Enhancing Precision in Question Answering Systems through a Dual-Model Approach

Gray Selby, Abraham Yang, Eunice Ngai

Abstract

Our long-term goal is to build a dual-model Question Answering system (Natural Language Inference (NLI) and text span extraction) that is efficient and precise, in order to serve use-cases demanding high precision and where latency is less of a concern. Within the context of an extractive question-answering system, this research focused on a classifier fine-tuned on General Language Understanding Evaluation (GLUE) Question-answering Natural Language Inference (QNLI) for the task of Natural Language Inference (entailment), serving as a screening layer to parse documents and identify potential answers to the question posed (Wang, Singh, Michael, Hill, Levy, and Bowman, 2019). Experiments were conducted and analyzed to determine a recipe for efficiently fine-tuning resulting in a pre-trained ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) model fine-tuned utilizing Low-rank Adaptation (LoRA) and half precision quantization. The experiments showed that this recipe is nearly three times more compute efficient to fine-tune than the baseline (pre-trained BERT fully fine-tuned). We also achieved 95% precision through optimization and threshold analysis. Further work will be pursued to implement the text span extraction task of our dual-model QA system, using ELECTRA fine-tuned on the SQuAD dataset.

Introduction

With the recent proliferation of generative AI (GenAI), institutions are looking to adopt GenAI solutions to reduce the cost of doing business. Our work focuses on the extractive Question Answering task from open-source solutions with an emphasis on cost efficiency and precision. More specifically, the scope of this project revolves around NLI, gearing towards building a more efficient and precise QA system through future work.

Within this context, many current QA systems consist of a single model that performs the tasks of parsing/understanding the question, finding/understanding context, and identifying and/or generating an answer (or the lack of an answer). Although such a system is efficient, effectively weight-sharing within a single model, it often struggles with precision and context relevance (Ishwari, Aneeze, Sudheesan, Karunaratne, Nugaliyadde, and Mallawarrachchi, 2019). Our research proposes a dual-model approach to extractive Question Answering whereby an NLI model effectively screens context for plausible answers to a question, followed by a second model focused on extractive, in-context QA to extract answer spans from contexts filtered by the NLI model. This dual-model QA system aims to enhance the precision of the QA system, where information precision is critical for business needs.

To this end, fine-tuning experiments are conducted to determine a recipe for efficiently training the proposed NLI classifier. Multiple models utilizing fine-tuning techniques are then analyzed to understand their shortcomings and to maximize precision.

Background

To address the challenges of question understanding, context identification, and answer generation, we look to NLI which in a broad sense refers to “machines’ capability of deep understanding of language that goes beyond what is explicitly expressed, rather relying on new conclusions inferred from knowledge about how the world works” (Storks, Gao, and Chai, 2020). In a more focused sense, NLI refers to the textual entailment task of whether a hypothesis is entailed by a premise. NLI uses an attention mechanism to focus on important parts of the hypothesis and premise, extract features, and compare those features.

In our search for efficiency, we discovered the ELECTRA model, which improves upon the BERT model by using a pre-training task known as Replaced Token Detection (RTD) (Clark, Luong, Le, and Manning, 2020). RTD allows ELECTRA to discern between closely related token alternatives. Unlike BERT’s Masked Language Model, which only learns from masked tokens, RTD improves learning efficiency by training the discriminator model (the main model) to predict whether any, all, or none of the tokens in the input sequence, replaced by a generator network (discarded after pre-training) with plausible alternatives, were the original tokens or not. This method allows ELECTRA to learn from all input tokens, which is where its efficiency lies. The original paper shows that the pre-training of ELECTRA is more efficient than other models by comparing the FLOPS. ELECTRA has also demonstrated improved performance on the GLUE datasets compared to BERT, RoBERTa, and XLNet (Clark, Luong, Le, and Manning, 2020). The ELECTRA model therefore fits well into our goals of improving efficiency and performance.

Methods

We investigated methods of efficient fine-tuning for our NLI model to present a recipe to maximize the efficiency of setting up this model by optimizing for the best performance with the least resources (money) used to train.

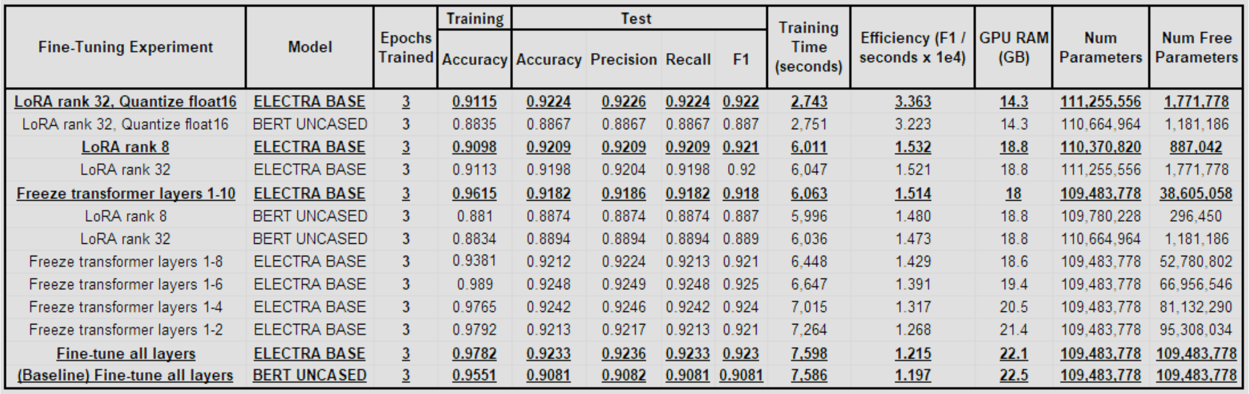
We started by replicating the GLUE benchmark results obtained by the developers of ELECTRA and briefly investigated transfer learning, fine-tuning on the QNLI dataset, and evaluating on the other tasks (see Appendix). Our results generally matched those reported by the ELECTRA authors. We compared BERT, ELECTRA and ALBERT by fine-tuning and evaluating on the GLUE QNLI dataset and found ELECTRA performed best followed by BERT.

Multiple fine-tuning approaches were conducted and analyzed with the objective of determining the most efficient fine-tuning recipe. These methods included fine-tuning all parameters, freezing select initial transformer layers, Low-Rank Adaptation (LoRA) and half precision quantization.

All models were fine-tuned on A100 GPU on Google Colaboratory because we believed that A100 and Google Colaboratory are accessible tools for companies, the experiments performed in this project could easily be replicated and further fine-tuned using the same tools. Another reason for training all models using A100 was that we would like to compare the efficiency between the models through the training time.

Experiments were run to determine the most efficient way to fine-tune the text classification model. All fine-tuning experiments were conducted with the GLUE QNLI dataset. Efficiency is defined here as the F1 score divided by the number of seconds required to fine-tune the model. The efficiency ratio is scaled to increase readability. Each model was fine-tuned for three epochs and the checkpoint with the smallest validation loss was selected. The classification metrics reflect an assumed decision threshold of 0.5 at this section of the paper. All experiments share the AdamW optimizer, weight decay of 0.1, a batch size of 32 and a learning rate of 5e-5 with the default linear learning rate scheduler.

Results and discussion



Fine-Tuning Experiment	Model	Epochs Trained	Training	Test				Training Time (seconds)	Efficiency (F1 / seconds x 1e4)	GPU RAM (GB)	Num Parameters	Num Free Parameters
			Accuracy	Accuracy	Precision	Recall	F1					
LoRA rank 32, Quantize float16	**ELECTRA BASE**	**3**	**0.9115**	**0.9224**	**0.9226**	**0.9224**	**0.922**	**2,743**	**3.363**	**14.3**	**111,255,556**	**1,771,778**
LoRA rank 32, Quantize float16	BERT UNCASED	3	0.8835	0.8867	0.8867	0.8867	0.887	2,751	3.223	14.3	110,664,964	1,181,186
LoRA rank 8	**ELECTRA BASE**	**3**	**0.9098**	**0.9209**	**0.9209**	**0.9209**	**0.921**	**6,011**	**1.532**	**18.8**	**110,370,820**	**887,042**
LoRA rank 32	ELECTRA BASE	3	0.9113	0.9198	0.9204	0.9198	0.92	6,047	1.521	18.8	111,255,556	1,771,778
Freeze transformer layers 1-10	**ELECTRA BASE**	**3**	**0.9615**	**0.9182**	**0.9186**	**0.9182**	**0.918**	**6,063**	**1.514**	**18**	**109,483,778**	**38,605,058**
LoRA rank 8	BERT UNCASED	3	0.881	0.8874	0.8874	0.8874	0.887	5,996	1.480	18.8	109,780,228	296,450
LoRA rank 32	BERT UNCASED	3	0.8834	0.8894	0.8894	0.8894	0.889	6,036	1.473	18.8	110,664,964	1,181,186
Freeze transformer layers 1-8	ELECTRA BASE	3	0.9381	0.9212	0.9224	0.9213	0.921	6,448	1.429	18.6	109,483,778	52,780,802
Freeze transformer layers 1-6	ELECTRA BASE	3	0.989	0.9248	0.9249	0.9248	0.925	6,647	1.391	19.4	109,483,778	66,956,546
Freeze transformer layers 1-4	ELECTRA BASE	3	0.9765	0.9242	0.9246	0.9242	0.924	7,015	1.317	20.5	109,483,778	81,132,290
Freeze transformer layers 1-2	ELECTRA BASE	3	0.9792	0.9213	0.9217	0.9213	0.921	7,264	1.268	21.4	109,483,778	95,308,034
Fine-tune all layers	**ELECTRA BASE**	**3**	**0.9782**	**0.9233**	**0.9236**	**0.9233**	**0.923**	**7,598**	**1.215**	**22.1**	**109,483,778**	**109,483,778**
(Baseline) Fine-tune all layers	**BERT UNCASED**	**3**	**0.9551**	**0.9081**	**0.9082**	**0.9081**	**0.9081**	**7,586**	**1.197**	**22.5**	**109,483,778**	**109,483,778**

Table 1: Fine-tuning experiment results with GLUE QNLI. The rows were sorted by "Efficiency" as defined by F1-score divided by the number of seconds trained.

Fine-tuning was approached multiple ways. Our baseline was to use a pre-trained BERT model and make all weights trainable.

Stemming from the observation that early layers in a Convolutional Neural Networks extract general features and do not need to be fine-tuned to achieve comparable results, experiments were run where the first number of transformer blocks were held frozen during fine-tuning (Sajjad, Dalvi, Durrani, and Nakov, 2023). The table above shows that fine-tuning on as few as the final two transformer blocks (of twelve) need to be fine-tuned to achieve performance near the result of fine-tuning all parameters. This was the fifth most efficient approach evaluated.

Low-Rank Adaptation (LoRA) was evaluated as an efficient fine-tuning approach (Hu, Shen, Wallis, Allen-Zhu, Li, Wang, Wang, and Chen, 2021). In the experiments, the base-weights were frozen and rank decomposition matrices were trained. This massively reduces the number of free parameters during training. LoRA decreased training times and importantly reduced the GPU memory required to train the parameters with back propagation. This can allow for further cost optimization as less expensive GPUs can be used during training. Two rank values were evaluated with LoRA, rank 8 and rank 32, the former resulting in roughly half the number of free parameters than the later.

Quantization was evaluated as the final method towards efficient fine-tuning. During training, weights were considered at half precision, float16 instead of the default float32. This dramatically increased training speed and GPU memory requirements.

The results of the experiments indicate that the most efficient fine-tuning approach evaluated consists of fine-tuning a pre-trained ELECTRA model, freezing the transformer weights, and training with LoRA and half precision quantization. This approach is ~3 times more efficient than the baseline approach of fully fine-tuning a pre-trained BERT transformer.

A larger table is presented in the Appendix which shows the same experiments run for up to six epochs to start to evaluate the stability of fine-tuning (Mosbach, Andriushchenko, and Klakow, 2021). These results showed that the recipe of using LoRA with half precision quantization during training when trained for six epochs (twice as many) was more efficient than any other fine-tuning method when trained for three epochs. Further, fine-tuning all and freezing the initial transformer layers resulted in less stable training with overfitting typically occurring in the first three epochs. Fine-tuning with LoRA proved more stable and showed continued learning performance through six epochs, suggesting that overfitting was not yet reached. This suggests that the best results may be achieved by following the recipe and fine-tuning for more epochs Fine-tuning with the recipe for nine epochs with an A100 GPU would cost nearly the same as training the baseline for three epochs. However, since the learning improvements are minimal after three epochs, following the recipe for three epochs proved the most efficient.

Performance Analysis

As baseline models, we chose the ELECTRA Base and the BERT Uncased models from Hugging Face. From there, we applied quantization, LoRA, and freeze tuning on the fine-tuning of our ELECTRA base model on QNLI (Question Natural Language Inference).

F1 scores were high across the board, with all models but BERT (0.8958) coming in above 0.9. These high F1 scores showed robust models, balancing precision and recall well. No model performed significantly better than the others. Precision scores were high, with all but the ELECTRA Base model (0.9136) hovering around the 0.93 mark. These scores indicated that our optimized models would predict a positive class correctly about 93% of the time.

Precision is of special interest, so we performed a confidence threshold analysis whereby we determined the best threshold for a given targeted precision value. Our aim is to build a system with a precision of 0.95 or higher in the classification of the hypothesis-premise entailment. We calculated the appropriate thresholds for each model, then we applied that threshold to the models, yielding the targeted precision of 0.95. The ELECTRA Base model required the strictest confidence threshold of 0.9162, therefore only providing a positive classification when absolutely sure. The result would be fewer False Positives but more False Negatives. For high precision, we aim towards more True Positives and fewer False Positives. Our three optimized models required the smallest thresholds to reach the targeted precision of 0.95, at an average increase in threshold of 0.1789 vs the base 0.5. Recall and accuracy decreased by an average of -0.0334 and -0.0062, respectively. The average F1 scores for the optimized models remained balanced at an average of 0.9127.

Error Analysis

We performed an error analysis in an attempt to understand the types of errors made by the model, as well as a qualitative review of a random sampling (30 per model) of those errors.

For our five models, we observed an overall average error rate of 8.56%, which is a strong indication of the models' accuracy. The most common questions that led to errors began with "what," accounting for about 4% of the mistakes. The errors appeared to not be correlated with question length. Both correct and incorrect predictions were associated with questions of similar word counts, ranging from 8 to 11 words. In particular, we did not see that the model struggled with very short (3-4 words) or very long (21-33 words) questions.

Our manual review categorized errors into four types: "Question understanding", "Context understanding", "Factual errors", and "Logical errors", with corresponding average error rates of 0.7600, 0.1067, 0.10000, and 0.0267. The models seemed most error-prone due to them not understanding the question being asked. Context understanding and logic appeared accurate.

To improve upon our results, we look to data augmentation to train our models with more questions that begin with “what” and a more uniform number of words per question. More domain-specific and/or targeted questions could help our models with question understanding.

Conclusion

Current Question Answering systems consist of one-model systems to perform the tasks of QA. Though lean and efficient, they tend to struggle with nuanced language understanding. We propose a dual-model system to first screen context for relevancy, then feed that pre-screened context to a text span extraction model (future work). The ELECTRA model fine-tuned on QNLI and optimized with quantization, LoRA, and freeze tuning showed promising results of improved efficiency (3x versus baseline) and inference precision (95%). Future work will address the second model for text extraction.

References

Clark, K., Luong, M.-T., Le, Q. V., & Manning, C. D. (2020). ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. arXiv. https://arxiv.org/abs/2003.10555

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. arXiv. https://arxiv.org/abs/2106.09685

Ishwari, K. S. D., Aneeze, A. K. R. R., Sudheesan, S., Karunaratne, H. J. D. A., Nugaliyadde, A., & Mallawarrachchi, Y. (2019). Advances in Natural Language Question Answering: A Review. arXiv. https://arxiv.org/abs/1904.05276

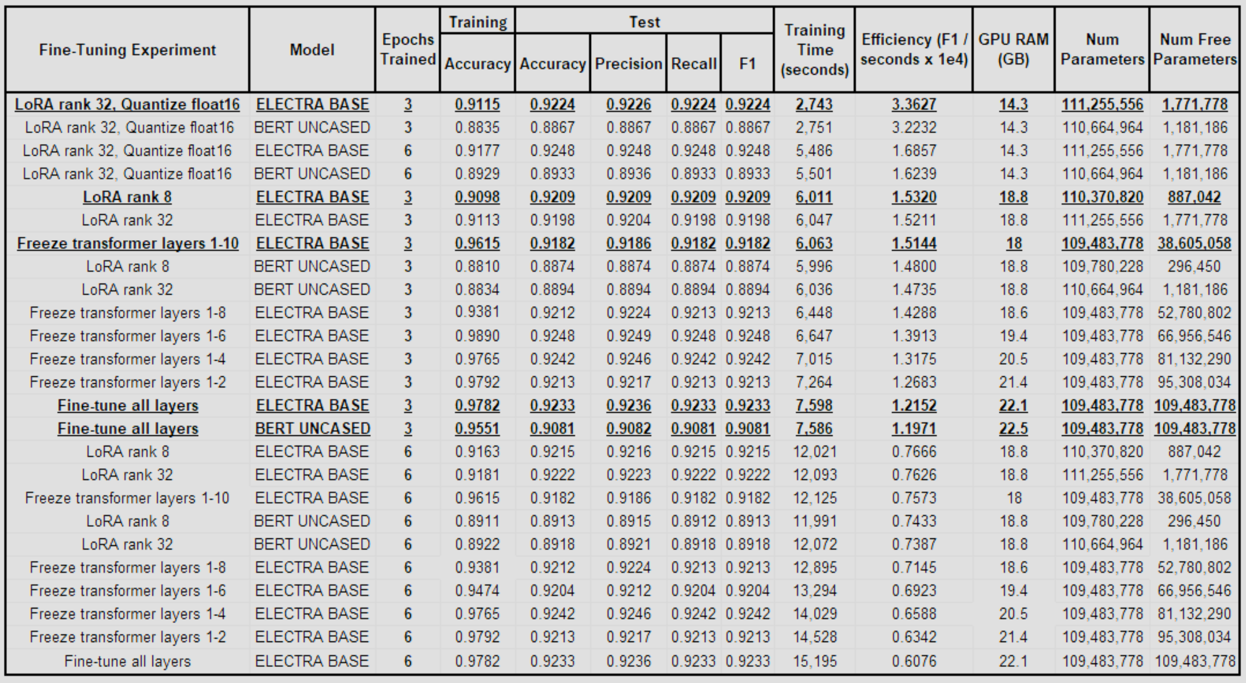
Mosbach, M., Andriushchenko, M., & Klakow, D. (2021). On the stability of fine-tuning BERT: Misconceptions, explanations, and strong baselines. arXiv. https://arxiv.org/abs/2006.04884

Sajjad, H., Dalvi, F., Durrani, N., & Nakov, P. (2023). On the effect of dropping layers of pre-trained transformer models. Computer Speech & Language, 77, 101429. https://doi.org/10.1016/j.csl.2022.101429

Storks, S., Gao, Q., & Chai, J. Y. (2020). Recent Advances in Natural Language Inference: A Survey of Benchmarks, Resources, and Approaches. arXiv. https://arxiv.org/abs/1904.01172

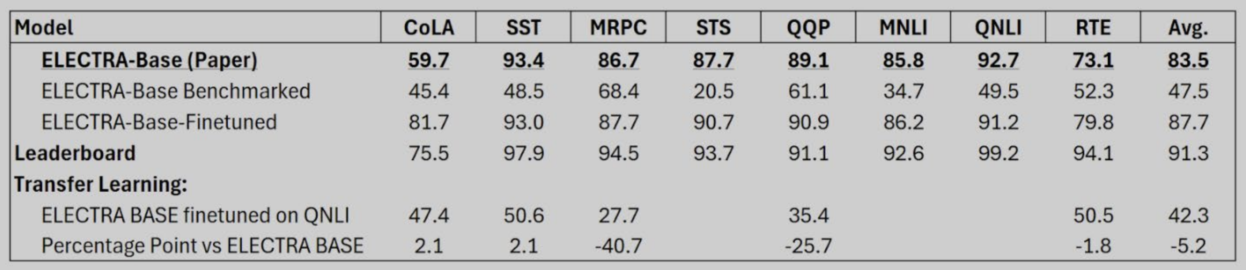
Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. R. (2019). GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. arXiv. https://arxiv.org/abs/1804.07461

Appendix



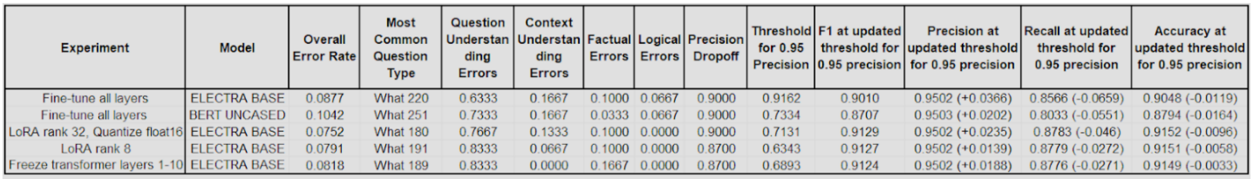
Fine-Tuning Experiment	Model	Epochs Trained	Training	Test				Training Time (seconds)	Efficiency (F1 / seconds x 1e4)	GPU RAM (GB)	Num Parameters	Num Free Parameters
			Accuracy	Accuracy	Precision	Recall	F1					
LoRA rank 32, Quantize float16	**ELECTRA BASE**	**3**	**0.9115**	**0.9224**	**0.9226**	**0.9224**	**0.9224**	**2,743**	**3.3627**	**14.3**	**111,255,556**	**1,771,778**
LoRA rank 32, Quantize float16	BERT UNCASED	3	0.8835	0.8867	0.8867	0.8867	0.8867	2,751	3.2232	14.3	110,664,964	1,181,186
LoRA rank 32, Quantize float16	ELECTRA BASE	6	0.9177	0.9248	0.9248	0.9248	0.9248	5,486	1.6857	14.3	111,255,556	1,771,778
LoRA rank 32, Quantize float16	BERT UNCASED	6	0.8929	0.8933	0.8936	0.8933	0.8933	5,501	1.6239	14.3	110,664,964	1,181,186
LoRA rank 8	**ELECTRA BASE**	**3**	**0.9098**	**0.9209**	**0.9209**	**0.9209**	**0.9209**	**6,011**	**1.5320**	**18.8**	**110,370,820**	**887,042**
LoRA rank 32	ELECTRA BASE	3	0.9113	0.9198	0.9204	0.9198	0.9198	6,047	1.5211	18.8	111,255,556	1,771,778
Freeze transformer layers 1-10	**ELECTRA BASE**	**3**	**0.9615**	**0.9182**	**0.9186**	**0.9182**	**0.9182**	**6,063**	**1.5144**	**18**	**109,483,778**	**38,605,058**
LoRA rank 8	BERT UNCASED	3	0.8810	0.8874	0.8874	0.8874	0.8874	5,996	1.4800	18.8	109,780,228	296,450
LoRA rank 32	BERT UNCASED	3	0.8834	0.8894	0.8894	0.8894	0.8894	6,036	1.4735	18.8	110,664,964	1,181,186
Freeze transformer layers 1-8	ELECTRA BASE	3	0.9381	0.9212	0.9224	0.9213	0.9213	6,448	1.4288	18.6	109,483,778	52,780,802
Freeze transformer layers 1-6	ELECTRA BASE	3	0.9890	0.9248	0.9249	0.9248	0.9248	6,647	1.3913	19.4	109,483,778	66,956,546
Freeze transformer layers 1-4	ELECTRA BASE	3	0.9765	0.9242	0.9246	0.9242	0.9242	7,015	1.3175	20.5	109,483,778	81,132,290
Freeze transformer layers 1-2	ELECTRA BASE	3	0.9792	0.9213	0.9217	0.9213	0.9213	7,264	1.2683	21.4	109,483,778	95,308,034
Fine-tune all layers	**ELECTRA BASE**	**3**	**0.9782**	**0.9233**	**0.9236**	**0.9233**	**0.9233**	**7,598**	**1.2152**	**22.1**	**109,483,778**	**109,483,778**
Fine-tune all layers	**BERT UNCASED**	**3**	**0.9551**	**0.9081**	**0.9082**	**0.9081**	**0.9081**	**7,586**	**1.1971**	**22.5**	**109,483,778**	**109,483,778**
LoRA rank 8	ELECTRA BASE	6	0.9163	0.9215	0.9216	0.9215	0.9215	12,021	0.7666	18.8	110,370,820	887,042
LoRA rank 32	ELECTRA BASE	6	0.9181	0.9222	0.9223	0.9222	0.9222	12,093	0.7626	18.8	111,255,556	1,771,778
Freeze transformer layers 1-10	ELECTRA BASE	6	0.9615	0.9182	0.9186	0.9182	0.9182	12,125	0.7573	18	109,483,778	38,605,058
LoRA rank 8	BERT UNCASED	6	0.8911	0.8913	0.8915	0.8912	0.8913	11,991	0.7433	18.8	109,780,228	296,450
LoRA rank 32	BERT UNCASED	6	0.8922	0.8918	0.8921	0.8918	0.8918	12,072	0.7387	18.8	110,664,964	1,181,186
Freeze transformer layers 1-8	ELECTRA BASE	6	0.9381	0.9212	0.9224	0.9213	0.9213	12,895	0.7145	18.6	109,483,778	52,780,802
Freeze transformer layers 1-6	ELECTRA BASE	6	0.9474	0.9204	0.9212	0.9204	0.9204	13,294	0.6923	19.4	109,483,778	66,956,546
Freeze transformer layers 1-4	ELECTRA BASE	6	0.9765	0.9242	0.9246	0.9242	0.9242	14,029	0.6588	20.5	109,483,778	81,132,290
Freeze transformer layers 1-2	ELECTRA BASE	6	0.9792	0.9213	0.9217	0.9213	0.9213	14,528	0.6342	21.4	109,483,778	95,308,034
Fine-tune all layers	ELECTRA BASE	6	0.9782	0.9233	0.9236	0.9233	0.9233	15,195	0.6076	22.1	109,483,778	109,483,778

Table 2: Fine-tuning experiment results with GLUE QNLI. The rows were sorted by "Efficiency" as defined by F1-score divided by the number of seconds trained. This is an expanded version of table 1.



Model	CoLA	SST	MRPC	STS	QQP	MNLI	QNLI	RTE	Avg.
ELECTRA-Base (Paper)	**59.7**	**93.4**	**86.7**	**87.7**	**89.1**	**85.8**	**92.7**	**73.1**	**83.5**
ELECTRA-Base Benchmarked	45.4	48.5	68.4	20.5	61.1	34.7	49.5	52.3	47.5
ELECTRA-Base-Finetuned	81.7	93.0	87.7	90.7	90.9	86.2	91.2	79.8	87.7
Leaderboard	75.5	97.9	94.5	93.7	91.1	92.6	99.2	94.1	91.3
Transfer Learning:									
ELECTRA BASE finetuned on QNLI	47.4	50.6	27.7		35.4			50.5	42.3
Percentage Point vs ELECTRA BASE	2.1	2.1	-40.7		-25.7			-1.8	-5.2

Table 3: Replicating the GLUE task fine-tuning from the ELECTRA paper.



Experiment	Model	Overall Error Rate	Most Common Question Type	Question Understanding Errors	Context Understanding Errors	Factual Errors	Logical Errors	Precision Dropoff	Threshold for 0.95 Precision	F1 at updated threshold for 0.95 precision	Precision at updated threshold for 0.95 precision	Recall at updated threshold for 0.95 precision	Accuracy at updated threshold for 0.95 precision
Fine-tune all layers	ELECTRA BASE	0.0877	What 220	0.6333	0.1667	0.1000	0.0667	0.9000	0.9162	0.9010	0.9502 (+0.0366)	0.8566 (-0.0659)	0.9048 (-0.0119)
Fine-tune all layers	BERT UNCASED	0.1042	What 251	0.7333	0.1667	0.0333	0.0667	0.9000	0.7334	0.8707	0.9503 (+0.0202)	0.8033 (-0.0551)	0.8794 (-0.0164)
LoRA rank 32, Quantize float16	ELECTRA BASE	0.0752	What 180	0.7667	0.1333	0.1000	0.0000	0.9000	0.7131	0.9129	0.9502 (+0.0235)	0.8783 (-0.046)	0.9152 (-0.0096)
LoRA rank 8	ELECTRA BASE	0.0791	What 191	0.8333	0.0667	0.1000	0.0000	0.8700	0.6343	0.9127	0.9502 (+0.0139)	0.8779 (-0.0272)	0.9151 (-0.0058)
Freeze transformer layers 1-10	ELECTRA BASE	0.0818	What 189	0.8333	0.0000	0.1667	0.0000	0.8700	0.6893	0.9124	0.9502 (+0.0188)	0.8776 (-0.0271)	0.9149 (-0.0033)

Table 4: Error analysis of select fine-tuned models.