How to Fine-Tune BERT for Multiple Tasks?

Stanford CS224N Default Project

Jingru Cheng
Department of Energy Science
Engineering
Stanford University
cjr63@stanford.edu

Bohao He
Department of Statistics
Stanford University
hebohao@stanford.edu

Abstract

BERT (Devlin et al., 2018) is proved to be successful on a variety of NLP tasks, and its fine-tuning strategies has been extensively discussed in literature. However, only a few of these discussions focused on multi-task fine-tuning. In this project, we investigate effects of additional pre-training, the Sentence-BERT structure with different loss functions, and multi-task training routines on multi-task fine-tuning. A robust BERT embedding performing well on the three default tasks simultaneously is obtained using a parallel fine-tuning routine.

1 Introduction

It has been proved that large language model pre-training is an effective approach to solve many classic problems in the field of Natural Language Processing (NLP) such as sentiment analysis, question answering, name entity recognition, etc. A common practice is the two-step regime of unsupervised pre-training on large corpus, aiming to obtain high quality embedding of the text, and supervised fine-tuning on labeled task-specific data, which allows the model to adapt to downstream tasks.

Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) is one of the most influential language models ever proposed due to the conciseness of its idea and the impressive performance it obtained on a variety of natural language understanding (NLU) tasks. Although the fine-tuning of BERT has been closely studied (Liu et al., 2019b; Sun et al., 2019), most works focus on single-task fine-tuning, i.e. fine-tuning the model separately for different tasks, and the idea of multi-task fine-tuning is yet to be fully explored. In most cases, multi-task fine-tuning is not considered necessary as it cannot enhance the performance of the model on single tasks (Sun et al., 2019). However, it can potentially deliver a robust embedding which incorporates multiple aspects of the text and performs well simultaneously on multiple tasks, which is desirable in terms of NLU and beneficial to domain adaptation when the available in-domain labels are substantially fewer (Liu et al., 2019a).

In this project, we investigate both single-task fine-tuning techniques and multi-task fine-tuning routines with the goal of obtaining a BERT embedding performing well on three NLP tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. In summary, our work consists of three major parts:

- We implement the Masked Language Model (MLM) pre-training and investigate its effect on BERT fine-tuning in both single-task and multi-task scenarios;
- We implement a Siamese network architecture named Sentence-BERT with multiple loss functions, and compare its performance with the vanilla BERT model on sentence-pair inputs;
- We implement and experiment with two multi-task fine-tuning routines, the sequential and parallel routines, and discuss their effectiveness on producing robust and semantically rich BERT embeddings.

2 Related Work

In this section, we briefly review BERT fine-tuning strategies and variations for downstream tasks, and the general idea of additional pre-training as a major approach of language model fine-tuning.

2.1 BERT for downstream NLP tasks

BERT has delivered impressive results on a variety of downstream NLP tasks. In the original paper (Devlin et al., 2018), BERT was fine-tuned for 11 NLP tasks including the General Language Understanding Evaluation (GLUE) benchmark, which consists of the tasks and datasets in our project. Sun et al. (2019) provides an exhaustive overview of BERT fine-tuning methodologies and strategies on multiple text classification tasks. Zhang et al. (2020) studied several fine-tuning techniques in few-sample scenarios.

Multi-task fine-tuning is a less discussed aspect of BERT fine-tuning. Liu et al. (2015, 2019a) proposed a framework called MT-DNN to fine-tune BERT on multiple tasks simultaneously, which achieved state-of-the-art performance on the GLUE benchmark and produced embeddings with significantly stronger capability of domain adaptation compared to the original BERT embeddings. Sun et al. (2019) viewed multi-task fine-tuning as an approach to take full advantage of available data, but there was little evidence that multi-task fine-tuning could help generalization on sub-tasks.

A number of variations of BERT have be proposed to adapt to specific downstream NLP tasks. Sentence-BERT (Reimers and Gurevych, 2019) used a Siamese network architecture consisting of two BERT encoders with shared weights and custom loss function designs for sentence-pair inputs. Gao et al. (2021) proposed a contrastive learning framework called SimCSE for the semantic textual similarity task. TwinBERT (Lu et al., 2020) is developed in the context of sponsored search and has a structure similar to Sentence-BERT, but the input pair is not considered symmetric and hence the two BERT encoders do not share weights.

2.2 Additional pre-training

Many works have shown that further pre-training of language models on specific domains or tasks can improve the performance on downstream tasks. Domain-adaptive pre-training (DAPT) can help the embedding better align with the target domain, while task-adaptive pre-training (TAPT) allows the model to focus on the task-related corpus (Gururangan et al., 2020). In Gururangan et al. (2020), both DAPT and TAPT are conducted on a RoBERTa model with domain-specific corpus, e.g. news and reviews, and it is proved that a combination of the two approaches can significantly enhance the performance of the model. In Sun et al. (2019), it is shown that DAPT or TAPT alone can help performance on multiple text classification tasks.

The original pre-training techniques for BERT proposed by Devlin et al. (2018) are the masked language model (MLM) and next sentence prediction (NSP) tasks. Although it is later argued by several works such as Liu et al. (2019b) and Joshi et al. (2020) that NSP is ineffective or even harmful to the acquisition of a robust embedding, MLM remains to be a canonical approach to pre-train large language models such as RoBERTa (Liu et al., 2019b), ERNIE (Zhang et al., 2019), ALBERT (Lan et al., 2019), etc.

3 Approach

3.1 Masked language model (MLM) pre-training

The Masked Language Model task was inspired by the Cloze task (Taylor, 1953) and was first introduced to language model pre-training in Devlin et al. (2018). In this task, we mask part of the input text and require the model to predict the masked text. Intuitively, this forces the model to learn semantic and grammatical information in the corpus in a holistic way to be able to recover the masked text solely from its context.

An illustration of the MLM task is shown in Figure 1, where we follow the procedure in Devlin et al. (2018). First we randomly mask 15% of the input tokens (i.e. replace with the '[MASK]' token) and unmask some special tokens, e.g. the '[CLS]' and '[PAD]' tokens. It should be noted that there are

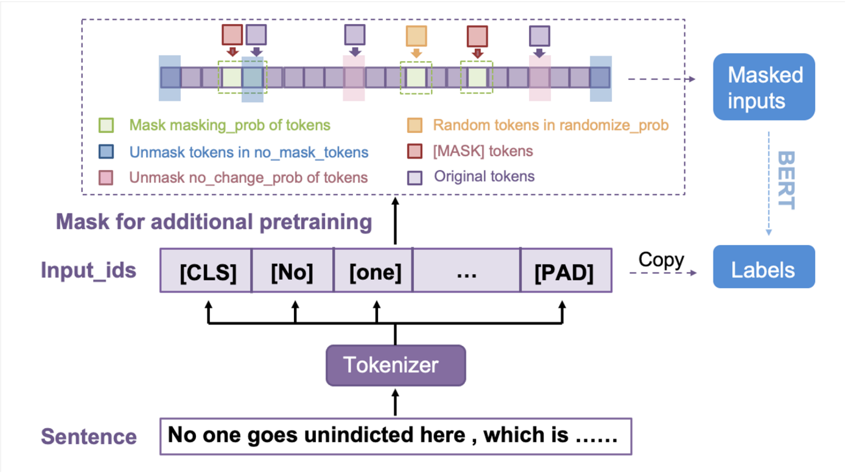
Masked inputs
Mask masking_prob of tokens
Unmask tokens in no_mask_tokens
Unmask no_change_prob of tokens
Random tokens in randomize_prob
[MASK] tokens
Original tokens
BERT
Mask for additional pretraining
Input_ids
[CLS]
[No]
[one]
...
[PAD]
Copy
Labels
Tokenizer
Sentence
No one goes unindicted here , which is ......

Figure 1: An illustration of the Masked Language Model task

no '[MASK]' tokens in fine-tuning, which might result in an invalid embedding in actual fine-tuning. To mitigate this effect, we replace 10% of the masked tokens with the original token, and another 10% with random tokens. Then we use the original input tokens as labels and cross-entropy as the loss function.

3.2 The Sentence-BERT model

Sentence-BERT (Reimers and Gurevych, 2019) is a model proposed to deal with a specific class of NLP tasks, namely tasks with sentence-pair inputs such as paraphrase detection and semantic textual similarity. It is based on the idea of Siamese networks, and is claimed to have comparable performance to vanilla BERT while greatly reducing the inference time (Reimers and Gurevych, 2019).

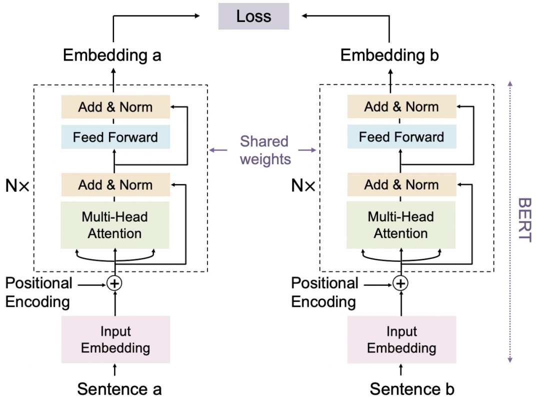

Figure 2: The Sentence-BERT model structure

The structure of the Sentence-BERT model is shown in Figure 2. It is simply composed of two identical BERT encoders with shared weights in training, and each encoder handle a sentence in the pair input. The embeddings of the two sentences are then passed to a cross layer, i.e. a customized loss function according to the specific task. In comparison, a vanilla BERT model concatenate the two sentences as the input to the encoder, and uses the embedding of the concatenated pair for classification or regression in the end.

3.2.1 Loss functions

The choice of loss function in the Sentence-BERT model has a significant impact on the performance, and requires careful design according to the nature of the downstream task. Here we list three loss functions for our paraphrase detection and semantic textual analysis. In the following discussion, l_i is the label for the input pair (x_i, y_i).

Cosine Similarity - Mean Squared Error loss. The cosine similarity - mean squared error loss is defined as

$$\mathcal{L}(x, y, l) = \frac{1}{n} \sum_{i=1}^{n} (l_i - S_C(x_i, y_i))^2,$$

where $S_C(\cdot, \cdot)$ is the cosine similarity defined as

$$S_C(u, v) = \frac{u \cdot v}{\|u\|_2 \|v\|_2}.$$

Reimers and Gurevych (2019) proposed to use this loss for the semantic textual analysis task as the labels l_i are already similarity scores. In our case $l_i \in [0, 5]$ and we can simply replace $S_C(\cdot, \cdot)$ with $5 \cdot S_C(\cdot, \cdot)$.

Contrastive loss. The contrastive loss function was first introduced by Hadsell et al. (2006) in the context of dimensionality reduction. In our task it is defined as

$$\mathcal{L}(x, y, l) = \frac{1}{n} \sum_{i=1}^{n} \left[l_i D(x_i, y_i) + (1 - l_i) \max(0, m - D(x_i, y_i)) \right],$$

where $D(\cdot, \cdot)$ is a distance on the space of sentence embeddings, and $m \in (0, 1)$ is a margin value. Here we choose $D = D_C(\cdot, \cdot) = 1 - S_C(\cdot, \cdot)$, namely the cosine distance, and $m = 0.5$. Intuitively, the loss penalize large distances within similar ($l_i = 1$) pairs and small distances within dissimilar ($l_i = 0$) pairs. Empirically we found that for continuous label $l_i \in [0, 5]$, directly replacing l_i with $l_i/5$ can deliver satisfactory performance.

Multiple Negative Ranking loss. The Multiple Negative Ranking (MNR) loss function was introduced in Henderson et al. (2017). It's originally designed for datasets in the form of $\{(a_i, b_i)\}$ where a_i and b_i are similar while a_i and b_j are dissimilar for all $i \neq j$. For our paraphrase detection dataset, we need to modify it as

$$\mathcal{L}(x, y, l) = -\frac{1}{\sum_{i=1}^{n} \mathbb{I}(l_i = 1)} \sum_{\substack{i=1 \\ l_i=1}}^{n} \left[S_C(x_i, y_i) - \log \sum_{\substack{j=1 \\ l_j=1}}^{n} e^{S_C(x_i, y_j)} \right].$$

It can be seen that similarity scores within similar pairs are maximized, and the similarity scores between a sentence x_i and sentences y_j in other pairs are simultaneously minimized. It is worth noting that we are only using $l_i = 1$ samples due to this modification, which might result in a significant performance gap compared to other loss functions. In fact, the discarded pairs (x_i, y_i) with $l_i = 0$ may include more subtle semantic differences compared to (x_i, y_j) pairs with $i \neq j$.

3.3 Multi-task fine-tuning routines

To fine-tune the BERT model on multiple downstream tasks, we propose two training routines, parallel training and sequential training. An illustration of the two routines is shown in Figure 3.

In the parallel training routine, we shift between tasks at the end of each batch. Due to the difference in dataset sizes of different tasks, we cycle through smaller datasets in one epoch on the largest dataset to produce the same number of batches for all datasets. One epoch on the largest dataset counts as one global epoch.

In the sequential training routine, we shift between tasks at the end of epochs. To balance the number of batches on datasets with different sizes, we train N_i sub-epochs on task i such that $N_i B_i \approx C$ for all i, where B_i is the number of batches of one sub-epoch on the dataset of task i. One global epoch consists of $N_1 + \cdots + N_k$ sub-epochs for k tasks. We point out that if we set the number of sub-epochs on the largest dataset as 1, i.e. $\min_{1 \leq i \leq k} N_i = 1$, then the sequential and parallel training routines have the same number of batches on each task within one global epoch.

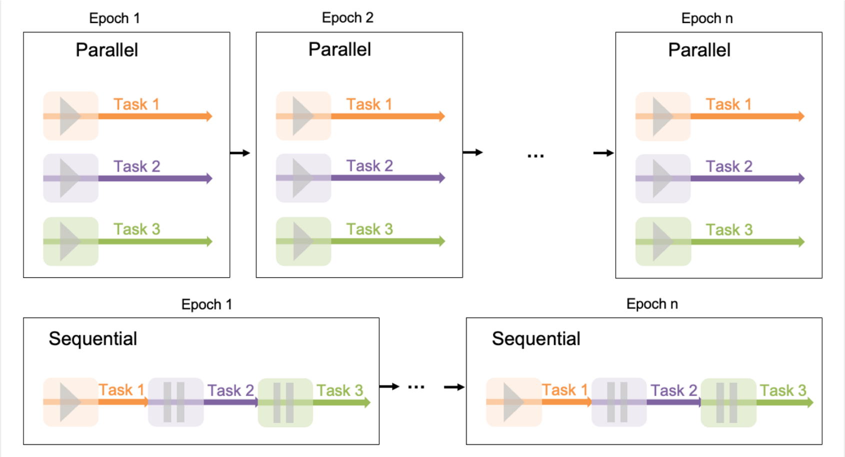

Figure 3: An illustration of the sequential and parallel multi-task fine-tuning routines

4 Experiments

4.1 Data

We use the datasets specified in the default project handout (CS224N, 2023). Specifically, we used the SST dataset for the sentiment analysis task, the Quora dataset for the paraphrase detection task and the STS dataset for the semantic textual similarity task. Inputs for the first task are individual sentences, while inputs for the other two tasks are sentence pairs.

4.2 Evaluation method

Our evaluation methods follow the requirements of the default project handout (CS224N, 2023). For the SST sentiment analysis task and the Quora paraphrase detection task, we use the classification accuracy as evaluation metrics. For the STS semantic textual anlysis task, we use the Pearson correlation coefficient (PCC) between the true similarity scores and the predicted ones, i.e.

$$r(y, \hat{y}) = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2 \sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})^2}}.$$

4.3 Experiment-1: Investigating additional pre-training

In this section, we perform additional pre-training on the SST and STS datasets with the unsupervised masked language model task as introduced in Section 3.1. Then, we used the additional pre-training weights as starting points of fine-tuning, investigating effectiveness of additional pre-training on fine-tuning by comparing the performances of BERT models with and without additional pre-training on the two tasks.

We perform 50 epochs of task-adaptive additional pre-training (i.e. pre-train on the dataset corresponding to the task), and 5 epochs of fine-tuning on both SST and STS datasets. For both pre-training and fine-tuning we use a learning rate of 1e-5. Results with and without additional pre-training (ADDPRETRAIN) are shown in Table 1. We observe a positive effect of additional pre-training on the STS dataset, but a negative effect on the SST dataset.

4.4 Experiment-2: Sentence-BERT with different loss functions

As mentioned in Section 4.1, inputs for the second and third tasks are sentence pairs, and both these two tasks aim to find out the semantic relationship between the two sentences in each pair. Therefore, we experiment with the Sentence-BERT architecture as introduced in Section 3.2, and compare the performances with a vanilla BERT model.

	SST	STS
Without AddPretrain	**0.524**	0.863
With AddPretrain	0.515	**0.870**

Table 1: Additional pre-training experiments results

Table 2 shows a comparison between the vanilla BERT model and the Sentence-BERT model with different loss functions selected according to the specific tasks. All results are obtained with 10 epochs of single-task fine-tuning, with a learning rate of 1e-5. It could be observed that there exists a significant performance gap between the Sentence-BERT model and the vanilla BERT model, regardless of the choice of loss functions.

		Quora	STS
Vanilla Bert		**0.889**	**0.869**
	CosSim+Mse	–	0.762
Sentence-Bert	Contrastive	0.748	0.760
	Mnr	0.474	–

Table 2: SBERT and BERT results on Quora and STS datasets

4.5 Experiment-3: Investigating multi-task fine-tuning routines

We experiment with the multi-task fine-tuning routines proposed in Section 3.3 and compare the results with our baseline and single-task fine-tuning. For the baseline we use the BERT-base-uncased weights and train the final classification/regression layer for 10 epochs with a learning rate of 1e-3, and the BERT encoder weights frozen. For single-task fine-tuning we run 5 epochs for Quora and 10 epochs for SST and STS (all with a learning rate of 1e-5) to obtain the best result available to our model. For the sequential and parallel routines we run 5 global epochs (defined in Section 3.3) with a learning rate of 1e-5; each sequential global epoch consists of 16, 1, and 23 sub-epochs on the three datasets, respectively. For sequential/parallel routines with additional pre-training, we pre-train for 2 epochs on the full dataset (SST+Quora+STS) and fine-tune with the routines for 3 epochs, with learning rate all set to 1e-5. The results are shown in Table 3. We also report our results of the parallel fine-tuning routine on the test sets in Table 4.

	SST	Quora	STS
Baseline (Bert-Base-Uncased)	0.408	0.736	0.540
SingleTask (Best)	**0.533**	**0.889**	**0.874**
Sequential	0.387	0.863	**0.879**
Parallel	**0.505**	**0.877**	0.863
AddPretrain+ Sequential	0.394	0.821	0.876
AddPretrain+ Parallel	**0.505**	0.873	0.870

Table 3: Results for different multi-task fine-tuning routines

	SST	Quora	STS
Parallel	0.514	0.879	0.859

Table 4: Results for different multi-task fine-tuning routines

We may observe that the parallel multi-task fine-tuning routine could deliver robust embeddings leading to performance comparable to single-task fine-tuning on each downstream task. The sequential routine achieves the best results on the STS dataset, acceptable on the Quora dataset, and results worse than baseline on the SST dataset, which is somehow expected due to our training sequence. We also observe different effects of additional pre-training on different datasets in the multi-task scenario. A more detailed analysis is provided in Section 5.

5 Analysis

We have observed a positive effect of additional pre-training on the STS dataset and a negative effect on the other two datasets in both single-task and multi-task scenarios. It is possible that the effect of task-adaptive pre-training is task/data-dependent. Another possibility is that, due to the high quality of the original BERT-base-uncased weights, an aggressive task-specific pre-training on a small corpus, which is indeed the case of our dataset, could potentially erase the general semantic or grammatical knowledge in the pre-trained weights and consequently harm generalization. The phenomenon is known as catastrophic forgetting (McCloskey and Cohen, 1989), and has also been observed by (Sun et al., 2019) when fine-tuning BERT with an over-aggressive learning rate.

A significant performance gap between the Sentence-BERT model and the vanilla BERT model is observed on all three tasks. A similar phenomenon has been observed by Lu et al. (2020) and their explanation was that BERT can learn through deep interactions between the sentences in the pair, while Sentence-BERT easily saturate due to the simplicity of its combination layer (merely one loss function). Here we propose another possible explanation from a different perspective: the vanilla BERT model can extract semantic relationship information as the embedding, while the Sentence-BERT model is forced to extract information required for the task from two single sentences. For different inputs in the dataset, the aspects of single-sentence semantic information relevant to the task can vary in a considerable range. Taking the paraphrase detection task as an example, some input pairs might emphasize on understanding certain grammatical structures, while others might emphasize on distinguishing between notional words. The vanilla BERT model can cover all these aspects using its powerful encoder structure, while the limited single-sentence embedding size in the sentence-BERT model can only capture a limited number of these aspects, which becomes a substantial bottleneck of the model.

We have also observed a huge performance gap between the MNR loss and other loss functions, which is most likely a consequence of the mismatch between the loss function and the dataset, as conjectured in Section 3.2.1.

From the experiments on multi-task fine-tuning routines, we observe that the parallel multi-task fine-tuning routine could deliver robust embeddings leading to performance comparable to single-task fine-tuning on each sub-task, with a minor performance reduction due to the potential underlying conflict of different learning goals. In contrast, performance of sequential fine-tuning routine could easily be affected by designs such as the training sequence and number of sub-epochs on each task in a global epoch. Specifically, the learned knowledge on tasks appearing earlier in the training sequence are forgotten in the training process of tasks appearing later. Although a carefully designed schedule (choices of number of sub-epochs, and ad hoc sequence design) might mitigate this issue, we believe that it is a intrinsic flaw of the sequential routine when the datasets are moderately large, and hence a parallel routine should always be advised.

6 Conclusion and Future Works

In this project, we have worked on three major aspects of BERT fine-tuning: additional pre-training with MLM, the Sentence-BERT structure and loss functions for pair inputs, and multi-task fine-tuning routines.

In terms of additional pre-training, we have only observed partially positive effects on the performance. As our current pre-training is limited to task-adaptive pre-training, a main direction of future work is to introduce other in-domain datasets and increase the dataset sizes to better exploit the potential of pre-training.

We have observed a significant performance gap between the Sentence-BERT model and the vanilla BERT model and provided a possible explanation. According to our conjecture, it could be further examined whether a larger output embedding size of Sentence-BERT can reduce this gap.

Finally, we have concluded that a parallel multi-task fine-tuning routine can deliver robust embeddings comparable to single-task fine-tuning on multiple downstream tasks simultaneously. Other routines can be designed and experimented with, e.g. merging and shuffling the datasets (Liu et al., 2019a), or using adaptive batch sizes for different datasets, etc.

References

CS224N. 2023. Cs 224n: Default final project: minbert and downstream tasks. `https://web.stanford.edu/class/cs224n/project/default-final-project-bert-handout.pdf`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

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

Wenhao Lu, Jian Jiao, and Ruofei Zhang. 2020. Twinbert: Distilling knowledge to twin-structured compressed bert models for large-scale retrieval. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2645–2652.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Quora. First quora dataset release: Question pairs. `https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs`.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

SST. Stanford sentiment treebank (sst) dataset. https://nlp.stanford.edu/sentiment/treebank.html.

STS. Sem 2013 shared task: Semantic textual similarity. https://aclanthology.org/S13-1004.pdf.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*, pages 194–206. Springer.

Wilson L Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2020. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*.