# Comparative Study of Transformer-Based Large Language Models Final Report 05/15/2023

**Man Liang, Charlie Lu, Sky Cen**
University of Maryland
{liang558, clue, scen1}@umd.edu

## Abstract

This project proposal aims to review the spectrum of uses for Large Language Models, or LLMs, in Transformer models. Since LLMs have achieved great success in the past few years, including the recent ChatGPT model, it is interesting to see how different transformer-based models perform on different tasks, for example, Named Entity Recognition (NER), Natural Language Inference (NLI), and Question Answering (QA).

After a literature review process, this project will compare the performance of widely used transformer-based deep learning architectures on benchmark datasets in natural language processing.

## 1 Introduction

Transformer-based models have become very popular in large language models and have achieved state-of-the-art performance on a variety of NLP tasks. The recent popular ChatGPT is a famous example. Transformer-based models were introduced in 2017 by Vaswani et al. The key innovation of them is the attention mechanism, which allows models to selectively focus on different parts of the input sequence while processing. Many transformer-based architectures are designed on various tasks, such as GPT, BERT, DistilBERT, XLNet, RoBERTa, etc. However, it is currently unknown how these models would perform on different tasks. It is a public perception that the GPT model is doing great in Question-Answering Tasks, but we don't know if or to what extent it will outperform the others. Motivated by this observation, we set the research objective as understanding the performance of backbone architectures on common NLP tasks via literature survey and finetuning experiments. our research is designed with three major parts. First of all, we are going to conduct a literature survey of 5 widely used transformer-based architectures to get an overall understanding of these models and the current research trend. Second, we will finetune the models with benchmark datasets for three common NLP tasks: NER, NLI, and QA. Third, we will analyze and compare the performance of each model and its transferability. Our result would be compared with the findings from the literature survey to suggest the model selection.

## 2 Literature Review

Recently, there have been substantial works to understand and improve transformer-based large language models. Clark, et al [1], studied the knowledge captured via the attention mechanisms of BERT. Hendrycks, et al [2] investigated out-of-distribution (OOD) generalization to improve model robustness. To get a holistic understanding of transformer-based models, several surveys were conducted. Kalyan, et al [3] and Min, et al [4] summarized the taxonomy and recent advances in this

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

field. However, the survey was conducted at a high level in getting the big picture of the research field. No experiments were conducted to perform comparisons between different models for common NLP tasks. Therefore, there is a lack of literature to suggest model selection for solving practical problems.

### 2.1 Transformer Model

The transformer (Vaswani et al., 2017) model utilizes the attention mechanism as its core, in contrast to recurrent neural networks. The attention mechanism enables the model to focus on specific aspects of the text by learning the varying levels of importance of different text elements for a given task. For instance, words like "shopping" or "offer" may be highly relevant for filtering spam emails but not as important when analyzing sentiment. The attention mechanism determines this relevance by calculating a dynamically learned score for each element of the input sequence (value) and then computing the relative importance of the input sequence (key) for an output (query). Formally, attention maps a set of query, key, and value elements to an output, which is obtained by summing the weighted values, where the weights are determined by the compatibility (via dot product) between the query and the corresponding key.

### 2.2 Architectures

- BERT
  BERT (Bidirectional Encoder Representations from Transformers) is a transformer-based language model that uses a pre-training technique called Masked Language Modeling (MLM). BERT is primarily used for tasks such as question answering, text classification, and text similarity.
- Distil-BERT
  Distil-BERT is a smaller and faster version of BERT that has been distilled to reduce its size and computational requirements. Despite the smaller size, Distil-BERT is still able to compete with BERT.
- XLNet
  XLNet is similar to other transformer-based models, such as BERT and RoBERTa, but it introduces several improvements that make it more powerful and flexible.
- RoBERTa
  RoBERTa (Robustly Optimized BERT pre-training approach) is a transformer-based language model that is similar to BERT but has been optimized for improved performance on various natural language processing tasks.
- ALBERT
  ALBERT (A Lite BERT) is a variant of the BERT (Bidirectional Encoder Representations from Transformers) model, which is a popular language representation model. ALBERT aims to reduce the computational resources required for training BERT by implementing parameter-sharing techniques, resulting in a smaller and more efficient model with comparable performance to BERT on various natural language processing tasks.

### 2.3 Tasks and Benchmark Datasets

As we delved into natural language processing (NLP) research, we examined several widely recognized benchmark datasets and their availability and reached the scope of our work below.

- NER
  Named Entity Recognition (NER) is a common task in NLP that involves identifying and classifying named entities in text, such as people, organizations, locations, and dates. The goal of NER is to automatically extract structured information from unstructured text. We are planning to use the WikiNEuRal EN dataset to perform model finetuning on NER.
  - CoNLL-2003
    CoNLL-2003 is a popular dataset used in natural language processing (NLP) that contains labeled named entity recognition (NER) annotations for English language text, commonly used for training and evaluating NER models.

- NLI
  Natural Language Inference (NLI) is another common task in NLP that determines the relationship between two text sequences. The goal is to infer and predict the relationship between a hypothesis and a given premise. We are planning to use the GLUE dataset for NLI task.
  - GLUE
    GLUE benchmark dataset: A collection of diverse NLP tasks used for evaluating language models' performance and generalization in one sentence. It's also available on HuggingFace Datasets. We are focusing on the NLI tasks in GLUE for this project.
- QA
  Question answering (QA) is a task in natural language processing (NLP) that involves answering questions. It's aiming to provide a machine-readable answer to a human-generated question, given a large corpus of text as input. Recently released ChatGPT by OpenAI is an example. We will use SQuAD to finetune the architectures and evaluate them.
  - SQuAD
    SQuAD is a question answering dataset that combines answerable questions from SQuAD1.1 with unanswerable questions designed to resemble answerable ones, challenging models to abstain from answering when no supported answer is available. It's accessible on HuggingFace Datasets.

# 3 Literature Survey

## 3.1 Survey Methods

The literature survey acts as an important role to make our project successful. The objective of the literature survey is to get an overall understanding of the performance of the widely used transformer-based architectures in NLP tasks. In order to implement this approach, we conducted a literature review to identify papers that reported the performance of the 5 selected architectures on the benchmark dataset of interest. The result is summarized in the next section.

## 3.2 Survey Result

We have found the performance of target models on benchmark data from numerous different sources and have compiled the result into Table 1. In this table, we got the general sense that XLNet is good for NER, ALBERT is the best for both NLI and QA.

| | NER(CoNLL 2003) Accuracy | NLI(GLUE MNLI) Matched | QA(SQuAD-1.1) EM | F1 |
|---|---|---|---|---|
| BERT finetune baseline | 92.4% [5] | | 79.71 | 87.51 |
| DistilBERT | | | 75.34 | 84.13 |
| RoBERTa | | 90.8% | 85.42 | 91.82 |
| XLNet | **93.28%** [6] | | **89.898** | **95.080** |
| ALBERT | | **91.3%** | 82.69 | 90.25 |

Table 1: Summarize of Backbone Model on Benchmark Data

From the literature survey, our observations are

- Missing info for specific models on benchmark datasets within our consideration, which we'd like to experiment and fill in.
- All models tend to perform the best on NER, least on QA
- We don't know if there would be any model beat all in all tasks, which we will explore.

# 4 Experiment Setup

In this study, we are aiming to finetune 5 backbone transformer-based architectures on 3 selected tasks with benchmark datasets. We have replaced GPT architecture with ALBERT because limited

literature was found for the former one. The other four architectures are BERT, Distil-BERT, XLNet, and RoBERTa. We are going to finetune the models according to the three common tasks, which are NER, NLI, and QA. A summary of our scope is in Table 2. For NER and NLI, we will use accuracy and F1 score to evaluate how good our models are and for QA, we will use EM(Exact Match) and F1 score. Accuracy is defined as the number of correct predictions divided by the total number of predictions. F1 score is another measure of accuracy. F1 score is calculated using the harmonic mean of precision and recall. This allows the F1 score to be a better measure of accuracy when there are class imbalances in the dataset. EM is an evaluation metric measuring the proportion of documents with the predicted answer being exactly the same as the correct answer.

| Tasks | Benchmark Dataset | Evaluation Metrics |
|---|---|---|
| NER | CoNll 2003 | Acc, F1 |
| NLI | GLUE MNLI | Acc |
| QA | SQuAD | EM, F1 |

Table 2: Task Scope

### 4.1 Code Setup and Test

We have set up the basic codes for three selected tasks using the Transformer library with HuggingFace. To test the codes, we sampled a small subset from the benchmark datasets to make sure it works. The generic steps in the codes include:

- import pre-trained model from HuggingFace Transformer library
- load benchmark datasets from HuggingFace Datasets library
- preprocessing the dataset: tokenization, padding, collation, etc.
- using HuggingFace Trainer API to train the model
- prepare the evaluation metrics
- evaluate and save the models

A link to the codes is provided:
https://github.com/censky2/CMSC742

As we are going to run 5 backbone models, we investigated and listed the pretrained models or model checkpoints below. As we planned, these pretrained model will be plugged into our code respectively and test, together with appropriate tokenizers and paddings.

- BERT checkpoint: "bert-base-cased"
- DistilBert checkpoint: "distilbert-base-uncased"
- XLNet checkpoint: "xlnet-base-cased"
- RoBERTa checkpoint: "roberta-base"
- ALBERT checkpoint: "albert-base-v2"

### 4.2 What we did

- Full Model Training
  Our plan involves training a total of 15 tasks for each selected Language Model (LLM) model. This comprehensive training will include three different tasks for each individual model, resulting in a diverse and extensive training approach. We will start with the codes setup and replace with each backbone models. Given the magnitude of the task, we are considering the option of sampling a subset from the benchmark dataset, rather than training the entire set, in order to streamline the training process.
- Comparative Analysis
  From the model finetuning and evaluation section, we expected to get the performance of 5 pre-trained architectures and their finetuned version on 3 tasks. We are going to compare, analyze, and explain the performance of each backbone architecture and transferability. Our

results would also be compared with the findings obtained from the literature survey section to capture similarities and differences. For example, GPT architecture is perceived as good at QA tasks, but our experiment result may suggest other architectures on a specific dataset.

# 5 Results and Discussion

| | NER(CoNLL 2003) ACC | F1 | NLI(GLUE MNLI) ACC | QA(SQuAD) EM | F1 |
|---|---|---|---|---|---|
| BERT | 98.57% | 94.44 % | 82.42% | 78.70 | 86.67 |
| DistilBERT | 98.58% | 93.84% | 80.68% | 72.89 | 82.24 |
| RoBERTa | 99.07% | 95.93% | **87.4%** | 78.34 | 89.795 |
| XLNet | **99.11%** | **96.02%** | 86.02% | 80.52 | 90.68 |
| ALBERT | 98.46% | 93.00% | 83.5% | **83.28** | **90.53** |

Table 3: Experiment Results

| | NER(CoNLL 2003) | NLI(GLUE MNLI) | QA(SQuAD) |
|---|---|---|---|
| Baseline | XLNet | ALBERT | XLNET |
| Experiment | XLNet | XLNet | ALBERT |

Table 4: Best Models

The experimental results are presented in Table 3. Upon comparing the performance across three benchmark tasks, notable observations can be made. The NER (Named Entity Recognition) task exhibits exceptional accuracy, with all models achieving over 98%. However, for the NLI (Natural Language Inference) task, accuracies range from 80% to 88%, while certain QA tasks experience a decline, with EM rates falling below 80%. Intuitively, the task difficulty follows an ascending order of NER, NLI, and QA, which aligns with our findings and supports the observations documented in the literature survey. Furthermore, Table 4 showcases our experiment results and provides distinct insights for model selection in the NLI and QA domains. We discovered that architectures with a higher number of hyperparameters tend to favor the NLI task. Conversely, the ALBERT architecture, which possesses a relatively lower number of hyperparameters, demonstrates greater benefits for the QA task. These findings highlight the importance of considering different architectural characteristics and hyperparameter configurations based on the specific task requirements.

## 5.1 NER task observation

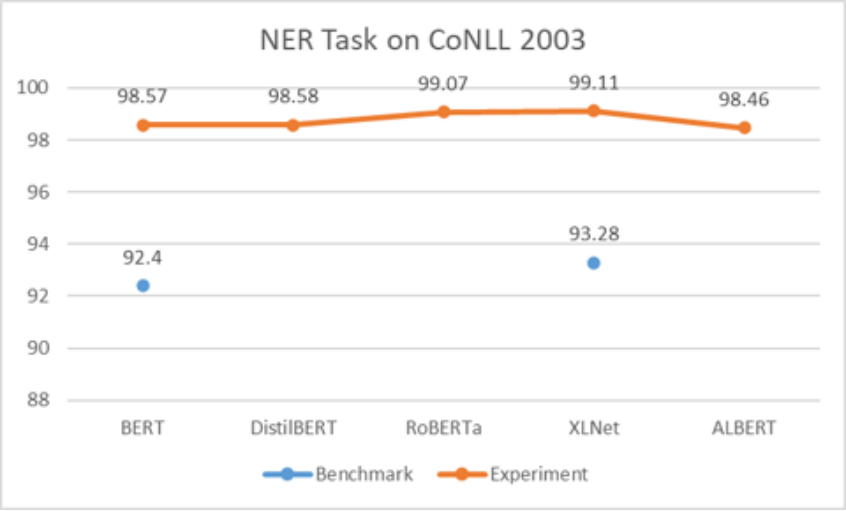


Figure 1: NER Task: Accuracy comparative to baseline

The 5 models trained for the NER task were all trained using 10 epochs, a weight decay of 0.01 and learning rates of $2 \times 10^{-5}$. All 5 models performed better than the benchmarks as our models had accuracies around $97 - 98\%$ whereas the benchmark accuracies were around $92\%$. However, it is important to note that those benchmarks were from 2018 and 2021. Current, more modern models are able to perform the NER task with accuracies up to $99\%$.
Additionally, BERT, DistilBERT, and RoBERTa were trained with the same parameters except for a varied learning rate of $2 \times 10^{-4}$. As expected, as learning rate decreased, the models tended to perform better. Interestingly, DistilBERT performed just as well as its nondistiled counterpart, BERT, despite having around 40% less parameters.

| | Learning Rate = $2 \times 10^{-4}$ | | Learning Rate = $2 \times 10^{-5}$ | |
|---|---|---|---|---|
| | ACC | F1 | ACC | F1 |
| BERT | 97.54% | 89.73% | 98.57% | 94.44% |
| DistilBERT | 97.57% | 90.40% | 98.58% | 93.84% |
| RoBERTa | 98.11% | 91.60% | 99.07% | 95.93% |

Table 5: NER Experiment Results

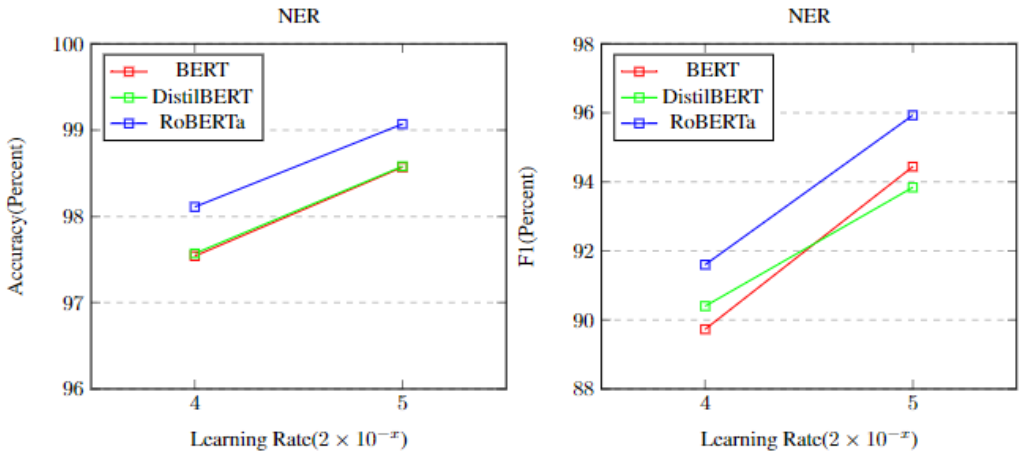


Figure 2: NER task experiment results regarding learning rate

## 5.2 NLI task observation

For the NLI task, we set the hyperparameters as the following: learning rate = $2 \times 10^{-5}$, number of epochs = 2, weight decay = 0.01. The comparative result over the baseline is provided in Figure 3. Surprisingly, our findings contradict the baseline results, where the ALBERT architecture performed the best. In our specific experiment, the RoBERTa model outperformed all others, with XLNet following closely behind. We speculate that this discrepancy arises from the substantial number of parameters in the RoBERTa model (125 million), which evidently provides significant advantages for the NLI task. Conversely, the embedding factorization and weight-sharing features inherent to the ALBERT architecture did not manifest effectively in our experiment.

## 5.3 QA task observation

The hyperparameters selected for the QA (Question Answering) task are as follows: learning rate = $2 \times 10^{-5}$, number of epochs = 1, weight decay = 0.01. The comparative results are depicted in Figure 4, revealing that our experiment's performance closely aligns with the baselines, albeit with a slight underperformance. Notably, in our QA experiment on the SQuAD dataset, the ALBERT model emerges as the top-performing model, surpassing the baseline with an impressive exact match (EM) score of 83.28. Despite having only 12 million parameters, ALBERT's embedding factorization and weight sharing features proved to be advantageous for the QA task, a characteristic that was

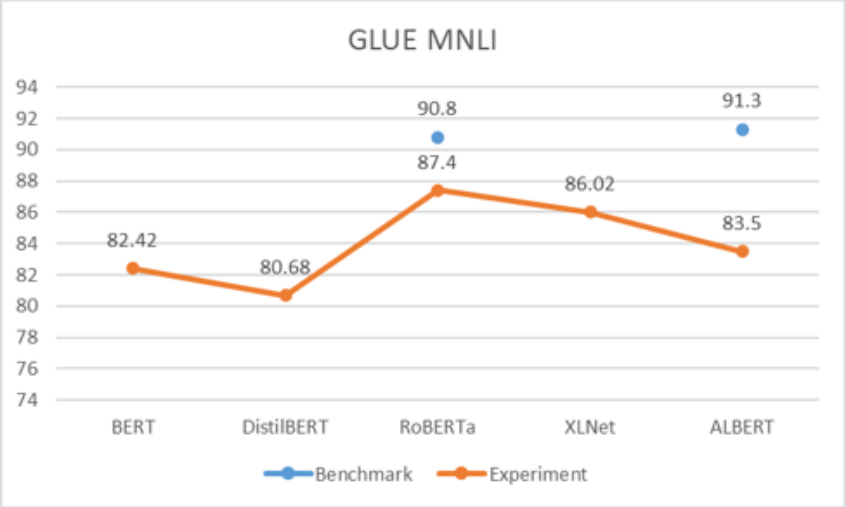


Figure 3: NLI Task: Accuracy comparative to baseline

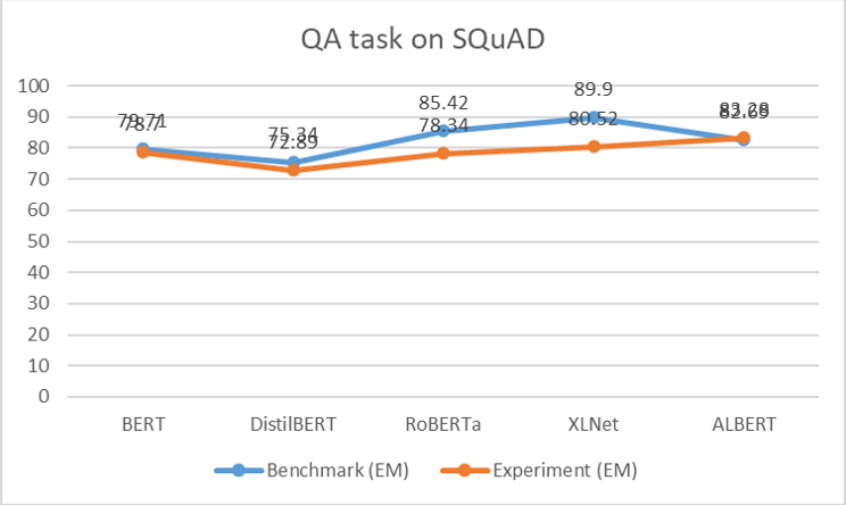


Figure 4: QA task: Accuracy comparative to baseline

not as impactful in the NLI task. In contrast, we deduce that a greater number of hyperparameters tends to benefit NLI tasks, while the embedding factorization and weight sharing features are more advantageous for QA tasks.

## 6 Conclusion

In this project, we conducted a literature survey to understand transformer models architecture and performance and empirically finetuned 15 models on 5 architectures over 3 benchmark datasets. What we found was that the results differ from benchmark models and believe this provides us with new model selection insights. We found that, for model performance across 3 tasks over the benchmark datasets, the models performed NER the most accurately and QA the least. XLNet was also performed the best on NER and NLI and ALBERT performed the best on QA. We also found that our observed accuracies show differences compared to baselines and hypothesize that higher number of parameters favor NLI tasks whereas embedding factorization and weight sharing favor QA tasks, which is why ALBERT outperforms our XLNet at QA tasks

# 7 Future works

It is important to acknowledge the limitations of our study. We must recognize that due to constraints in knowledge and computational resources at the time of conducting the work, we were unable to explore the complete landscape of hyperparameters and generate comprehensive results. Also, each of us is in charge of one task and the implementation details may be slightly different. which requires further validation of the results. We list the future works to improve our experiments as next steps.

- Investigate reasons for the gap between our results and the baseline
- Include other popular transformer models such as GPT-4
- Test and compare other hyper-params and metrics such as epochs, weight decays, to draw more general insights on model selection
- Employ ensemble learning to combine the models for each task
- Explore multi-task learning to train multi-head for all tasks

# References


[1] Clark, K., Khandelwal, U., Levy, O. and Manning, C.D., 2019. What does bert look at? an analysis of bert's attention. arXiv preprint arXiv:1906.04341.

[2] Hendrycks, D., Liu, X., Wallace, E., Dziedzic, A., Krishnan, R. and Song, D., 2020. Pretrained transformers improve out-of-distribution robustness. arXiv preprint arXiv:2004.06100.

[3] Kalyan, K.S., Rajasekharan, A. and Sangeetha, S., 2021. Ammus: A survey of transformer-based pretrained models in natural language processing. arXiv preprint arXiv:2108.05542.

[4] Min, B., Ross, H., Sulem, E., Veyseh, A.P.B., Nguyen, T.H., Sainz, O., Agirre, E., Heinz, I. and Roth, D., 2021. Recent advances in natural language processing via large pre-trained language models: A survey. arXiv preprint arXiv:2111.01243.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, volume abs/1810.04805, 2018. URL: `http://arxiv.org/abs/1810.04805`.

[6] Tran Thi Hong Hanh, Antoine Doucet, Nicolas Sidere, José G. Moreno, and Senja Pollak. Named entity recognition architecture combining contextual and global features. *CoRR*, volume abs/2112.08033, 2021. URL: `https://arxiv.org/abs/2112.08033`.

[7] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, volume abs/1910.01108, 2019. URL: `http://arxiv.org/abs/1910.01108`. eprinttype = arXiv, eprint = 1910.01108, timestamp = Tue, 02 Jun 2020 12:48:59 +0200. BibURL: `https://dblp.org/rec/journals/corr/abs-1910-01108.bib`. BibSource: dblp computer science bibliography, `https://dblp.org`.