minBERT and Downstream Tasks

Stanford CS224N Default Project

Qian Zhong
Stanford University NDO Program
qzhongx@stanford.edu

Abstract

The project implemented and trained a mini-BERT model to perform sentiment analyses, paraphrase detection, and semantic textual similarity tasks. Stanford Sentiment Treebank (SST) dataset, Quora dataset, and the SemEval STS benchmark dataset are used to train and evaluate the model on the three tasks. With a pretrained mini-BERT as the base, task-specifc layers were added on the top for each downstream task. We applied the Multi-Task Deep Neural Networks (MT-DNN) introduced in Liu et al. (2019) to train the model on multiple tasks simultaneously. By comparing with directly finetuning the model for individual tasks, we showed that MT-DNN could efficiently utilize training data and having tasks benefiting from transfer learning. The effects of dataset imbalance in MT learning was also shown from experiments. For individual tasks, different hyper parameters and loss functions were experimented regarding improving model performance. We showed that more training data helps in a large sense. Combining the MT-DNN and improvements made on individual tasks, we obtained our best model performance on the test leaderboarding, with the accuracy/correlation scores being 0.511, 0.841, 0.68 for the three tasks respectively.

1 Key Information to include

- Mentor: Andrew Hyungmin Lee
- External Collaborators (if you have any): n/a
- Sharing project: n/a

2 Introduction

Bidirectional Encoder Representations from Transformers, or "BERT"Devlin et al. (2018), has become one of the most influential and widely-used models in natual language processing (NLP) since its release in 2018. It captures the bidirectional context of words in a sentence, allowing it to understand the meaning of words based on the surrounding context. BERT can be pre-trained on large text corpora using unsupervised learning techniques, followed by fine-tuning on specific downstream tasks with labeled data, making it adaptable to various NLP tasks such as text classification, named entity recognition, and question answering. It achieved state-of-the-art performance on a wide range of NLP benchmarks with pretraining and finetuning, surpassing its previous models on various NLP tasks.

In 2019, Liu et al. (2019) proposed MT-DNN (Multi-Task Deep Neural Network) model that extends the capabilities of BERT for multi-task learning. MT-DNN is designed to simultaneously perform multiple NLP tasks, such as text classification, sequence labeling, and sentence pair matching. By jointly training on multiple tasks, MT-DNN aims to leverage the shared representations across tasks to improve overall performance. It achieved competitive results and outperforming benchmark scores comparing to task-specific model including finetuned BERT.

In this project, we build a mini BERT model with the transformer architecture, the attention mechanism, and the BERT tokenizer text preprossing framework. The model is applied on three downstream NLP tasks:

- Sentiment Analysis: classifying the polarity of a given text.
- Paraphrase Detection: finding paraphrases of texts in a large corpus of passages.
- Semantic Textual Similarity: measuring the degree of semantic equivalence bewteen texts.

Pretrained weights are used for the model as backbone. We add task-specific layers on the top and finetune on labeled datasets including Stanford Sentiment Treebank (SST) dataset, Quora dataset, and the SemEval STS benchmark dataset for the three tasks. MT-DNN is adopted to train the model performing three tasks simultaneously. The effects of MT-DNN on the given tasks are studied. It shows that MT-DNN efficiently utilizes training data and similar tasks could benefit from transfer learning. Data imbalance among the datasets are discussed. We also reported experiment results with hyperparameter tuning, different loss functions, and different head layer structures.

3 Related Work

The main techniques we studied with the project is the MT-DNN (Multi-Task Deep Neural Network) introduced in Liu et al. (2019). It applies the multi-task learning on pre-trained langurage models for learning representations across multiple natural langurage understanding (NLU) tasks.

The original work indicates that the multi-task learning, inspired by the human learning process, shows advantages in two aspects: first the effective use of supervised data as it can combine labeled data from many related tasks; and secondly, it leverages a relgularization effect in that by training towards performing multiple tasks, it mitigates the risk of overfitting to a specific task and making the learning more robust. And in the same time, the pretrained model, especially most prominent ones like GPT (Radford et al., 2018) and BERT (Devlin et al., 2018) shows capability to learn universal language representations from large amounts of unlabled data. Thus, Liu et al. (2019) proposes to combine multi-task learning and language model pretraining, with the goal to leverage strengths of both sides and boost performances on various NLU tasks with limited amount of domain data. It incorporates BERT as shared text encoding layers and adds task-specific top layers for different NLU tasks including single-sentence classification, pairwise text classification, text similarity, and relevacne ranking. The paper presents evaluation results of the MT-DNN on NLU benchmarks including GLUE (Wang et al., 2018), SNLI (Khot et al., 2018), and SciTail tasks, and compares it with those of the state-of-the-art models at the time to show the effectiveness of the proposed approach.

We also experimented with the pooling strategy introduced in the "Sentence-BERT" (Reimers and Gurevych, 2019). Unlike traditional methods that use only the final hidden state of the [CLS] token in BERT for sentence representation, they explore various pooling strategies to capture the entire contextual information encoded by BERT. The strategies includes mean pooling and max pool, which takes the average or the maximum value of all token embeddings in the sentence, as well as combinations of the two strategies. These strategies was experimented to find the most effective approach for generating sentence embeddings that preserve semantic similarity. This is adopted in the current project for improving the model performance on the semantic textual similarity task.

4 Approach

4.1 Inidividual tasks finetuning

We used the model architecture shown in Fig. 1. All tasks share the same lower layers from mini-BERT with pretrained weights, while the top layers represent task-specific outputs described in the following.

Sentiment Analysis: Given an input X, assuming $\mathbf{x}$ is the contextual embedding l_2 of the token [CLS], the probability that X is labeled as class c (i.e. the sentiment) is predicted by a logistic regression with softmax:

$$P_r(c|X) = \text{softmax}(\mathbf{W}_{SST}^T \cdot \mathbf{x}) \tag{1}$$

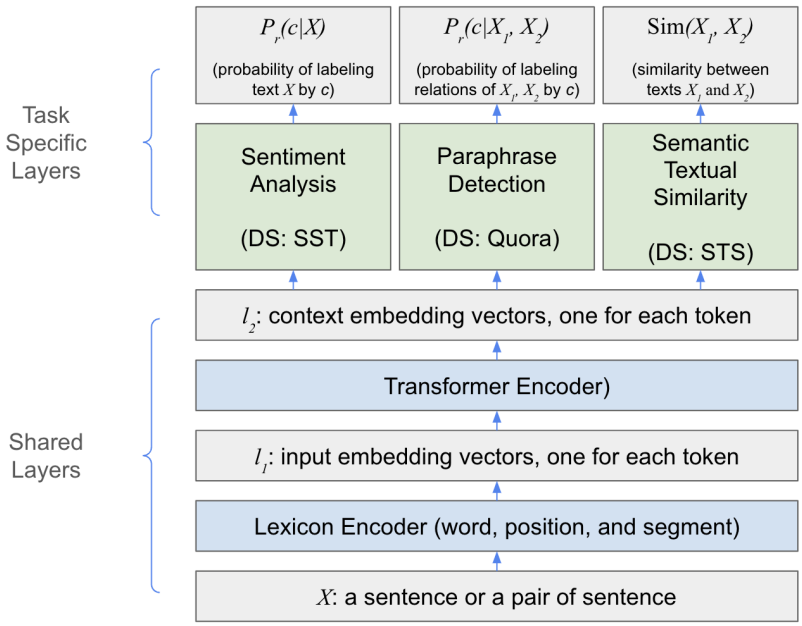

Figure 1: Model architecture used in the project. Inspired by Figure 1 in Liu et al. (2019).

Paraphrase Detection: Given inputs X_1 and X_2 with $\mathbf{x}_1$ and $\mathbf{x}_2$ being the contextual embeddings, we concatenate the two inputs and designed that the probability that the relation bewteen X_1 and X_2 is labeled as class c (i.e. whether they are paraphrases or not) can be predicted by a logistic regression with softmax with the concatenated input:

$$P_r(c|X_1, X_2) = \text{softmax}(\mathbf{W}_{para}^T \cdot [\mathbf{x}_1; \mathbf{x}_2]) \tag{2}$$

Semantic Textual Similarity Given inputs X_1 and X_2, we use cosine similarity to compute the two inputs and scale the output to the expected range of $[0, 5]$. The mean squared error is used as objective during training. We experimente different pooling strategies as in Reimers and Gurevych (2019) to extract the contextual embeddings $\mathbf{x}_1$ and $\mathbf{x}_2$. We also test using a linear transformation of concatenated inputs to compute the similarity score .

$$\text{Sim}(X_1, X_2) = (1 + \text{cos-sim}(\mathbf{x}_1 \cdot \mathbf{x}_2)) * 5/2 \tag{3}$$

4.2 Multi-task learning

The multi-task learning uses the same model structure shown in Fig. 1. The algorithm introduced in Liu et al. (2019) is adopted to jointly train the model on three tasks, as illustrated in Fig. 2. During the multi-task learning, mini-batch based stochastic gradient descent (SGD) is used to learn the parameters of the model, including all shared layers and task-specific layers. In each epoch, one mini-batch from one dataset (among three training datasets) is selected and the model is updated acoording to the objective of the corresponding task. As pointed out in Liu et al. (2019), this approximately optimizes the sum of all multi-task objectives.

By default, we truncate the datasets so that each contains approximately the same number of samples that can be used in multi-task training. We additionally examine the impact of data imbalance by augmenting one dataset with more samples and adjusting the sampling strategy to ensure that datasets are selected with a probability proportional to their sample sizes.

Algorithm 1: Training a MT-DNN model.

Initialize model parameters Θ randomly.
Pre-train the shared layers (i.e., the lexicon encoder and the transformer encoder).
Set the max number of epoch: $epoch_{max}$.
//Prepare the data for T tasks.
for t *in* $1, 2, ..., T$ **do**
  Pack the dataset t into mini-batch: D_t.
end
for $epoch$ *in* $1, 2, ..., epoch_{max}$ **do**
  1. Merge all the datasets:
   $D = D_1 \cup D_2 ... \cup D_T$
  2. Shuffle D
  for b_t *in* D **do**
    //b_t is a mini-batch of task t.
    3. Compute loss : $L(\Theta)$
      $L(\Theta)$ = Eq. 6 for classification
      $L(\Theta)$ = Eq. 7 for regression
      $L(\Theta)$ = Eq. 8 for ranking
    4. Compute gradient: $\nabla(\Theta)$
    5. Update model: $\Theta = \Theta - \epsilon\nabla(\Theta)$
  end
end

Figure 2: Algorithm of the MT-DNN model for representation learning from Liu et al. (2019).

5 Experiments

5.1 Data and Evaluation method

Subsets from the Stanford Sentiment Treebank (SST) dataset, Quora dataset, and the SemEval STS benchmark dataset are used to train and evaluate the model for the three tasks respectively.

The **SST** consists of $11,855$ single sentences extracted from movie reviews, parsed with the Stanford parser and resulting in $215,154$ unique phrases annotated by human judges. The subset used for training the mini-BERT contains $8,544$ examples for train, $1,101$ examples for dev, and $2,210$ examples for test. Each example is labeled with one of the five categories: negative, somewhat negative, neutral, somewhat positive, positive. Accuracy fo the sentiment prediction results is used to evaluate the model.

The **Quora** dataset consists of $400,000$ question pairs with labeld indicating whether particular instances are paraphrases of one another. The subset used in this project contains $141,506$ examples for train, $20,215$ examples for dev, and $40,431$ examples for test. Given the binary labels of the dataset, accuracy is used to evaluate the model performance.

The **SemEval STS** dataset in this project contains $6,041$ examples for train, 864 examples for dev, and $1,726$ examples for test. As in the original SemEval (Agirre et al., 2013) paper, Pearson correlation of the true similarity values against the predicted similarity values are used to test the model on the dataset.

5.2 Experimental details

We started with pretraining and finetuning the model for sentiment analysis task on both *SST* and *CFIMDB* datasets. Table 1 shows accuracy obtained by the pretrained and finetuned model for the sentiment classification task. We observed noticeable enhancements by the finetuning techniques. Thus for the following study, finetuning is used for all model training.

The model training and evaluation are run in the Colab with T4 GPU. Default params were used to config the model if not stated otherwise: hidden size of 768, dropout rate of 0.3, batch size of 8, learning rate of $1e^{-5}$ for finetuning.

For multi-task learning, since *Quora* dataset contains samples that are an order of magnitude larger than the others, we truncated the three training datasets to be of the same length of the smallest

Model	SST	CFIMDB
miniBERT-pretrain	0.400	0.784
miniBERT-finetune	0.517	0.967

Table 1: Table comparing accuracy of the pretrained and finetuned model on *SST* and *CFIMDB* datasets.

dataset. Because of the truncation, some of the training data may not be used. To avoid this, we introduced random sampling of the data in each dataset so that at each epoch, the training dataset will be resampled from the original training dataset and used for the epoch training. For model saving, after each epoch evaluation, we will save the model if it achieves better scores on two out of three tasks.

6 Results and Analysis

Our test scores obtained from the TEST leaderboard are: 0.511, 0.841, 0.68, with the total being 0.731.

6.1 Results of multi-task learning

With the same default params, Table 2 compares results of finetuning on individual tasks (SST/Quora/STS), applying MT-DNN to joinly finetune for three tasks (MT), and further finetuning on each dataset based on MT-DNN trained model (MT+SST/Quora/STS).

Model	SST	Quora	STS
miniBERT-SST	**0.520**	0.628	0.167
miniBERT-Quora	0.164	0.818	0.265
miniBERT-STS	0.231	0.459	0.490
miniBERT-MT	0.482	0.687	0.526
miniBERT-MT+SST	0.514	0.697	0.487
miniBERT-MT+Quora	0.351	**0.842**	0.531
miniBERT-MT+STS	0.475	0.686	**0.553**

Table 2: Table comparing evaluation results of the finetuned models on three NLU tasks.

Looking at the Quora and STS tasks, the model trained with MT-DNN achieved better performance than the directly finetuned model (0.842 vs 0.818 and 0.553 vs 0.490.) This demontrates that the model could benefit from knowledge learned from all three tasks and transfer those when performing a single task. In particular for STS task, the SemEval STS dataset contains the lowest number of samples comparing to the other two. But the task in essense is similar to paraphrase detection on Quora in the way that both can use understanding of similarities of two inputs. Thus, with MT learning, the model can learn from Quora with large samples and transfers that when conducting the SST task, which improves the higher correlation score; and vice versa. In short, the MT-DNN shows its strength in efficient use of training samples and taking advantage of transfer learning.

For the SST task, the model obtains a good score 0.514 with MT-DNN but has its best performance 0.520 with simply finetuning. This could be due to the fact that the sentiment classification task is less similar to the other two tasks. Thus, the context learned from the other tasks may not benefit the model as much on the STS task; thus, the directly finetuning gives the best result.

To further see how the scale of training data affects the model performance, we conducted a second round of finetuning using the $MT + Quora$ model as the base. Table 3 shows the result. It can be seen that more data from relevant tasks, e.g. Quora and STS, helps improve the model performance. The model trained on both Quora and STS achieves the best score on STS. However, this does not

Model	SST	STS
miniBERT-SST	**0.520**	-
miniBERT-MT+SST	0.514	-
miniBERT-MT+Quora+SST	0.517	-
miniBERT-STS	-	0.490
miniBERT-MT+STS	-	0.553
miniBERT-MT+Quora+STS	-	**0.567**

Table 3: Table shows results with more training data.

hold if the two tasks are less relevant. On SST, directly finetuning the model on SST still gives the best score.

Model	SST	Quora	STS
miniBERT-MT-1:1:1	0.482	0.687	0.526
miniBERT-MT-1:2:1	0.497	0.736	0.526
miniBERT-MT-1:4:1	0.253	0.625	0.085

Table 4: Table shows results with imbalanced data. 1:x:1 indicates the ratio of samples in SST:Quora:STS datasets.

Though more data could improve the model performance, we think balanced datasets are critical for MT-learning. Table 4 shows our results of increasing the Quora dataset by 2x and 4x during MT-DNN training. It can be seen that with 2x Quora, the model performance increased a bit with boost from more data. But when the dataset ratio increased to $1:4:1$ for SST:Quora:STS, the model scores are adversely impacted and decreased.

6.2 Improvements on individual tasks

Model	STS
miniBERT-[CLS]-linear	0.403
miniBERT-[CLS]-cos	0.490
miniBERT-mean pooling-cos	**0.707**
miniBERT-mean pooling-cos-small lr	0.649
miniBERT-max pooling-cos	0.457
miniBERT-mean-max pooling-cos	0.627

Table 5: Table showing experiment results on the STS task.

To improve model performance, we experiment with different techniques for individual tasks.

For STS, we adopted the pooling strategies described in Reimers and Gurevych (2019). In Table 5, [CLS] indicates by default using the embedding of [CLS] to represent the sentence context, whereas mean and max pooling indicate taking the average or the maximum value of all token embeddings in the sentence, as well as combinations of the two. The result shows the mean pooling gives the best score. However, when combining it with MT-DNN, we found this pool strategy makes the model less trainable as the dev score stops to show improvements at the early stage. We have tried change the head layers to be a dense layer or using a smaller learning rate, but hardly see improvements.

For sentiment classification task on SST, we experimented with different learning rate and different number of fully connected layers on top of miniBERT. The results are presented in Table 6. It shows that the best was achived with lr being $1e-5$ and 2 fc layers; but all results are close and do not differ much.

Model	SST
miniBERT-1 fc, lr 1e-5	0.520
miniBERT-2 fc	**0.526**
miniBERT-3 fc	0.520
miniBERT-lr 1e-4	0.46
miniBERT-lr 1e-6	0.510
miniBERT-lr 1e-7	0.498

Table 6: Table showing experiment results on the SST task.

7 Conclusion

We implemented a mini-BERT and applied it to sentiment analyses, paraphrase detection, and semantic textual similarity tasks. The tasks share lower layers from mini-BERT with task-specific layers on top. Pretrained weights were used as base; the model params of both lower and top layers were finetuned on task datasets.

We applied the MT-DNN to jointly train the model to perform three tasks. We found that through MT learning, the model can make efficient use of training samples and achieve better evaluation scores with learned context from similar tasks. But with less similar task, the MT learning did not perform as well compared to directly finetuning the model for the individual task. During MT learning, more training data helps improve the model in general. But the number of samples in multiple datasets need to be reasonably balanced. A single dataset that dominants the sample set can harm the model performance even with more training data.

To tasks with less satisfactory evaluation scores, we experimented different techniques to improve the model performance, including tuning the head layer structure, changing the learning rate, and using different pooling strategy to extract sentence embedding.

The final accuracy/correlation scores we achieved on the TEST leaderboard are 0.511, 0.841, 0.68 with total being 0.731.

References

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. * sem 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity*, pages 32–43.

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*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

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

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.