.38}_{\pm0.65}$	$\mathbf{81.72}_{\pm0.32}$	$\mathbf{76.35}_{\pm0.22}$	$\mathbf{74.46}_{\pm0.35}$
SBERT-base									
+ No tuning	CLS	73.66	69.71	70.15	71.17	68.89	75.53	70.16	71.32
+ No tuning	Mean	76.98	72.91	70.97	76.53	73.19	79.09	74.30	74.85
+ No tuning	WK	78.38	74.31	69.75	76.92	72.32	81.17	76.25	75.59
+ Flow[‡]	Mean-2	81.03	74.97	68.95	78.48	**77.62**	81.95	78.94	77.42
+ Contrastive (BT)	CLS	$74.67_{\pm0.30}$	$70.31_{\pm0.45}$	$71.19_{\pm0.37}$	$72.41_{\pm0.60}$	$69.90_{\pm0.43}$	$77.16_{\pm0.48}$	$71.63_{\pm0.55}$	$72.47_{\pm0.37}$
+ **Contrastive (SG)**	CLS	$81.05_{\pm0.34}$	$75.78_{\pm0.55}$	$73.76_{\pm0.76}$	$80.08_{\pm0.45}$	$75.58_{\pm0.57}$	$83.52_{\pm0.43}$	$79.10_{\pm0.51}$	$78.41_{\pm0.33}$
+ **Contrastive (SG-OPT)**	CLS	$\mathbf{81.46}_{\pm0.27}$	$\mathbf{76.64}_{\pm0.42}$	$\underline{\mathbf{75.16}}_{\pm0.56}$	$\mathbf{81.27}_{\pm0.37}$	$76.31_{\pm0.38}$	$\underline{\mathbf{84.71}}_{\pm0.26}$	$\mathbf{80.33}_{\pm0.19}$	$\mathbf{79.41}_{\pm0.17}$
SBERT-large									
+ No tuning	CLS	76.01	70.99	69.05	71.34	69.50	76.66	70.08	71.95
+ No tuning	Mean	79.19	73.75	72.27	78.46	74.90	80.99	76.25	76.54
+ No tuning	WK	61.87	67.06	49.95	53.02	46.55	62.47	60.32	57.32
+ Flow[‡]	Mean-2	81.18	74.52	70.19	80.27	<u>78.85</u>	82.97	<u>80.57</u>	78.36
+ Contrastive (BT)	CLS	$76.71_{\pm1.22}$	$71.56_{\pm1.34}$	$69.95_{\pm3.57}$	$72.66_{\pm1.16}$	$70.38_{\pm2.10}$	$77.80_{\pm3.24}$	$71.41_{\pm1.73}$	$72.92_{\pm1.53}$
+ **Contrastive (SG)**	CLS	$\underline{\mathbf{82.35}}_{\pm0.15}$	$\mathbf{76.44}_{\pm0.41}$	$\mathbf{74.84}_{\pm0.57}$	$82.89_{\pm0.41}$	$77.27_{\pm0.35}$	$84.44_{\pm0.23}$	$79.54_{\pm0.49}$	$79.68_{\pm0.37}$
+ **Contrastive (SG-OPT)**	CLS	$82.05_{\pm0.39}$	$\mathbf{76.44}_{\pm0.29}$	$74.58_{\pm0.59}$	$\mathbf{83.79}_{\pm0.14}$	$76.98_{\pm0.19}$	$84.57_{\pm0.27}$	$79.87_{\pm0.42}$	$\underline{\mathbf{79.76}}_{\pm0.33}$

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 <u>underlined</u>. 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^{base} and L^{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

Models	Spanish
Baseline (Agirre et al., 2014)	
UMCC-DLSI-run2 (Rank #1)	80.69
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.

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	Arabic (Track 1)	Spanish (Track 3)	English (Track 5)
Baselines			
Cosine baseline (Cer et al., 2017)	60.45	71.17	72.78
ENCU (Rank #1, Tian et al. (2017))	74.40	85.59	85.18
MBERT			
+ CLS	30.57	29.38	24.97
+ Mean pooling	51.09	54.56	54.86
+ WK pooling	50.38	55.87	54.87
+ Contrastive (BT)	54.24	68.16	73.89
+ Contrastive (SG)	57.09	78.93	78.24
+ Contrastive (SG-OPT)	58.52	80.19	78.03

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

Models	MR	CR	SUBJ	MPQA	SST2	TREC	MRPC	Avg.
BERT-base								
+ Mean	81.46	86.71	95.37	87.90	85.83	90.30	73.36	85.85
+ WK	80.64	85.53	95.27	88.63	85.03	**94.03**	71.71	85.83
+ SG-OPT	**82.47**	**87.42**	**95.40**	**88.92**	**86.20**	91.60	**74.21**	**86.60**
BERT-large								
+ Mean	84.38	89.01	95.60	86.69	89.20	90.90	72.79	86.94
+ WK	82.68	87.92	95.32	**87.25**	87.81	91.18	70.13	86.04
+ SG-OPT	**86.03**	**90.18**	**95.82**	87.08	**90.73**	**94.65**	**73.31**	**88.26**
SBERT-base								
+ Mean	82.80	89.03	94.07	89.79	88.08	86.93	75.11	86.54
+ WK	82.96	89.33	**95.13**	**90.56**	88.10	**91.98**	**76.66**	**87.82**
+ SG-OPT	**83.34**	**89.45**	94.68	89.78	**88.57**	87.30	75.26	86.91

Table 4: Experimental results on SentEval.

English data and test it on datasets written in other languages. As in Section 4.2, we use the English STS-B for training. We consider two datasets for evaluation: (1) SemEval-2014 Task 10 (Spanish; Agirre et al. (2014)) and (2) SemEval-2017 Task 1 (Arabic, Spanish, and English; Cer et al. (2017)). Performance is measured in Pearson correlation ($\times$ 100) for a fair comparison with previous work.

From Table 2, we see that MBERT with mean pooling already outperforms the best system (at the time of the competition was held) on SemEval-2014 and that our method further boosts the model's performance. In contrast, in the case of SemEval-2017 (Table 3), MBERT with mean pooling even fails to beat the strong Cosine baseline.[8] However, MBERT becomes capable of outperforming (in English/Spanish) or being comparable with (Arabic) the baseline by adopting our algorithm. We observe that while cross-lingual transfer using MBERT looks promising for the languages analogous to English (e.g., Spanish), its effectiveness may shrink on distant languages (e.g., Arabic). Compared against the best system which is trained on task-specific data, MBERT shows reasonable performance considering that it is never exposed to any labeled STS datasets. In summary, we demonstrate that MBERT fine-tuned with our method has a potential to be used as a simple but effective tool for multilingual (especially European) STS tasks.

4.4 SentEval and Supervised Fine-tuning

We also evaluate BERT sentence vectors using the SentEval (Conneau and Kiela, 2018) toolkit. Given sentence embeddings, SentEval trains linear classifiers on top of them and estimates the quality of the vectors via their performance (accuracy) on down-stream tasks. Among available tasks, we employ 7: MR, CR, SUBJ, MPQA, SST2, TREC, MRPC.[9]

In Table 4, we compare our method (SG-OPT) with two baselines.[10] We find that our method is helpful over usual mean pooling in improving the performance of BERT-like models on SentEval. SG-OPT also outperforms WK pooling on BERT-base/large while being comparable on SBERT-base. From the results, we conjecture that self-guided contrastive learning and SBERT training suggest a similar inductive bias in a sense, as the benefit we earn by revising SBERT with our method is relatively lower than the gain we obtain when fine-tuning BERT. Meanwhile, it seems that WK pooling provides an orthogonal contribution that is effective in the focused case, i.e., SBERT-base.

In addition, we examine how our algorithm impacts on supervised fine-tuning of BERT, although it is not the main concern of this work. Briefly reporting, we identify that the original BERT(-base) and one tuned with SG-OPT show comparable performance on the GLUE (Wang et al., 2019) validation set, implying that our method does not influence much on BERT's supervised fine-tuning. We refer readers to Appendix A.4 for more details.

5 Analysis

We here further investigate the working mechanism of our method with supplementary experiments. All the experiments conducted in this section follow the configurations stipulated in Section 4.1 and 4.2.

---

[8]The Cosine baseline computes its score as the cosine similarity of binary sentence vectors with each dimension representing whether an individual word appears in a sentence.

[9]Refer to Conneau and Kiela (2018) for each task's spec.

[10]We focus on reporting our own results as we discovered that the toolkit's outcomes can be fluctuating depending on its configuration (we list our settings in Appendix A.3). We also restrict ourselves to evaluating SG-OPT for simplicity, as SG-OPT consistently showed better performance than other contrastive methods in previous experiments.

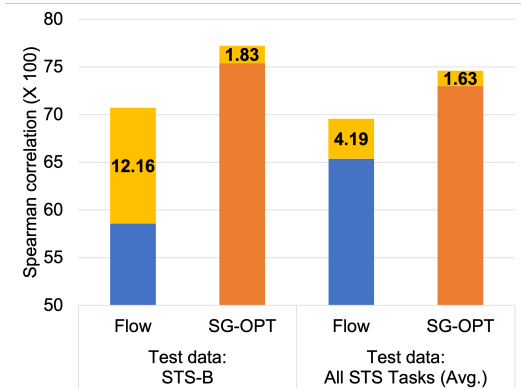| Models	STS Tasks (Avg.)
BERT-base	
+ **SG-OPT** (L^{opt3})	74.62
+ L^{opt2}	73.14 (-1.48)
+ L^{opt1}	72.61 (-2.01)
+ **SG** (L^{base})	72.17 (-2.45)
BERT-base + **SG-OPT** ($\tau = 0.01, \lambda = 0.1$)	74.62
+ $\tau = 0.1$	70.39 (-4.23)
+ $\tau = 0.001$	74.16 (-0.46)
+ $\lambda = 0.0$	73.76 (-0.86)
+ $\lambda = 1.0$	73.18 (-1.44)
- Projection head (f)	72.78 (-1.84)

Table 5: Ablation study.

Figure 4: Domain robustness study. The yellow bars indicate the performance gaps each method has according to which data it is trained with: in-domain (STS-B) or out-of-domain (NLI). Our method (SG-OPT) clearly shows its relative robustness compared to Flow.

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

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

Table 6: Computational efficiency tested on STS-B.

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

Models	Pooling	STS-B	SICK-R	STS12	STS13	STS14	STS15	STS16	Avg.
BERT-base									
+ Contrastive (BT)	CLS	$63.27_{\pm 1.48}$	$66.91_{\pm 1.29}$	$54.26_{\pm 1.84}$	$64.03_{\pm 2.35}$	$54.28_{\pm 1.87}$	$68.19_{\pm 0.95}$	$67.50_{\pm 0.96}$	$62.63_{\pm 1.28}$
+ Contrastive (SG-OPT)	CLS	$77.23_{\pm 0.43}$	$68.16_{\pm 0.50}$	$66.84_{\pm 0.73}$	$\mathbf{80.13}_{\pm 0.51}$	$71.23_{\pm 0.40}$	$81.56_{\pm 0.28}$	$77.17_{\pm 0.22}$	$74.62_{\pm 0.25}$
+ Contrastive (BT + SG-OPT)	CLS	$\mathbf{77.99}_{\pm 0.23}$	$\mathbf{68.75}_{\pm 0.79}$	$\mathbf{68.49}_{\pm 0.38}$	$80.00_{\pm 0.78}$	$\mathbf{71.34}_{\pm 0.40}$	$\mathbf{81.71}_{\pm 0.29}$	$\mathbf{77.43}_{\pm 0.46}$	$\mathbf{75.10}_{\pm 0.15}$

Table 7: Ensemble of the techniques for contrastive learning: back-translation (BT) and self-guidance (SG-OPT).
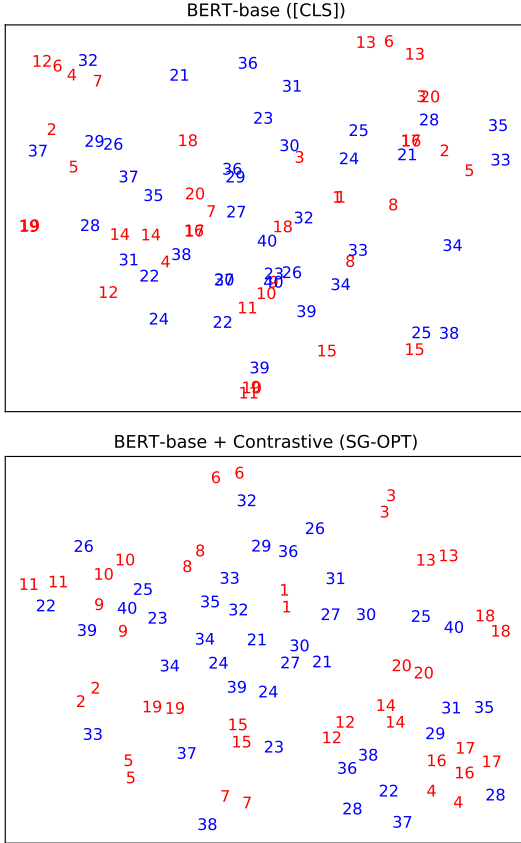

Figure 5: Sentence representation visualization. (Top) Embeddings from the original BERT. (Bottom) Embeddings from the BERT instance fine-tuned with SG-OPT. Red numbers correspond to positive sentence pairs and blue to negative pairs.

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 set-ting (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 $BERT_T$ 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.

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, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. 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*.

Eneko Agirre, 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, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *SemEval*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *ICML*.

Xinlei Chen and Kaiming He. 2020. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *LREC*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Hongchao Fang and Pengtao Xie. 2020. Cert: Contrastive self-supervised learning for language understanding. *arXiv preprint arXiv:2005.12766*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

John M Giorgi, Osvald Nitski, Gary D Bader, and Bo Wang. 2020. Declutr: Deep contrastive learning for unsupervised textual representations. *arXiv preprint arXiv:2006.03659*.

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 multitask 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 pretrained natural language models through principled regularized optimization. In *ACL*.

Phuc H Le-Khac, Graham Healy, and Alan F Smeaton. 2020. Contrastive representation learning: A framework and review. *IEEE Access*.

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*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. In *ICLR*.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *LREC*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *NeurIPS*.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *arXiv preprint arXiv:2006.04884*.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook fair's wmt19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *EMNLP-IJCNLP*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *ACL*.

Junfeng Tian, Zhiheng Zhou, Man Lan, and Yuanbin Wu. 2017. ECNU at SemEval-2017 task 1: Leverage kernel-based traditional NLP features and neural networks to build a universal model for multilingual and cross-lingual semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Bin Wang and C-C Jay Kuo. 2020. Sbert-wk: A sentence embedding method by dissecting bert-based word models. *arXiv preprint arXiv:2002.06652*.

Kexin Wang, Nils Reimers, and Iryna Gurevych. 2021. Tsdae: Using transformer-based sequential denoising auto-encoder for unsupervised sentence embedding learning. *arXiv preprint arXiv:2104.06979*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. Clear: Contrastive learning for sentence representation. *arXiv preprint arXiv:2012.15466*.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. In *NeurIPS*.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *ICLR*.

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 (β_1, β_2=(0.9, 0.9))
Learning rate	0.00005
Early stopping endurance	10
τ	0.01
λ	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 traslated 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$_{\pm 0.49}$	91.08$_{\pm 0.73}$	**56.63**$_{\pm 3.82}$	**87.09**$_{\pm 1.87}$	62.50$_{\pm 2.77}$
+ SG-OPT	**91.28**$_{\pm 0.28}$	**91.68**$_{\pm 0.41}$	56.36$_{\pm 3.98}$	86.96$_{\pm 1.11}$	**62.75**$_{\pm 3.91}$

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.
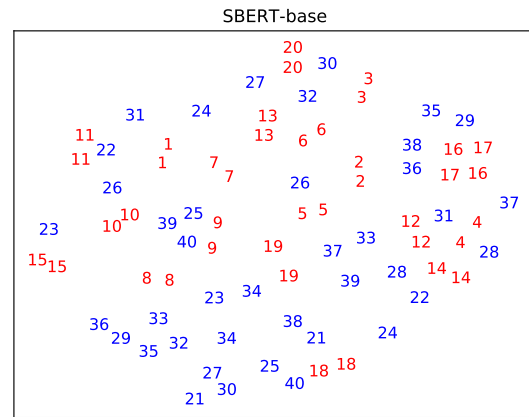
A.5 Representation Visualization (SBERT)

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

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

2539

Models	Pooling	STS-B	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	$\mathbf{79.93}_{\pm 1.08}$	$\mathbf{71.97}_{\pm 1.00}$	$62.34_{\pm 2.41}$	$78.60_{\pm 1.74}$	$68.65_{\pm 1.48}$	$79.31_{\pm 0.65}$	$77.49_{\pm 1.29}$	$\mathbf{74.04}_{\pm 1.16}$
+ Contrastive (SG)	CLS	$78.38_{\pm 0.43}$	$69.74_{\pm 1.00}$	$\mathbf{62.85}_{\pm 0.88}$	$78.37_{\pm 1.55}$	$68.28_{\pm 0.89}$	$\mathbf{80.42}_{\pm 0.65}$	$\mathbf{77.69}_{\pm 0.76}$	$73.67_{\pm 0.62}$
+ Contrastive (SG-OPT)	CLS	$77.60_{\pm 0.30}$	$68.42_{\pm 0.71}$	$62.57_{\pm 1.12}$	$\mathbf{78.96}_{\pm 0.67}$	$\mathbf{69.24}_{\pm 0.44}$	$79.99_{\pm 0.44}$	$77.17_{\pm 0.24}$	$73.42_{\pm 0.31}$
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	$77.05_{\pm 1.22}$	$67.83_{\pm 1.34}$	$57.60_{\pm 3.57}$	$72.14_{\pm 1.16}$	$62.25_{\pm 2.10}$	$71.49_{\pm 3.24}$	$71.75_{\pm 1.73}$	$68.59_{\pm 1.53}$
+ Contrastive (SG)	CLS	$76.15_{\pm 0.54}$	$66.07_{\pm 0.82}$	$\mathbf{64.77}_{\pm 2.52}$	$71.96_{\pm 1.53}$	$64.54_{\pm 1.04}$	$78.06_{\pm 0.52}$	$75.14_{\pm 0.94}$	$70.95_{\pm 1.13}$
+ Contrastive (SG-OPT)	CLS	$\mathbf{78.14}_{\pm 0.72}$	$67.97_{\pm 1.09}$	$64.29_{\pm 1.54}$	$\mathbf{76.36}_{\pm 1.47}$	$\mathbf{68.48}_{\pm 1.58}$	$\mathbf{80.10}_{\pm 1.05}$	$\mathbf{76.60}_{\pm 0.98}$	$\mathbf{73.13}_{\pm 1.20}$

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.