SQuAD 2.0 Project Report

George Washington University

Columbian College of Arts & Sciences

DATS 6312 Natural Language Processing

Final Project - Group 7

Members: Ei Tanaka

Date: December 11, 2023

Table of Contents

1. Introduction

In the dynamic realm of Natural Language Processing (NLP), the advent of question-answering (QA) systems marks a significant stride in our ability to interact with and process digital information. This project is dedicated to developing a reading comprehension-based QA system inspired by the comprehensive insights in "Speech and Language Processing" by Daniel Jurafsky & James H. Martin (2023). We aim to create a system that can interpret and respond to questions posed in natural language, drawing answers from provided text passages.

2. Description of Individual Work

I was responsible for the following tasks:

- ELECTRA Fine-tuning
- Building the Streamlit Demo

3. ELECTRA

ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) represents a significant shift in pre-training text encoders in Natural Language Processing. Introduced by Clark, Luong, Le, Manning, and others (2020), ELECTRA diverges from the traditional denoising autoencoder methods, such as those employed by BERT (Devlin et al., 2019) and XLNet (Yang et al., 2019), which mask a portion of the input tokens and train the network to predict these masked tokens.

ELECTRA innovates on this concept by introducing replaced token detection as its pre-training task. Unlike the masked language modeling (MLM) approach, which learns from only a fraction of the tokens (typically 15%) in an input sequence, ELECTRA corrupts the input by replacing some tokens with samples from a generator model. The network is then trained as a discriminator to identify whether each token in the input is original or a replacement. This methodology addresses a mismatch in BERT—where the network encounters artificial [MASK] tokens during pre-training but not during fine-tuning for downstream tasks.

One of the critical advantages of ELECTRA's approach is its computational efficiency. Since the model learns from all input tokens rather than just a tiny subset, it demonstrates a more efficient use of computational resources. Clark et al. (2020) also note that ELECTRA, when trained on datasets like GLUE (Wang et al., 2019) and SQuAD (Rajpurkar et al., 2016), outperforms MLM-based methods like BERT and XLNet given the same model size, data, and compute. The approach is also notable for its scalability and effectiveness, with variants like ELECTRA-Small and ELECTRA-Large achieving high performance on benchmark tasks, showcasing compute and parameter efficiency compared to existing generative approaches in language representation learning.

3.1 Pre-trained Method

The methodology centers on replaced token detection, a novel approach Clark et al. (2020) introduced in their ELECTRA model. This technique utilizes two neural networks, a generator and a discriminator, based on the Transformer architecture. The generator, tasked with masked language modeling, randomly masks tokens in an input sequence and tries to predict their original identities. The discriminator, conversely, is trained to distinguish between authentic tokens and those replaced by the generator.

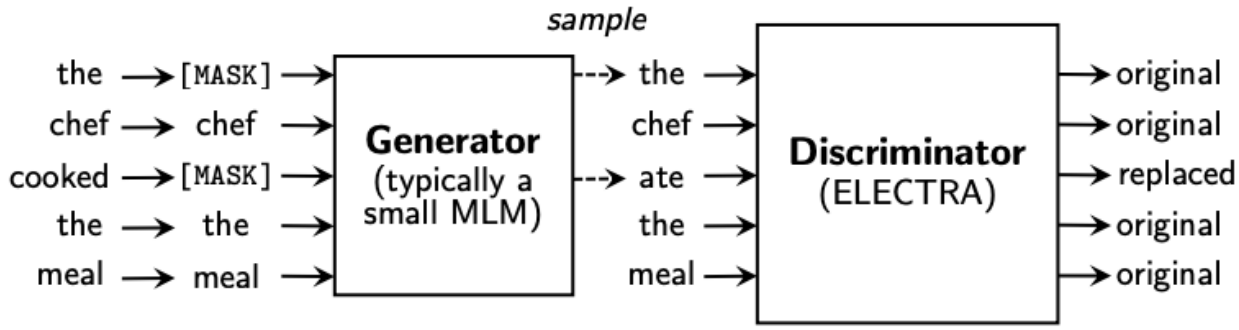

Fig-[n]: An overview of replaced token detection [Adapted from “ELECTRA: PRE-TRAINING TEXT ENCODERS AS DISCRIMINATORS RATHER THAN GENERATORS,” by K. Clark et al., 2020, Figure 2].

A vital element of this approach is creating a corrupted input sequence, where tokens masked by the generator are replaced with its predictions. The discriminator is trained on this sequence to identify tokens that align with the original input. This method diverges from Generative Adversarial Networks (GANs) in significant ways. Notably, tokens accurately predicted by the generator are considered "real" by the discriminator. Additionally, the generator is trained using maximum likelihood, as adversarial training is challenging due to backpropagation complexities associated with sampling.

$$p_G(x_t|\boldsymbol{x}) = \exp\left(e(x_t)^T h_G(\boldsymbol{x})_t\right) / \sum_{x'} \exp\left(e(x')^T h_G(\boldsymbol{x})_t\right)$$

Equation 1: The generator outputs the probability of generating the particular token x_t with the softmax layer for a given portion t. Where e denotes the embedding, x denotes a sequence of input, and h denotes a contextualized vector representation.
By K. Clark et al., 2020, Equation 1.

$$D(\boldsymbol{x}, t) = \text{sigmoid}(w^T h_D(\boldsymbol{x})_t)$$

Equation 2: The discriminator predicts whether the token x_t is “real” for a given portion t.
By K. Clark et al., 2020, Equation 2.

Equation 3 describes the process of generating training data for the pre-training task. It consists of two primary operations involving the generator G and the creation of a corrupted version of the input sequence for the discriminator D to analyze.

Firstly, a set of indices m is randomly selected from the uniform distribution spanning the length of the input sequence, where i ranges from 1 to k, and k represents the number of tokens to be masked, typically 15% of the total tokens in the sequence. These indices correspond to the positions of the tokens in the input sequence x that will be masked.

Once the positions to be masked are determined, the generator G predicts replacements $\hat{x}_{i,}$ for the actual tokens at these positions. The generator's predictions are based on probabilities $P_G(x_i | x^{masked})$ calculated given the masked input x^{masked}. The masked input is obtained by replacing the selected tokens in the original input sequence x with a [MASK] token, a process denoted as $x^{masked} = REPLACE(x, m, [MASK])$.

Finally, the corrupted sequence $x^{corrupt}$ is produced by replacing the masked tokens in the original sequence x with the tokens predicted by the generator $\hat{x}$. The discriminator D uses $x^{corrupt}$ to learn to differentiate between the 'real' tokens from the input sequence and the 'fake' tokens generated by

$$m_i \sim \text{unif}\{1, n\} \text{ for } i = 1 \text{ to } k \qquad \boldsymbol{x}^{\text{masked}} = \text{REPLACE}(\boldsymbol{x}, \boldsymbol{m}, \texttt{[MASK]})$$
$$\hat{x}_i \sim p_G(x_i | \boldsymbol{x}^{\text{masked}}) \text{ for } i \in \boldsymbol{m} \qquad \boldsymbol{x}^{\text{corrupt}} = \text{REPLACE}(\boldsymbol{x}, \boldsymbol{m}, \hat{\boldsymbol{x}})$$

Equation 3: The process of generating the training data.
By K. Clark et al., 2020, Equation 3

The goal is to minimize a combined loss function over a substantial corpus, considering both the generator's masked language modeling loss and the discriminator's loss. Crucially, the discriminator's loss is not back propagated through the generator. Post-pre-training, the focus shifts to fine-tuning the discriminator for downstream tasks.

$$\mathcal{L}_{\text{MLM}}(\boldsymbol{x}, \theta_G) = \mathbb{E}\left(\sum_{i \in \boldsymbol{m}} -\log p_G(x_i | \boldsymbol{x}^{\text{masked}})\right)$$
$$\mathcal{L}_{\text{Disc}}(\boldsymbol{x}, \theta_D) = \mathbb{E}\left(\sum_{t=1}^{n} -\mathbb{1}(x_t^{\text{corrupt}} = x_t) \log D(\boldsymbol{x}^{\text{corrupt}}, t) - \mathbb{1}(x_t^{\text{corrupt}} \neq x_t) \log(1 - D(\boldsymbol{x}^{\text{corrupt}}, t))\right)$$

Equation 4: Loss functions of Generator G and Discriminator D.
By K. Clark et al., 2020, Equation 4

$$\min_{\theta_G,\theta_D} \sum_{\boldsymbol{x}\in\mathcal{X}} \mathcal{L}_{\mathrm{MLM}}(\boldsymbol{x},\theta_G) + \lambda\mathcal{L}_{\mathrm{Disc}}(\boldsymbol{x},\theta_D)$$

Equation 5: The combined loss to minimize.
By K. Clark et al., 2020, Equation 5

3.2 Experiment Setup

3.2.1 Data

Our research addresses the SQuAD 2.0 challenge, utilizing the designated datasets for training, development, and testing. Adhering to the distribution specified in the provided documentation, our dataset is partitioned into 130,319 training examples, 11,873 development examples, and 8,862 test examples.

Two preprocessing functions are coded to handle training and validation data. These functions tokenize the inputs, create sliding windows to manage sequence length, pad inputs to the maximum length, and calculate answers' start and end positions within the context. Special attention is paid to handling overflow tokens and offset mapping for precise answer localization.

3.2.2 Evaluation Method

Our model's performance is assessed using the Exact Match (EM) and F1 metrics. The F1 score is given greater weight as it is the primary criterion for judgment in the competition. Throughout the training phase, we also monitored the negative log-likelihood to indicate model fit to the data, which helped us identify potential overfitting issues.

3.2.3 ELECTRA Model - Fine-tuning Setup

The ELECTRA model is fine-tuned for the SQuAD 2.0 challenge in this experiment setup using a robust and structured approach based on HuggingFace's PyTorch implementation of BERT for question answering.

Key hyperparameters are specified, including the maximum sequence length of inputs, stride for tokenization, and the number of best predictions to generate like the following:

The maximum sequence length: 384
Stride: 128
The number of predictions to generate: 20
The maximum length of answer: 30

The tokenizer is initialized with the ELECTRA base discriminator checkpoint from Google, and training parameters like learning rate, batch size, and number of epochs are set.

Learning rate: 5e-5
Batch size: 32
Number of epochs: 3 (It took one and a half hours to train.)

The ELECTRA model is initialized for question answering, with an AdamW optimizer set with the defined learning rate. A training function orchestrates the training loop, leveraging the Accelerator for efficient multi-device training, and incorporates evaluation steps to monitor model performance using the F1 and Exact Match metrics. The losses, including negative log-likelihood, are tracked to assess the model's fit to the data and detect overfitting.

A fine-tuning step utilizes Hugging Face's Trainer API, encapsulating the training process within a streamlined API call that includes model saving. A prediction function is established to run inference on new questions and contexts, utilizing the trained model to generate answers.

4. Results

The results of our project are instrumental in understanding the capabilities of the ELECTRA and BERT with LoRA models in the context of the SQuAD 2.0 dataset. The primary focus was on evaluating the performance of these models in terms of Exact Match (EM) and F1 scores on the development set.

Models	Exact Match - Dev	F1 Score - Dev
ELECTRA (Fine-tuned)	50.07	50.07
BERT with LORA	50.98	50.98

Table-1:

4.1 ELECTRA Model Performance

The ELECTRA model, fine-tuned for the SQuAD 2.0 challenge, showcased promising results. In terms of the Exact Match (EM) score, the model achieved a score of 50.07, indicating that in 50.07% of the cases, the model's answer matched the human answer. Similarly, the model's F1 score, which measures the average overlap between the predicted answers and the ground truth, was also 50.07. This indicates a balanced performance in terms of both precision and recall.

4.2 Discussion

The results from this project provide valuable insights into the complexities and challenges associated with the SQuAD 2.0 dataset, particularly given its inclusion of unanswerable questions. Our study, focusing on the ELECTRA model, demonstrated its efficacy in addressing these challenges. However, the gap between the model's performance and human-level

accuracy underscores the nuanced difficulties inherent in natural language processing. This divergence highlights the need for continued advancements in developing models capable of more accurately interpreting and responding to the intricacies of human language.

The performance of the ELECTRA model in this context underscores its strengths and limitations in dealing with complex question-answering tasks. While the model shows promise in understanding and processing language, the results also point to areas where further improvement is necessary, particularly in handling questions deliberately designed to be unanswerable.

4.3 Limitations and Future Work

While providing valuable insights, the study has its limitations, mainly its focus on a single model (ELECTRA) and its application to one specific dataset (SQuAD 2.0). Future research could expand on this by exploring a broader range of models or applying the ELECTRA model to different datasets to assess its versatility and adaptability. Such expansion would help understand the model's generalizability and performance across various contexts.

Furthermore, an in-depth exploration into optimizing the ELECTRA model for this specific task could be beneficial. This could involve fine-tuning the model's architecture, exploring different training strategies, or experimenting with various parameter settings. Investigating the model's performance in other languages or domain-specific contexts could provide more comprehensive insights.

Looking ahead, there are several promising directions for further research. For instance, assessing the performance of the ELECTRA model in real-world applications, exploring its adaptability to different NLP tasks, or evaluating its scalability and efficiency are all areas that warrant further exploration. As the field of NLP continues to advance rapidly, such investigations are critical for advancing our understanding and capabilities in language processing and comprehension.

5. Summary and Conclusions

This project centered on developing a sophisticated question-answering (QA) system using the ELECTRA model, significantly contributing to the Natural Language Processing (NLP) field. Our focus was to build a system that can adeptly interpret and respond to questions in natural language, drawing upon the insights from "Speech and Language Processing" by Daniel Jurafsky and James H. Martin and leveraging the SQuAD 2.0 dataset. The ELECTRA model, known for its efficiency in learning from all input tokens and for its replaced token detection methodology, was fine-tuned and tested for its proficiency in QA tasks. The results indicated a commendable performance with an Exact Match score and an F1 score of 50.07, reflecting the model's capability to align closely with human accuracy in specific scenarios. Despite these promising results, the project highlighted the challenges in bridging the gap between machine and human comprehension, especially in handling unanswerable questions inherent in SQuAD 2.0. The experience gained from this project underscores the potential and limitations of current NLP technologies. It opens avenues for future research, particularly in optimizing models like ELECTRA for diverse and more complex NLP tasks.

6. Calculation

The number of lines I wrote: 576
The number of lines I borrowed: 490
The number of lines I modified: 70
(490 - 70) / (490 + 576) * 100 = 39.39%

References

[1] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training.

[2] Clark, K., Luong, M.-T., Brain, G., Le Google Brain, Q., & Manning, C. (2020). ELECTRA: PRE-TRAINING TEXT ENCODERS AS DISCRIMINATORS RATHER THAN GENERATORS. https://arxiv.org/pdf/2003.10555.pdf

[3] ELECTRA. (n.d.). Huggingface. co. https://huggingface.co/docs/transformers/model_doc/electra

[4] Elias, G. (2020, December 29). Tutorial: How to pre-train ELECTRA for Spanish from Scratch. Skim AI. https://skimai.com/how-to-train-electra-language-model-for-spanish/

[5] Hu, Edward J., et al. “Lora: Low-rank adaptation of large language models.” arXiv preprint arXiv:2106.09685 (2021).

[6] Jurafsky, D., & Martin, J. H. (2023). Question answering and information retrieval. In Speech and Language Processing (3rd ed. draft). [Jan 7, 2023 draft]. https://web.stanford.edu/~jurafsky/slp3/

[7] Morzkowski, K., Sun, L., & Hoovestol, P. (n.d.). SQuAD 2.0 Project Report. https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/reports/default/15785042 pdf

[8] Rajpurkar, P., Jia, R., & Liang, P. (n.d.). Know What You Don’t Know: Unanswerable Questions for SQuAD. https://arxiv.org/pdf/1806.03822.pdf

[9] Question answering. (n.d.). Huggingface. co. https://huggingface.co/docs/transformers/tasks/question_answering

[10] Transformers.modeling_electra — transformers 3.0.2 documentation. (n.d.). Huggingface. co. Retrieved December 10, 2023, from https://huggingface.co/transformers/v3.0.2/_modules/transformers/modeling_electra.html #ElectraForQuestionAnswering

[11] What is Question Answering? - Hugging Face. (2001, September 26). https://huggingface.co/tasks/question-answering