Statistical Natural Language Processing

Final Project Report (ELEC-E5550)

General language understanding evaluation using many variants of BERT model

Nguyen Xuan Binh (887799) (binh.nguyen@aalto.fi)
Nguyen Long (783903) (long.l.nguyen@aalto.fi)

Aalto University

Statistical Natural Language Processing Project Report

Author
Nguyen Xuan Binh (887799) (binh.nguyen@aalto.fi)
Nguyen Long (783903) (long.l.nguyen@aalto.fi)

Title
General language understanding evaluation using many variants of BERT model

School School of Science

Degree programme
Master of Science in Computer, Communication and Information Sciences (CCIS)

Major Machine Learning, Data Science and Artificial Intelligence (MACADAMIA)

Advisor Kurimo Mikko **Code** SCI3044

Date 19 April 2024 **Pages** 15 **Language** English

Abstract

BERT is a transformers model pretrained on a large corpus of English data in a self-supervised fashion. It was pretrained with two objectives: Masked language modeling (MLM) and Next sentence prediction (NSP) [1]. In addressing the project of using pre-trained BERT models for General Language Understanding Evaluation (GLUE) tasks, the focus will be on exploring and comparing the performance of cased and uncased base of BERT. For performance analysis, GLUE is a benchmark consisting of various language tasks for evaluating and analyzing natural language understanding systems [2]. There are lots of dataset and challenges, but it is decided that this report will try to measure the performance of BERT models on two tasks.

Task 1: Quora question pairs (QQP)

This dataset consists of 400,000 lines of potential question duplicate pairs. Each line contains IDs for each question in the pair, the full text for each question, and a binary value that indicates whether the line truly contains a duplicate pair. The task is thus binary classification and the performance metric is accuracy category.

Task 2: Stanford Sentiment Treebank (SST-2)

This dataset is a corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiment in language. It was parsed with the Stanford parser and includes a total of 215,154 unique phrases from those parse trees, each annotated by 3 human judges. The review can be classified with either positive v.s negative review. The task is thus binary classification and the performance metric is accuracy category.

The project analyses the performance metrics to conclude which BERT model variant (BERT base uncased, RoBERTa, DeBERTa, ALBERT, DistilBERT) performs the best in each respective tasks and provides explanation for such differences in performance.

Keywords BERT, GLUE, QQP, SST2, RoBERTa, DeBERTa, ALBERT, DistilBERT

urn https://aaltodoc.aalto.fi

Contents

1 Introduction 1
1.1 General overview 1
1.2 Project goal 2
2 Variant models of BERT architecture 3
2.1 BERT: The original architecture 3
2.2 RoBERTa 3
2.3 DeBERTa 4
2.4 DistilBERT 5
2.5 ALBERT 5
3 Experiments 6
4 Results 8
4.1 Quora Question Pairs - QQP 8
4.2 Stanford Sentiment Treebank - SST-2 11
5 Discussions and conclusions 13
5.1 Discussions 13
5.2 Conclusions 14
6 Division of labor 14
7 Acknowledgments 15
8 Appendix 15
References 15

1 Introduction

1.1 General overview

The General Language Understanding Evaluation (GLUE) benchmark is a vital tool for assessing natural language understanding models across various tasks. Among these tasks, Quora Question Pairs (QQP) assesses semantic similarity between questions, crucial for applications like question-answering systems, while Stanford Sentiment Treebank (SST-2) focuses on sentiment analysis in sentences or documents. In this project, we will evaluate the performance of different state-of-the-art NLP models.

Traditional linguistic rule-based methods for such tasks were limited by manual crafting and maintenance of rules, while statistical approaches like support vector machines (SVMs) faced challenges with annotated training data. In recent years, deep learning models like BERT and its variants have excelled in GLUE tasks, leveraging pre-training on large text corpora and fine-tuning on specific data.

Transformer-based models, notably BERT, have revolutionized NLP due to their ability to capture contextual dependencies and semantic relationships. The architecture comprises tokenization, positional encoding, self-attention, feedforward, and cross-attention layers, enabling effective handling of long-range dependencies and variable-length sequences.

BERT and similar models outperform traditional methods due to their bidirectional contextual understanding, as demonstrated in tasks like QQP and SST-2. Their capacity to analyze entire sentences simultaneously enhances accuracy in semantic similarity and sentiment analysis.

Our project aims to evaluate Transformer-based models' performance, primarily BERT and some of its variations, on GLUE tasks like QQP and SST-2. The framework involves defining the problem, selecting models, devising evaluation metrics, and iteratively enhancing performance. Intuitively, we could compare the performance of these models against previously state-of-the-art approaches on the same GLUE tasks (Figure 1) such as Attention-select (an attention-based mechanism) from Goyal et al. (2020) [3] or Average-pool (a strided mean pooling applied to each sliding window) from Dai et al. (2020) [4].

Dataset	1st-I	1st	Rand	Pool	Att	CS-k-1	CS-opt	$\text{BERT}_{\text{Base}}$
STS-B	**87.9**	**87.9**	87.8	87.8	**87.9**	87.7	**87.9**	87.9
MRPC	86.8	86.4	87.2	87.0	87.1	86.9	**87.3**	87.3
SST-2	92.1	91.5	91.9	90.3	92.3	**92.4**	**92.4**	92.4
QNLI	90.8	90.8	90.8	90.2	90.7	**90.9**	**90.9**	90.9
COLA	53.0	52.7	53.1	25.6	53.2	**53.3**	**53.3**	53.3
RTE	65.6	65.2	**65.7**	61.5	**65.7**	65.4	65.6	65.8
MNLI_M	**84.0**	83.8	83.9	80.9	**84.0**	**84.0**	**84.0**	84.0
MNLI_MM	84.1	84.0	83.9	84.0	84.5	**84.6**	**84.6**	84.6
QQP	87.1	86.9	87.4	85.7	87.4	**87.5**	**87.5**	87.5
Mean	81.3	81.0	81.3	76.8	81.4	81.4	**81.5**	81.5

Figure 1. *Performance of some state-of-the-art models on GLUE [5]*

1.2 Project goal

The project is expected to serve several research purposes. First, comparing different variants of BERT allows researchers to gauge which model performs best on specific tasks. This evaluation can help in understanding the strengths, weaknesses, and behaviors of each model and guide future research efforts better. This comparison allows researchers to evaluate the effectiveness of the innovations in each variant, which might shed light on future hybrid variants that combine the strengths and mitigate the negative sides. This progress is essential for developing more accurate and reliable NLP models with practical applications

Furthermore, this can assist future practical applications and real-world integration. At the same time, it can also be beneficial for research on similar or more generalized tasks. Assessing how different variants of BERT generalize across tasks not only speeds up the assessment but also provides insights into the variants' robustness and adaptability, which can aid our decisions about model deployment in various scenarios.

Besides the References, this report includes 8 sections. The first one, this section, is an introduction to the project and our motivations. In section 2, we study the main innovations of BERT and each variant, especially their differences compared to their "siblings". Section 3 goes into detail the stages of our experiments. Next, section 4 discusses the results achieved on each task. Section 5 includes further discussions on the results attained in section 4. Section 6 specifies the division of labor between our team members. In section 7, we include an acknowledgment for our supervisor and professor, who assisted us during the completion of this project. Finally, section 8 contains the source code and a tutorial on how to replicate the fine-tuning results for all BERT models.

2 Variant models of BERT architecture

2.1 BERT: The original architecture

BERT (Bidirectional Encoder Representations from Transformers) [1] has emerged as a groundbreaking method in natural language processing (NLP) with its ability to capture contextual information bi-directionally. This means it considers both the left and right context of each word in a sentence, enabling a deeper understanding of language semantics. BERT achieves this through a transformer architecture, which utilizes attention mechanisms to weigh the importance of different words in a sentence when processing it.

We chose BERT for QQP and SST-2, tasks that require a complex semantic understanding, for several reasons. Firstly, QQP involves understanding the semantic similarity between pairs of questions, a task that requires capturing nuanced contextual information. The bi-directional approach of BERT enables it to grasp the intricate relationships between words and phrases within each question, leading to more accurate similarity judgments. Similarly, SST-2, a sentiment analysis task, benefits from BERT's advanced contextual understanding. Sentiment analysis requires grasping the sentiment expressed in a sentence, which often relies on understanding the context and subtle nuances. BERT's ability to capture contextual information helps it accurately discern the sentiment expressed in various contexts.

On top of that, BERT's pre-training on large corpora enables it to learn robust representations of language, which can be fine-tuned for specific tasks like QQP and SST-2 with relatively small amounts of task-specific data. This transfer learning approach saves time and computational resources while achieving state-of-the-art performance.

BERT has proved to be a compelling option for many NLP tasks with state-of-the-art performance [6]. However, in this report, we will also investigate and compare BERT's performance against some of its popular variations.

2.2 RoBERTa

RoBERTa (Robustly optimized BERT approach) [7] builds upon the BERT's foundation, refining its architecture and training methodology to achieve even better performance across various NLP tasks. One of the key differences lies in

RoBERTa's pre-training process, which involves significantly larger batch sizes and longer training duration in comparison to BERT. This extensive pre-training enables RoBERTa to capture more nuanced language representations and improves its generalization ability. Moreover, RoBERTa introduces dynamic masking during pre-training, where different masking patterns are applied to each training instance. This technique helps RoBERTa better utilize the available training data and enhances its ability to understand diverse linguistic patterns.

At the same time, RoBERTa employs byte pair encoding (BPE) tokenization with a larger vocabulary size, enabling it to effectively handle rare words and out-of-vocabulary tokens. This contributes to RoBERTa's robustness and enhances its performance on tasks with diverse vocabulary.

In terms of architecture, RoBERTa retains the transformer-based architecture of BERT but removes the next sentence prediction (NSP) task during pre-training. Instead, RoBERTa focuses solely on the masked language modeling task, allowing it to allocate more resources to learn contextual representations from the input.

2.3 DeBERTa

DeBERTa (Decoding-enhanced BERT with disentangled attention) [8] represents a novel approach in the realm of transformer-based NLP models. One of the key innovations of DeBERTa lies in its disentangled attention mechanism, which separates the attention weights into different dimensions, allowing the model to attend to different aspects of the input text independently. This mechanism enables DeBERTa to capture more complex linguistic dependencies and long-range contextual information, improving the performance on various NLP tasks. Additionally, DeBERTa introduces a decoding mechanism that facilitates more effective generation tasks by adjusting the attention mechanism during decoding.

Compared to BERT, RoBERTa, ALBERT, and DistilBERT, DeBERTa can capture more nuanced relationships between words and phrases in the input text, leading to superior performance on tasks that require a deep understanding of language semantics such as QQP and SST-2. Additionally, the decoding-enhanced mechanism enhances DeBERTa's ability to generate coherent and contextually appropriate responses in generation tasks. While the other methods have made significant contributions to the field of NLP, DeBERTa's innovations in attention mechanisms and decoding strategies set it apart, offering a promising avenue for further advancements in transformer-based models. We expect DeBERTa's ability

to capture complex linguistic relationships would make it a compelling choice excelling in the tasks at hand - QQT and SST-2.

2.4 DistilBERT

DistilBERT [9], as suggested by its name, focuses on distillation techniques to compress or "mimic" the knowledge from large models like BERT into a smaller, more efficient architecture. By distilling the knowledge from BERT, DistilBERT achieves a notable reduction in the number of parameters, making it more lightweight and resource-efficient. Furthermore, DistilBERT also utilizes a modified training objective that combines distillation with a novel form of self-distillation, where it distills knowledge from its intermediate layers during training. This self-distillation process encourages the model to focus on learning more generalizable representations, further enhancing its efficiency and performance. For these reasons, DistillBERT stands out compared to previous variations of BERT for its emphasis on compression and efficiency, making it suitable in resource-constrained environments.

2.5 ALBERT

Another variation of BERT that we investigate is ALBERT (A Lite BERT) [10] which aims to address some of the limitations of BERT while maintaining a high performance. One of ALBERT's key innovations is its parameter reduction techniques, which significantly reduce the number of parameters compared to BERT while preserving or even improving performance. This parameter reduction makes ALBERT more memory-efficient and faster to train and deploy, making it more accessible for various applications. Furthermore, ALBERT allows better parameter utilization by introducing cross-layer parameter sharing, where the parameters of different layers in the transformer architecture are shared to encourage more effective learning across layers, leading to improved performance.

Alongside masked language modeling, ALBERT incorporates sentence-order prediction (SOP) as an auxiliary task during pre-training, aiming to predict whether two sentences appear consecutively in the original document, encouraging the model to learn deeper contextual relationships between sentences.

Compared to BERT and RoBERTa, ALBERT's parameter reduction techniques result in a more compact model without sacrificing performance. This makes ALBERT particularly suitable for scenarios where computational resources are limited or where fast inference times are crucial.

3 Experiments

The goal of the experiment is to fine tune these existing pretrained BERT models to fit on the new datasets using the API <BERT model>ForSequenceClassification on HuggingFace. Only models that can support sequence classification can be used on QQP and SST2, such as BERT and XLNET. In contrast, GPT models are trained for text generation, and it may lack support for sequence classification.

Here are the stages of our experiment on the code implementation

- Stage 1 - Data uploading: first, we upload two datasets QQP and SST-2 onto Kaggle. The reason we run on Kaggle is because they have GPU support and we can leave the training model running overnight. We have access to High Performance Computing service, but since it is not a free open computing resource, we chose a common platform Kaggle that everyone has access to.
- Stage 2 - Data preprocessing: we split the data into training, validation, and test sets, with training and testing ratio of 80/20 for both QQP and SST-2. The validation dataset however comes from dev.tsv file. For QQP task, we reduce the dataset size due to memory and time constraints (subsetting the data to 20000 question pairs instead of 400000 pairs in the original train.tsv)
- Stage 3 - Choosing BERT variant model: Our code implementation recognizes that HuggingFace API has a convenient way of switching models and tokenizers using the two generic imports

```
# PLEASE CHOOSE THE BERT VARIANT MODEL
bert_model_name = "bert-base-uncased"
# bert_model_name = "roberta-base"
# bert_model_name = "distilbert-base-uncased"
# bert_model_name = "albert-base-v2"
# bert_model_name = "deberta-base"

from transformers import AutoTokenizer, AutoModelForSequenceClassification
bert_temp_name = "microsoft/deberta-base" if bert_model_name == "deberta-base
    " else bert_model_name
tokenizer = AutoTokenizer.from_pretrained(bert_temp_name)
```

```
bert_classification_model = AutoModelForSequenceClassification.
    from_pretrained(
pretrained_model_name_or_path=bert_temp_name,
    num_labels=2, # We choose 2 for binary classification for both QQP and
        SST-2 tasks
        # num_labels can be higher for many classes classification
    output_attentions=False, # returns attentions weights or not
    output_hidden_states=False, # returns all hidden-states or not)
bert_classification_model = bert_classification_model.to(device)
```

By structuring our code like this, it is extremely easy to switch among the BERT models without having to rewrite the entire code. There is a catch, that RoBERTa and DistilBERT do not have token type IDs, so latter functions to process these two BERT variants must exclude this feature field.

- Stage 4 - Tokenizer: since BERT models needs tokenization (much like any NLP tasks) instead of raw strings, this stage aims to use the specific tokenizer to tokenize sentences that goes with the BERT model. These tokenizers can add special tokens ([CLS], [SEP]) necessary for BERT, and also add padding and truncating sequences to a fixed length.

- Stage 5 - preparing dataloader: at this stage, we convert tokenized data into PyTorch datasets. Then we create DataLoader for handling batches of data efficiently during training. There are four dataloaders, one for training, one for validation, one for testing with known truth values and one for performing inference on the GLUE test.tsv that does not have truth labels.

- Stage 6 - Model fine-tuning: In this stage, we set up the optimizer (AdamW) and learning rate scheduler. Then, we define the training loop, which includes batch training with gradient calculation and model updates. Then, we perform validation to evaluate the model on unseen data.

- Stage 7 - Calculating performance metrics: We used the fine-tuned model to make predictions on the testing datasets. Particularly, we calculate performance metrics like accuracy, precision and recall, and we generate confusion matrices for all BERT models.

- Stage 8 - Result analysis: We saves training statistics and plots for further analysis, including both training time and average inference time.

4 Results

4.1 Quora Question Pairs - QQP

Due to the long training time and limited cap on GPU usage, only a subset of question pairs from this dataset are used to fine-tune the BERT models. Specifically, 16000 question pairs from train.tsv are used to train the model, 2000 question pairs from dev.tsv is for validating the model during training, and 4000 question pairs from train.tsv is for testing the model. The file test.tsv of GLUE benchmark task does not contain the true labels, so it is only used for making inferences, and its performance can only be measured by human annotation.

The 5 BERT models were fine-tuned for 4 epochs using the same sub-dataset above on GPU T4x2. In order words, they are benchmarked on the same settings and same computational resources, which are ideal for truthful comparison. Figure 2 below shows that the training accuracy monotonically increases (which is hardly surprising), but the validation accuracy is levelled out, suggesting that 4 epochs are enough for fine tuning the models.

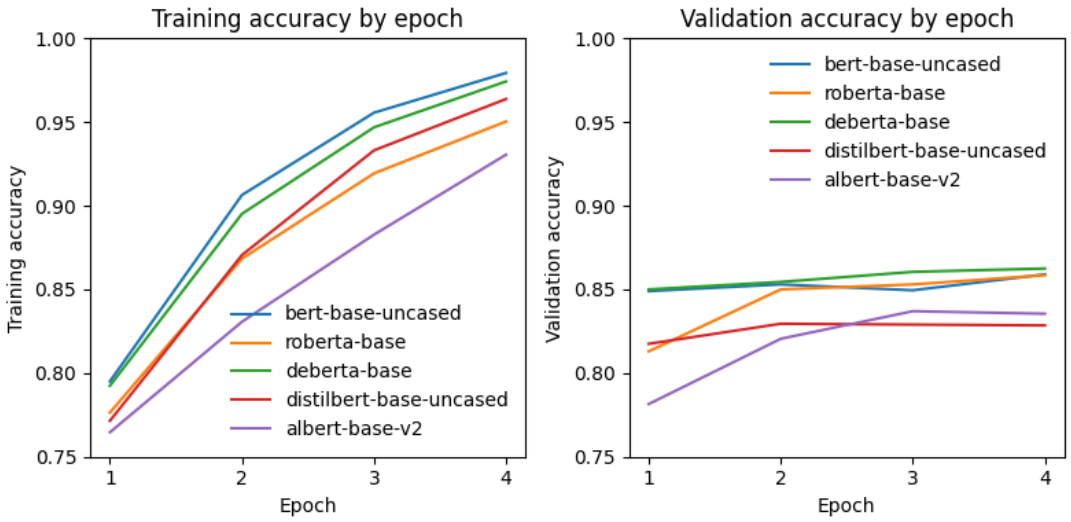


Figure 2. *Training and accuracy measured by 4 epochs for QQP dataset*

During training, we can obtain metrics such as training and inference time, training and validation accuracy and their respective loss. After training, we can obtain 4 metrics on the testing dataset, which are accuracy, precision, recall and F1. In this QQP task, we see that recall is higher than precision, which means BERT models are effective at identifying pairs of duplicate questions, even if it comes at higher cost of mistakenly judge two unrelated questions as similar (lower precision). All of the metrics are documented in Table 1 below.

Model	train_acc	val_acc	test_acc	test_precision	test_recall	test_F1
BERT	0.9794	0.8590	0.8460	0.7672	0.8302	0.7975
RoBERTa	0.9504	0.8585	0.8480	0.7667	0.8392	0.8013
DeBERTa	0.9743	0.8625	0.8528	0.7831	0.8255	0.8037
DistilBERT	0.9638	0.8285	0.8208	0.7218	0.8275	0.7710
ALBERT	0.9306	0.8355	0.8245	0.7410	0.7988	0.7688

Table 1. *Performance metrics of five BERT models on QQP*

Nonetheless, the table below is simply numerical results, and it is quite hard to compare which model generally performs the best. We can compare their performance much better in the radar plot below

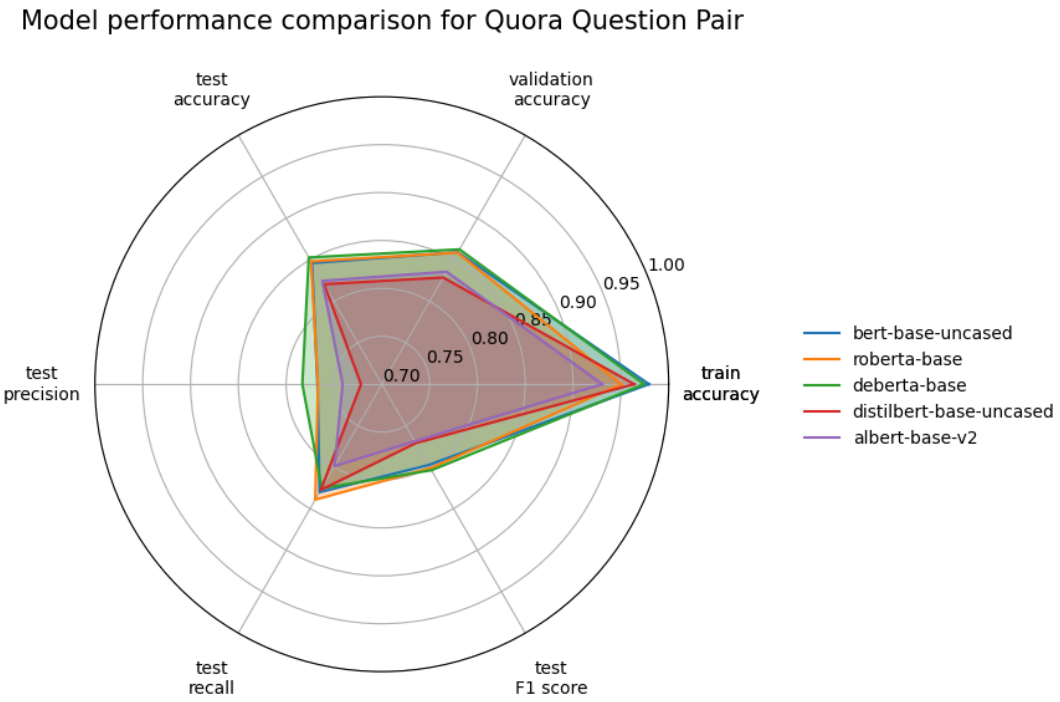
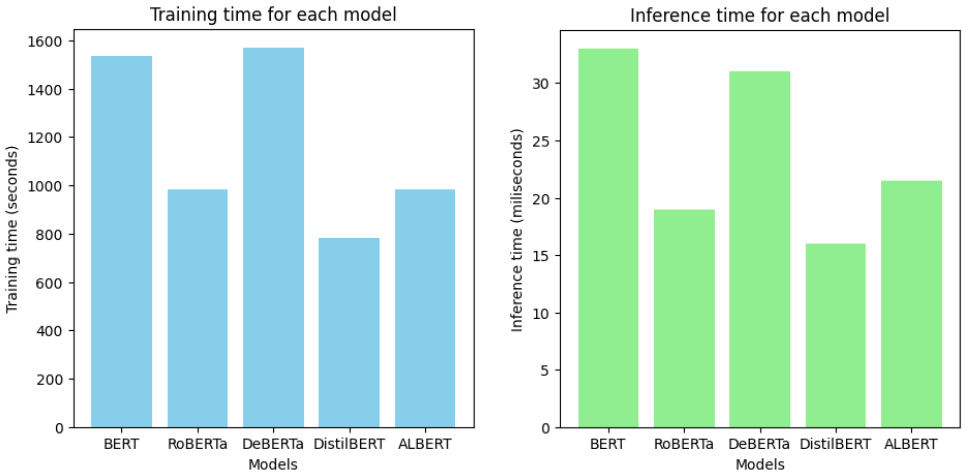


Figure 3. *Metric performance comparison between BERT models on QQP testing dataset*

If we exclude training accuracy, DeBERTa and RoBERTa model tends to perform best on the QQP sub-dataset, while ALBERT and DistilBERT performs equally worst. It is now clear that performance is correlated with the size of the pretrained models, as DeBERTa has the most parameters while ALBERT and DistilBERT have much fewer parameters. However, saying so does not justify the competitiveness of these two models since they are supposed to exchange model performance for less computational resources. As a result, their total training time and their inference time per question pair should be compared, which can be seen in figure 4



(a) **(b)**

Figure 4. *Training and inference time of five BERT models on QQP*

We can conclude that the base BERT model and DeBERTa has the highest training and inference time, while RoBERTa, DistilBERT and ALBERT have only about half training and inference time. Taking into account both performance metrics and computational resource judged by training and inference time, we conclude that the order of performance from best to worst for the QQP dataset is

(1) RoBERTa (2) DeBERTa (3) BERT base (4) DistilBERT (5) ALBERT

Finally, the confusion matrix shows the quantitative count of prediction for the best model - RoBERTa. The 0 label means not duplication question pairs and 1 label means duplicates. It has higher FPR than FNR.

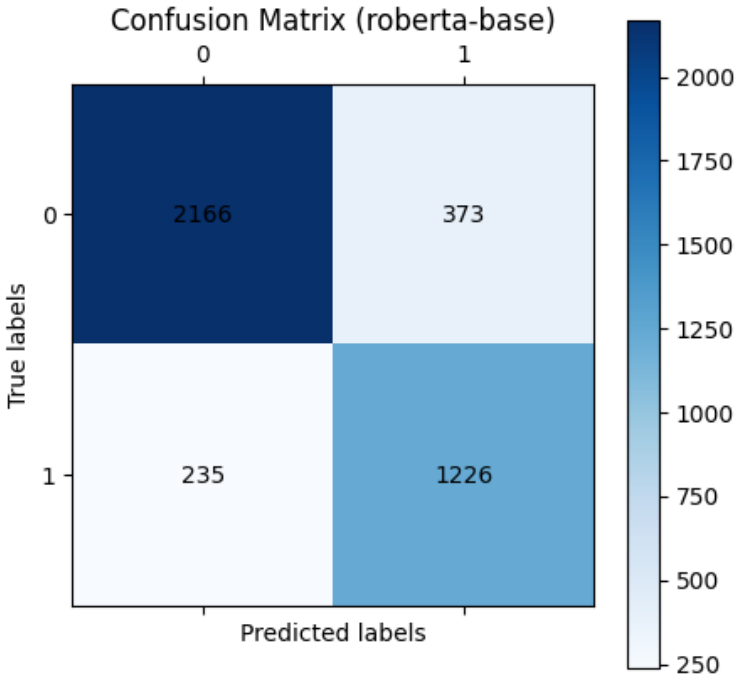


Figure 5. *Confusion matrix of RoBERTa on QQP testing dataset*

4.2 Stanford Sentiment Treebank - SST-2

In contrast to QQP, the sentence maximum tokens in SST-2 dataset is only 64, so the computational resource required to fine tune the BERT models is not as significant as the QQP case. Specifically, all sentence feedback from train.tsv are used to train the model with training and testing ratio of 80/20, and all sentence feedbacks from dev.tsv is for validating the models. Like QQP, the file test.tsv of GLUE benchmark task does not contain the true labels, so it is only used for making inferences, and its performance can only be measured by human.

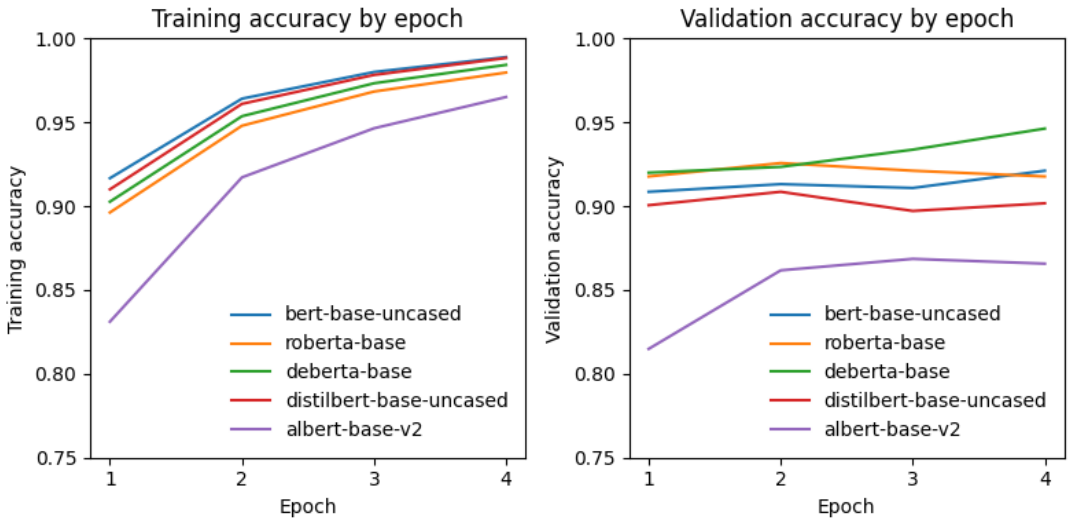


Figure 6. *Training and accuracy measured by 4 epochs for SST-2 dataset*

Like QQP, the performance of five models reaches their peak training accuracy after 4 epochs of fine-tuning, but their validation accuracy has reached a plateau, so no more training is required to prevent overfitting.

After training, table 2 contains the performance metrics of 5 models on SST-2. Generally speaking, SST-2 is a much easier task than QQP, since all models achieve superb results on all performance metrics.

Model	train_acc	val_acc	test_acc	test_precision	test_recall	test_F1
BERT	0.9889	0.9211	0.9514	0.9414	0.9482	0.9448
RoBERTa	0.9797	0.9177	0.9510	0.9455	0.9416	0.9435
DeBERTa	0.9843	0.9463	0.9491	0.9411	0.9431	0.9421
DistilBERT	0.9884	0.9017	0.9476	0.9403	0.9403	0.9403
ALBERT	0.9651	0.8657	0.9333	0.9146	0.9354	0.9249

Table 2. *Performance metrics of five BERT models on SST-2*

Again, we can visualize their performance on SST-2 in the radar plot below

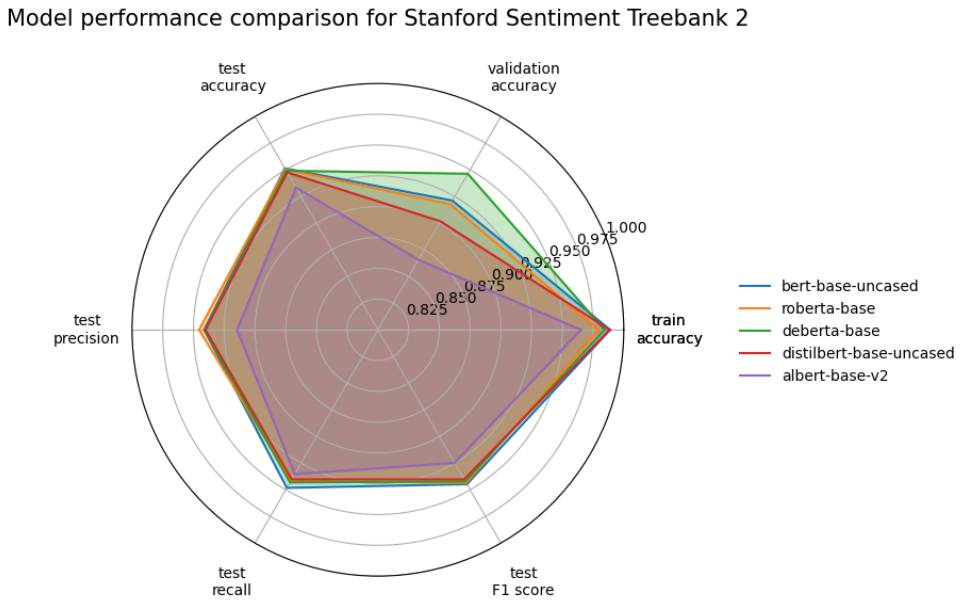
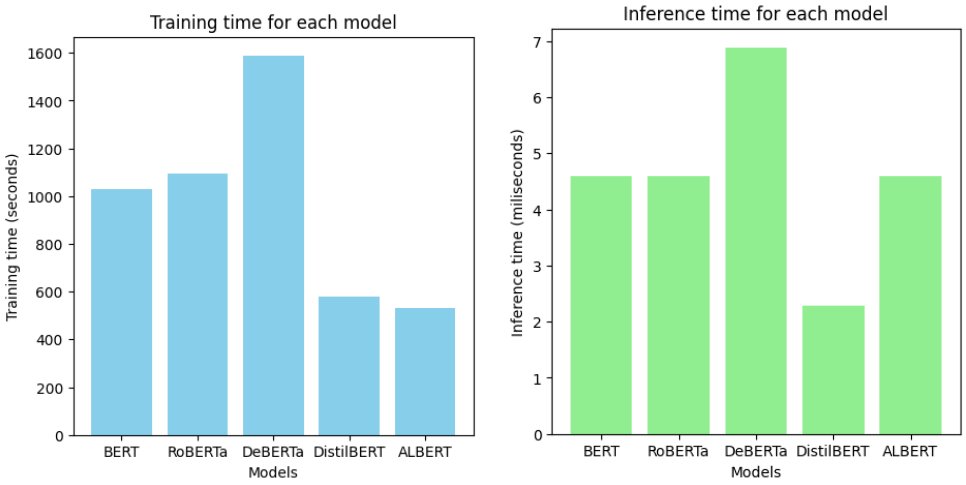


Figure 7. *Metric performance comparison between BERT models on SST-2 testing dataset*

In SST-2, it is quite hard to tell which model performs the best judging solely by testing metrics. Relying on validation accuracy is also not reliable since the validation dataset is very small and there could be lots of variation. Based on testing accuracy, BERT base model and RoBERTa perform equally good. From figure 8, they also have the same training and inference time. Since performance metrics cannot tell which is the best model, we can rely on the training and inference time to judge their computational efficiency.



(a) **(b)**

Figure 8. *Training and inference time of five BERT models on SST-2*

We conclude that the order of performance from best to worst for the SST-2 dataset is as follows

(1) DistilBERT (2)(3) BERT and RoBERTa (4) DeBERTa (5) ALBERT

Since RoBERTa performs well on both datasets, for consistency, we can proceed to plot the confusion matrix of fine-tuned RoBERTa on the SST-2 testing dataset. It appears now that RoBERTa have FPR than FNR just like the QQP task.

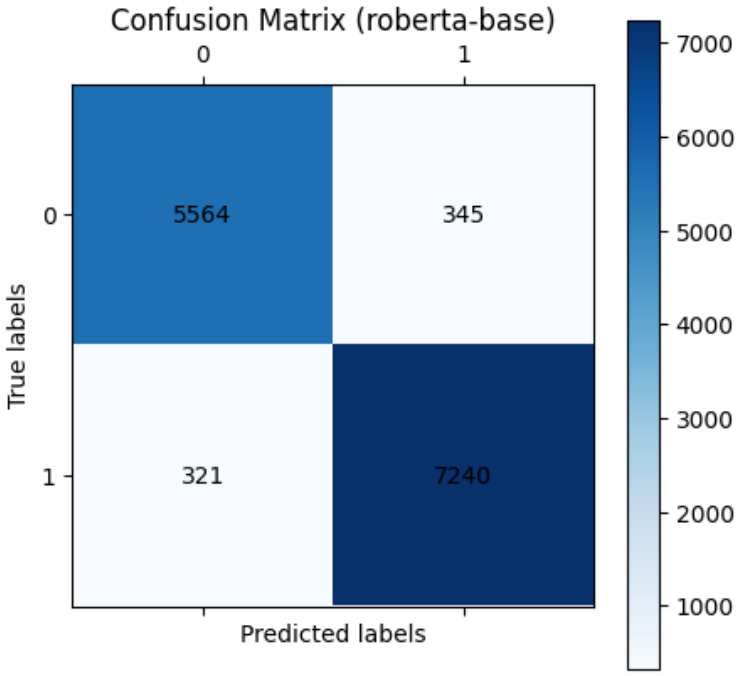


Figure 9. *Confusion matrix of RoBERTa on SST-2 testing dataset*

5 Discussions and conclusions

5.1 Discussions

In this project, we evaluated the performance of 5 variations of BERT (BERT, RoBERTa, ALBERT, DistilBERT, and DeBERTa) on 2 natural language tasks QQP and SST-2. These are 2 of the most popular benchmarks to evaluate the performance of NLP models. However, due to resource constraints, we only use a subset of the data for training and validation. Nonetheless, the results should be close to those attained using the full dataset.

We conducted a comprehensive comparison between the models using multiple evaluation metrics, such as accuracy, precision, recall, and F1-score. Overall, all BERT models perform extremely well on both QQP and SST-2 tasks, achieving remarkable results compared to state-of-the-art accuracy 1. It can also be observed that the results for SST-2 are relatively higher than those for QQP. This is reasonable as SST-2 is a simplified version of the QQP task.

Between variations, our observations concluded that there is no significant difference in performance. However, we can see that BERT, RoBERTa, and DeBERTa achieved relatively better statistics compared to ALBERT and DistilBERT in both tasks. While DeBERTa has a slightly higher performance for QQP, BERT and RoBERTa seem to surpass it in SST-2. Furthermore, BERT and RoBERTa takes much less time to train on both tasks compared to DeBERTa. On the other hand, the difference in training time is quite noticeable with DistillBERT when these it consumes much less time to train on both tasks. For ALBERT, the training time is significantly improved compared to other variants in QQP, but there is no improvement in SST-2. From these observations, RoBERTa gives the most consistent and stable performance, while ALBERT fails to outshine its cousins.

5.2 Conclusions

These results are consistent with the innovations that we discussed in Section 2. The improvement in speed and efficiency of DistilBERT can be explained by its main mechanism which tries to mimic the knowledge of BERT into a smaller and compact model, exchanging some performance for efficiency. Similarly, ALBERT sacrifices performance for a smaller model with fewer parameters, which results in a faster model but less stable and robust. On the other hand, RoBERTa outperforms other variants due to its novel pre-training process and dynamic masking, both enable RoBERTa to capture a more nuanced representation of language to understand more complex linguistic patterns.

Future work might include running the project on the whole dataset to have more comprehensive results. We expect all models to perform well on the full data, but it might be clearer which model outperforms the rest. The framework of this project is also solidified for future extension, where it is possible to add new variations of BERT or other models into the evaluation.

6 Division of labor

Nguyen Xuan Binh: Responsible for the whole code implementation and five BERT models training on Kaggle. He also documented the section Experiment, Result and proof-read other sections.

Nguyen Long: Responsible for writing majority of the report, such as introduction, literature review, model architecture review, discussion and conclusion.

7 Acknowledgments

We would love to extend our thanks to professor Mikko Kurimo and Mr. Nhan Phan for helping us to understand clearly project requirements.

8 Appendix

All of our project codes are hosted on Github at this repository. They should be run entirely on Kaggle to make use of GPU, since training with CPU is extremely slow

https://github.com/SpringNuance/SNLP-Project

The tutorial on how to run the notebooks on Kaggle is documented here

https://github.com/SpringNuance/SNLP-Project/blob/main/SNLP_tutorial.pdf

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[2] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," 2019.

[3] S. Goyal, A. R. Choudhury, S. Raje, V. Chakaravarthy, Y. Sabharwal, and A. Verma, "PoWER-BERT: Accelerating BERT inference via progressive word-vector elimination," in *Proceedings of the 37th International Conference on Machine Learning* (H. D. III and A. Singh, eds.), vol. 119 of *Proceedings of Machine Learning Research*, pp. 3690–3699, PMLR, 13–18 Jul 2020.

[4] Z. Dai, G. Lai, Y. Yang, and Q. V. Le, "Funnel-transformer: Filtering out sequential redundancy for efficient language processing," 2020.

[5] X. Huang, A. Khetan, R. Bidart, and Z. Karnin, "Pyramid-bert: Reducing complexity via successive core-set based token selection," 2022.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[7] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[8] P. He, X. Liu, J. Gao, and W. Chen, "Deberta: Decoding-enhanced bert with disentangled attention," 2021.

[9] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," 2020.

[10] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," 2020.