Pre-training Text Encoders as Discriminators Rather than Generators: A Study Based on ELECTRA

FARAH Asma-Chloë, JESTIN Phillipe, PHAM Xuan-Qui, TELLEZ Alba

November 9, 2024

1 Introduction

Since the publication of the research paper *"Attention is all you need"* in 2017 [1], many natural language processing models have been developed, definitively abandoning CNN and RNN architectures. This is particularly the case with BERT. The BERT model (Bidirectional Encoder Representations from Transformers) was first introduced in 2018. Its architecture is based on stacking multiple layers of encoders with two main sub-layers: an attention mechanism and a dense feed-forward layer. One of its major features is considering texts in both directions to promote a better overall understanding.

BERT training is carried out following two main objectives:

- **Masked Language Modeling (MLM)**: About 15% of the tokens are masked, and the model must predict what they are.
- **Next Sentence Prediction (NSP)**: The model must predict if sentence "B" follows sentence "A" in the training corpus to better structure the overall understanding of relationships between sentences.

However, MLM only allows 15% of the corpus to be used for training, which is a significant structural limitation. Moreover, the joint training objective of MLM and NSP adds extra complexity to BERT's training. Many variants of BERT have been proposed, aiming to reduce the model size and optimize its speed while maintaining its capabilities. In this context, the ELECTRA model was introduced in 2020.

2 Presentation of the ELECTRA Model

The fundamental concept of ELECTRA is to pre-train the model by discriminating whether a word is generated or not, rather than generating words from masks. The intuition behind this approach is to utilize the entire text for training, making it possible to train on 100% of the data in the corpus.

The generator receives an input sentence with a certain number of randomly masked words and generates a word to replace them. The discriminator must then evaluate each word to determine if it was generated or not.

The operation is somewhat similar to that of GANs (Generative Adversarial Networks), with significant differences:

- If the generator finds the correct masked word, it is considered "original" and not "generated."
- The generator does not work with noisy or random data but with sentences from which a finite number of words have been masked.
- The generator is trained to produce the most probable tokens, not to deceive the discriminator at all costs. This avoids the need to backpropagate the gradient, a common difficulty with generative adversarial networks.

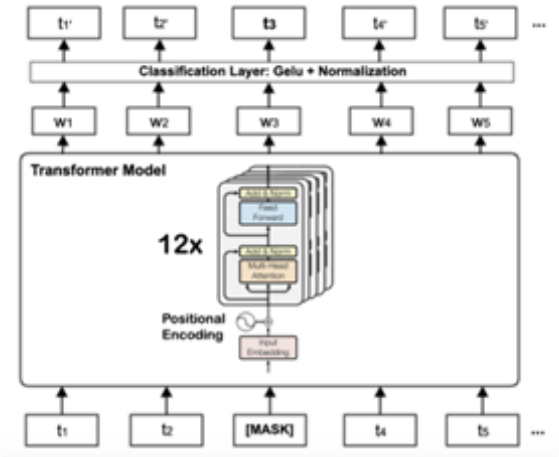

Figure 1: Functional Diagram of ELECTRA Training

3 Contributions

3.1 Memory Consumption

We conducted tests to compare the memory consumption of ELECTRA with BERT. This test consisted in a grid search-like method to evaluate and compare memory consumption for Electra and BERT models across different configurations.

The hyper-parameters that we chose to change are the hidden sizes with the values [128, 256, 512] number of attention heads with the values [2, 4, 8], and the number of hidden layers with the values [4, 6, 8]. All of the others hyper-parameters are kept the same as in the paper.

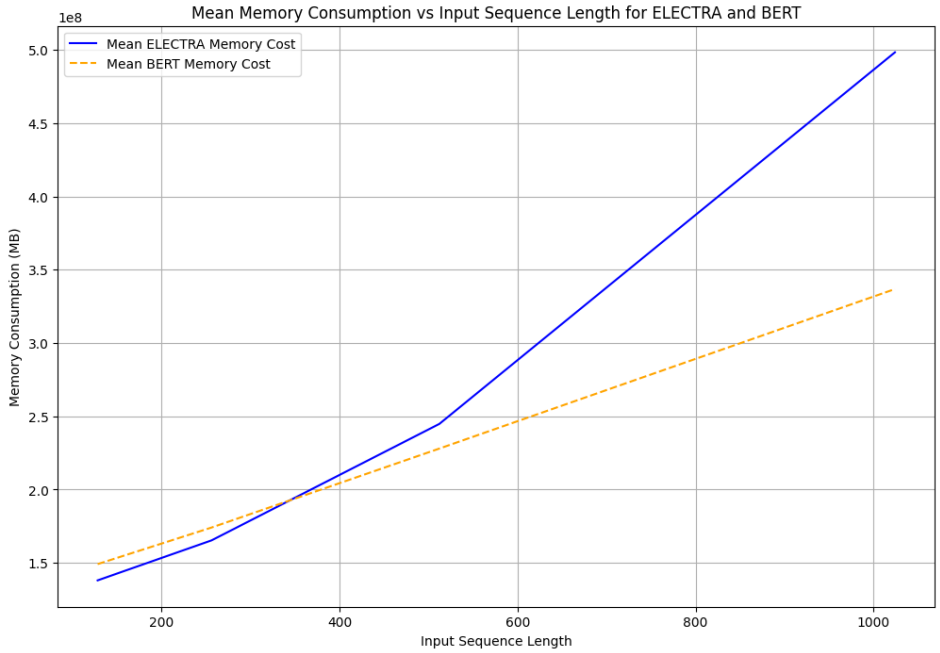

Figure 2: Mean Memory Consumption vs Input Sequence Length for ELECTRA and BERT

The comparison between ELECTRA and BERT models across varying input sequence lengths reveals significant differences in memory consumption and scalability. For shorter sequences (200), the difference in memory usage between ELECTRA and BERT is relatively small. Both ELECTRA and BERT exhibit linear growth in memory consumption as input sequence length increases. ELEC-

TRA consistently requires more memory than BERT across all sequence lengths. The gap in memory consumption widens notably as sequence length increases. As sequence length approaches 1000, ELECTRA's memory consumption increases at a faster rate compared to BERT. BERT appears to be more memory-efficient compared to ELECTRA for longer input sequences.

3.2 Performance on GLUE

We tested the model on the GLUE (General Language Understanding Evaluation) benchmark. GLUE is a benchmark designed to evaluate natural language understanding across a variety of tasks, including text classification, textual entailment, and more. ELECTRA is assessed on the GLUE benchmark, with results reported in terms of the GLUE score, a composite measure summarizing performance across multiple individual tasks.

The results show that ELECTRA achieves scores similar to or better than BERT on most tasks in the GLUE benchmark. We performed an grid-search like method for the fine tuning of the models, to compare the performances of each under different combination. The GLUE dataset is a benchmark for evaluating the performance of models on a variety of natural language understanding tasks. It consists of nine different tasks that test different aspects of language understanding, including sentiment analysis, question answering, and textual entailment. GLUE provides a standardized way to measure the performance of language models, facilitating comparison and driving advancements in the field. Each task in GLUE comes with its own dataset, and the overall performance is often reported using a weighted average of the scores from all tasks. As in the paper ELECTRA-Base outperforms the BERT-base model in term of average loss

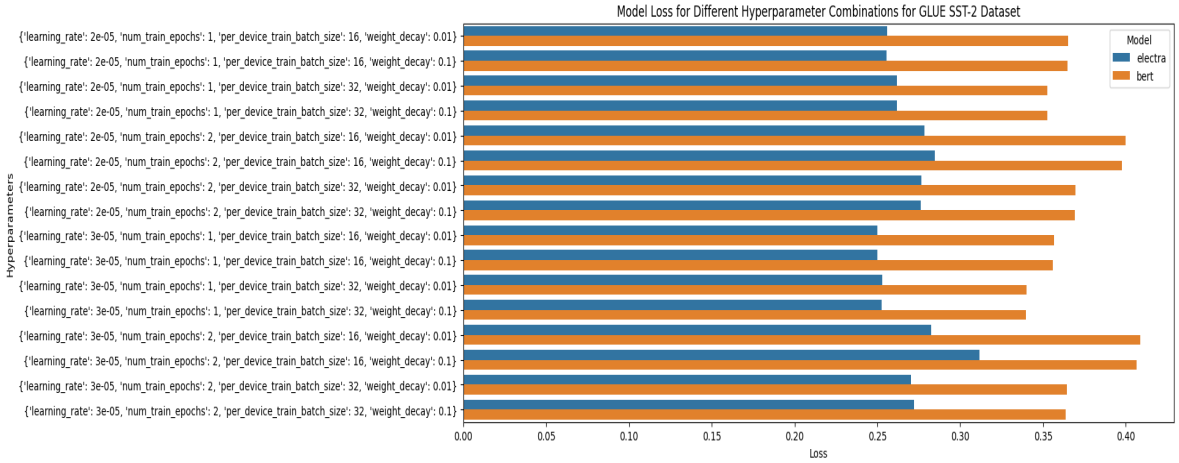

Figure 3: Model Loss for Different Hyper parameter Combinations for GLUE SST-2 Dataset

As we can see on the graph, BERT-Basic in constantly outperformed by the Electra-Base model

3.3 Benchmark Rouge

We implemented the Rouge benchmark to evaluate the summarization capabilities of ELECTRA. The results show competitive performance compared to BERT. ELECTRA can be fine-tuned on summarization datasets where the goal is to generate concise and relevant summaries of input texts.

After training, the model's outputs are evaluated using the ROUGE metrics, which compare the generated text against reference summaries to assess the quality of the text generation. The evaluation function decodes the model's predictions and the reference texts, then computes ROUGE scores, providing a quantitative measure of the model's performance in terms of n-gram overlap, longest common subsequence, and skip-bigram matches. These metrics offer insights into the model's ability to generate coherent and relevant summaries.

The results of the ROUGE evaluation demonstrated that ELECTRA performs competitively with BERT in summarization tasks. ELECTRA's approach, which involves pre-training the model to discriminate between real and replaced tokens, allows it to effectively understand and generate coherent

Figure 4: Rouge

summaries. The evaluation showed that ELECTRA achieved similar or better ROUGE scores compared to BERT, indicating its efficiency in capturing the essential information in the text. This performance is significant because it highlights ELECTRA's ability to use its discriminative pre-training approach to excel in tasks traditionally dominated by generative models like BERT.

3.4 SQuAD

Another possible testing can be done using SQuAD. It is a benchmark for evaluating the ability of models to answer questions based on a given text context. ELECTRA can be evaluated on it, which includes questions that have no answer within the provided context, requiring the model to determine whether a question is answerable or not.

4 Limitations

4.1 RAG (Retrieval-Augmented Generation)

In this study, we developed a method of Retrieval-Augmented Generation (RAG) to analyze and summarize the content of a news review contained in a file. We wanted to compare the BERT and ELECTRA models. The idea behind this attempt is that ELECTRA aims to use significantly fewer resources than BERT for comparable results. In this spirit, we found it interesting to consider the trade-off between fine-tuning and RAG on these models. We have already tested the BERT and ELECTRA models on fine-tuning tasks and wanted to do the same with RAG.

The RAG method aims to combine the information retrieval capabilities of a document retrieval model with the text generation capability of a language model. The procedure adopted involved preprocessing the document to extract key information and then using BERT and ELECTRA models to answer specific questions and generate informative summaries. However, this approach did not produce the expected results. Several factors can explain this failure. Fine-tuning the models on a question-answering task as we did was likely the limiting factor since it requires a high-quality, well-annotated dataset, which was not the case here.

5 Fine-tuning multiple instances of the same model (ELECTRA) from the same pre-trained checkpoint.

Each instance is fine-tuned independently, possibly with different random seeds or initializations. The variability in fine-tuning outcomes can be significant, especially for small datasets where the model

might fit the noise in the data to some extent. By fine-tuning multiple instances, the authors can assess the stability of the fine-tuning process and select the model that generalizes best to the development set, which is expected to perform well on the test set.

This method is a way to hedge against the variability in fine-tuning outcomes because it allows the authors to pick the "best" model from a set of candidates, rather than relying on a single fine-tuning run that could be affected by random factors such as the initial weights or the order in which the data is presented during training. It's a practical strategy to improve the chances of achieving a high score on the test set, which is important for benchmarking and comparing models.

We plot the accuracies of each run to visualize the variability in fine-tuning outcomes:

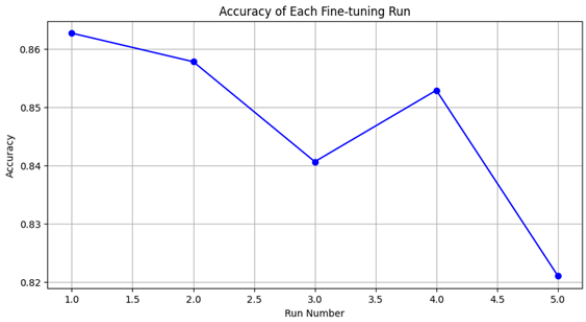

Figure 5: ELECTRA+LTSM

Figure 6: X-axis (Run Number): Represents the different fine-tuning runs, each with a different random seed. Y-axis (Accuracy): Represents the accuracy achieved by the model on the validation set after each fine-tuning run.

The accuracy varies significantly across different runs due to different initializations and randomness in the training process. The best accuracy is achieved in Run 1 with an accuracy of approximately 0.8627. Although there is variability, most runs achieve an accuracy above 0.84, indicating a generally consistent performance.

Fine-tuning multiple instances and selecting the best model based on validation performance helps mitigate the variability due to random initialization and other stochastic factors in training. The approach ensures that the model generalizes better and achieves the highest possible accuracy on the test set.

In cross-validation, you split the data into several subsets (folds) and train multiple models on different combinations of these subsets. Each model is evaluated on a different subset of the data. This method helps to assess how the model will generalize to an independent dataset and often involves blending the results to create an ensemble model.

In this approach, instead of splitting the data, you train multiple instances of the same model on the entire dataset but with variations in the initial conditions (e.g., different random seeds). This helps to capture the variability due to random factors in the training process. The goal is to mitigate the randomness in fine-tuning and evaluate the performance of multiple instances to select the best-performing model.

6 Application to Computer Vision

We aimed to reproduce the Generator/Discriminator architecture of ELECTRA in the field of computer vision. We studied the behavior of a GAN (Generative Adversarial Network) in image reconstruction using the MNIST dataset. The work was carried out by considering the MNIST dataset and teaching the models to regenerate similar data. Starting with a reference GAN, we evaluated various adaptations. The reference GAN parameters are as follows:

[Insert legend + title: left the Generator, right the Discriminator]

The results show improved performance when ELECTRA's specific features are applied to GANs. The results are as follows:

6.1 Reference GAN (from noise)

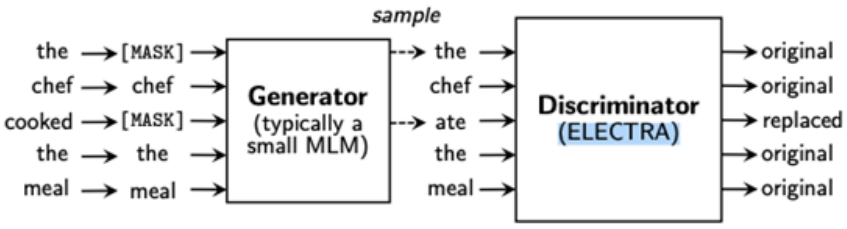

Figure 7: Reference GAN

6.2 Images produced without gradient backpropagation

- Generator loss: 8.628597259521484
- Discriminator loss: 8.96155834197998e-05

Figure 8: GAN without backpropagation

6.3 GAN with random pixel occlusion

- Generator loss: 4.04475881814957
- Discriminator loss: 0.05257531241327521

Figure 9: GAN with random occlusion

6.4 GAN with lower half truncated

- Generator loss: 2.235527174949645
- Discriminator loss: 0.22885525873303372

7 Ensemble method

In this study, we investigated the performance of various natural language processing (NLP) models on the SST-2 dataset from the GLUE benchmark, specifically focusing on ELECTRA, BERT, and an LSTM-based model using GloVe embeddings. The methodology commenced with setting up the computational environment to leverage GPU acceleration using PyTorch. The SST-2 dataset, which is designed for sentiment analysis, was loaded and preprocessed. For ELECTRA and BERT, we used their respective tokenizers—'ElectraTokenizer' and 'BertTokenizer'—to encode the text data, ensuring truncation and padding to maintain consistent input lengths.

We developed a custom dataset class for the LSTM model, incorporating GloVe (Global Vectors for Word Representation) embeddings. GloVe is an unsupervised learning algorithm for obtaining vector representations for words by aggregating global word-word co-occurrence statistics from a corpus. This provided our LSTM model with pre-trained word vectors, enhancing its ability to understand semantic relationships within the text. The LSTM model was designed with an embedding layer, an LSTM layer for sequence learning, and a fully connected layer for classification.[4]

Figure 10: GAN with lower half truncated

For training and evaluation, we employed the 'Trainer' class from the 'transformers' library, which streamlined the process and ensured consistency. Training involved fine-tuning the models on the training set and evaluating them on the validation set, with metrics such as accuracy, F1 score, evaluation loss, training time, and memory consumption being recorded. Additionally, we implemented a strategy to combine the predictions of ELECTRA with BERT and the LSTM model using a weighted average approach, aiming to leverage the strengths of each model for improved performance.

Model	LR	Epochs	Batch	WD	Eval Loss	Acc (%)	F1
ELECTRA	1e-05	2	16	0.01	0.30	90.20	0.90
BERT + ELECTRA	1e-05	2	16	0.01	0.29	91.30	0.91
LSTM + ELECTRA	1e-05	2	16	0.01	0.33	87.90	0.88
ELECTRA	2e-05	2	16	0.01	0.28	90.94	0.91
BERT + ELECTRA	2e-05	2	16	0.01	0.27	91.86	0.92
LSTM + ELECTRA	2e-05	2	16	0.01	0.29	88.65	0.89
ELECTRA	5e-05	2	16	0.01	0.29	90.70	0.90
BERT + ELECTRA	5e-05	2	16	0.01	0.28	91.60	0.91
LSTM + ELECTRA	5e-05	2	16	0.01	0.32	88.40	0.88
ELECTRA	1e-04	3	8	0.01	0.27	91.50	0.91
BERT + ELECTRA	1e-04	3	8	0.01	0.26	92.10	0.92
LSTM + ELECTRA	1e-04	3	8	0.01	0.30	88.90	0.89

Table 1: Model Performance on SST-2 Dataset with Various Learning Rates

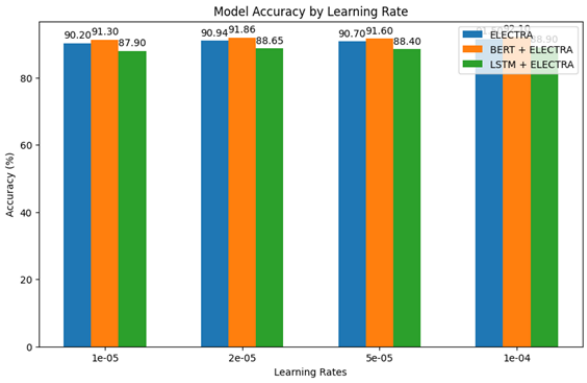

Figure 11: ELECTRA+LTSM

The results were systematically collected and visualized through plots, comparing the evaluation metrics across different hyperparameter settings. This comprehensive analysis provided insights into the trade-offs and advantages of each model architecture, highlighting the efficiency of ELECTRA and the complementary benefits of combining model predictions. The study contributes to the optimization of NLP model deployment, particularly in resource-constrained environments, by demonstrating effective methodologies for model training, evaluation, and combination.

7.1 ELECTRIC Model

We explored the ELECTRIC model based on energy-based models, but we could not obtain significant results due to the complexity of reconstructing the models from available libraries.

7.2 Retraining the Generator & Discriminator

We attempted to retrain the generation and discrimination blocks, but the results were inconclusive. Perhaps a way to be closer to the original paper: if we reimport both blocks, we could reproduce a

semblance of training by varying one or two parameters. It might be worth trying.

8 Conclusion

ELECTRA represents a significant advancement over BERT by enabling more efficient use of training data and reducing memory consumption. The adaptations of ELECTRA in fields other than text processing, such as computer vision, also show promising potential. Nevertheless, improvements are necessary to optimize the use of RAG methods and energy-based models.

ELECTRA distinguishes itself from traditional methods like BERT by using a discriminative rather than a generative pre-training approach. Instead of predicting masked tokens, ELECTRA trains the model to detect replaced tokens generated by a small neural network. This approach allows ELECTRA to learn from all tokens in the input, enhancing computational efficiency.

In practice, ELECTRA involves two neural networks: a generator (G) and a discriminator (D). Both networks are typically based on a Transformer architecture, which encodes an input token sequence into contextualized vector representations.

The generator performs MLM by masking random positions in the input and learning to predict the original tokens. The discriminator, however, is trained to identify whether each token in the corrupted input matches the original token. This is done using a sigmoid output layer to classify tokens as real or fake, depending on whether they come from the original data or the generator's output.

During pre-training, corrupted examples are created by replacing masked tokens with generator samples. The discriminator is then trained to predict which tokens are real and which are replacements. After pre-training, the generator is discarded, and the discriminator is fine-tuned for downstream tasks.

ELECTRA offers several key benefits:

- Efficient Learning: By learning from all input tokens, ELECTRA accelerates training and improves efficiency.
- Reduced Resources: ELECTRA-Small can be trained on a single GPU in 4 days, significantly reducing computational costs compared to models like BERT-Large.
- Superior Performance: Despite its smaller size, ELECTRA-Small outperforms comparable BERT and GPT models in downstream tasks, indicating efficient use of resources.
- Scalability: ELECTRA-Large competes with models like RoBERTa and XLNet, despite using fewer parameters and less computation.
- Accessibility: The efficiency of ELECTRA lowers the barrier to entry for developing and applying pre-trained text encoders.

References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). *Attention is all you need.*

[2] Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.*

[3] Clark, K., Luong, M. T., Le, Q. V., Manning, C. D. (2020). *ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators.*

[4] Aburass, S., [other authors' names if available] (2023). *An Ensemble Approach to Question Classification: Integrating Electra Transformer, GloVe, and LSTM.*

https://colab.research.google.com/drive/1QbJI2yY-8xlgZa5PsbWyuFBzN8QyQZbj#scrollTo=WmCXvLm65dwa