Individual Final Report

Natural Language Processing

Final Project

December 9, 2021

Yuchen Ma

Introduction

Our project utilizes the transfer learning method to apply the transformers models from the Hugging Face website to solve the GLUE benchmark tasks. First, we loaded the BERT and the RoBERTa pre-trained models, added an LSTM layer, and fine-tuned the model. Then, we followed the pipeline of the lecture codes and the Hugging Face transformers guide to load and process the data and train and evaluate models. The main coding part of our project is to build a class of custom model. Then, we compared two models and found the RoBERTa worked better.

Description of individual part

I make the preprocess for each task. As the code shown below.

```
task_to_keys = {
    "cola": ("sentence", None),
    "mnli": ("premise", "hypothesis"),
    "mnli-mm": ("premise", "hypothesis"),
    "mrpc": ("sentence1", "sentence2"),
    "qnli": ("question", "sentence"),
    "qqp": ("question1", "question2"),
    "rte": ("sentence1", "sentence2"),
    "sst2": ("sentence", None),
    "stsb": ("sentence1", "sentence2"),
    "wnli": ("sentence1", "sentence2"),
}
sentence1_key, sentence2_key = task_to_keys[task]
```

In [25]:

```
def tokenize_function(example):
    if sentence2_key is None:
        return tokenizer(example[sentence1_key], truncation=True)
    return tokenizer(example[sentence1_key], example[sentence2_key], truncation=True)
```

In [26]:

```
num_labels = 3 if task.startswith("mnli") else 1 if task=="stsb" else 2
tokenized_datasets = dataset.map(tokenize_function, batched=True)
data_collator = DataCollatorWithPadding(tokenizer=tokenizer)

tokenized_datasets = tokenized_datasets.remove_columns(["sentence", "idx"])
tokenized_datasets = tokenized_datasets.rename_column("label", "labels")
tokenized_datasets.set_format("torch")
tokenized_datasets["train"].column_names
```

For each task, they have different structures. Therefore, in the preprocess, they should be tokenized and are split into data and label separately.

Description of the Portion of my work

In the project, I make the preprocess for each task. Then, we can run codes of each task. I run some tasks in the colab. However, some tasks are too large to run in the colab. Therefore, with the help of Yuan, I got all results for all tasks which trained by Bert model. In the end, I combine all results and compare them with the results got by Abdulaziz. From analyzing results, I find that Roberta works better than Bert. In the report and slides, I make the summary and conclusion.

Results

Here are my training results

Task	Batch size	Epoch	Learning rate	BERT
CoLA	64	30	5e^5	Matthew's Corr: 0.55727640631709
SST-2	64	5		Accuracy: 0.9243
MRPC	16	5		F1: 0.9091 Accuracy: 0.8701
STS-B	8	5		Pearson: 0.8780 Spearman: 0.8757
QQP	32	5		F1: 0.8789 Accuracy: 0.9095
MNLI	32	5		Accuracy: 0.8338
QNLI	32	5		Accuracy: 0.9028
RTE	16	5		Accuracy: 0.6245
WNLI	4	30		Accuracy: 0.5493

This is result I got from Bert model. Then, after I compare it with the result from Roberta model, I find that for each task, RoBERTa mode always has a better result, although the difference is not much.

Summary and conclusion

In the project, we learn how to build the architecture for Bert and RoBerta and analyze natural language through nine tasks. For each task, we use different metrics to evaluate their performance. When we run the code, we also met some problems due to the large size of the task, so we change hyperparameters and upgrade the GPU to run our codes. For our results, we use different metrics to get results in two models and conclude that Bert and Roberta both work well for nine tasks and Roberta works better. There are still many transformers which have similar functions. It is also interesting for us to explore in the future.

References

1. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz M., Davison J., Shleifer, S., Platen, P.V., Ma, C., Jernite Y., Plu, J., Xu, C., Scao, T.L., Gugger, S., Drame, M., Lhoest, Q., Rush, A.M. (2020, October). Transformers: State-of-the-Art Natural Language Processing, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (pp. 38-45). Association for Computational Linguistics. Retrieved December 7, 2021, from https://www.aclweb.org/anthology/2020.emnlp-demos.6.

2. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. R. (2019, February 22). *Glue: A multi-task benchmark and analysis platform for natural language understanding*. arXiv:1804.07461. Retrieved December 7, 2021, from https://arxiv.org/abs/1804.07461.

3. Hugging Face Hub. (n.d.). *Text Classification on GLUE*. Retrieved December 8, 2021, from https://colab.research.google.com/github/huggingface/notebooks/blob/master/examples/text_classification.ipynb#scrollTo=7k8ge1L1IrJk.

4. Grable, A., & Geet D'Sa, A. (n.d.). *How can I add a Bi-LSTM layer on top of Bert Model?* Stack Overflow. Retrieved December 7, 2021, from https://stackoverflow.com/questions/65205582/how-can-i-add-a-bi-lstm-layer-on-top-of-bert-model?rq=1.