TIAA KTP ASSESSMENT

To perform fine-tuning on the t5-efficient-tiny model using the alpaca-gpt4 dataset for a question-answering model.

Mohammed Talib

University of Essex

Interview Assessment

Contents

1 Introduction

This report covers the findings and approach used to complete the technical assignment associated with the position of a KTP (Knowledge Transfer Partnership) partner for the TIAA Ltd and the University of Essex collaborative project.

2 Dataset

The Alpaca-GPT4 dataset is a collection of 52,000 instruction-following prompts and their corresponding answers, generated by GPT-4. The dataset is designed to be used for training question answering models.
Data format: The dataset is in JSON format. Each entry in the dataset consists of an instruction, an input, and an output. The instruction is a natural language description of what the model should do. The input is a piece of text that the model should use to generate the output. The output is the answer to the question that is implied by the instruction.
Size: The dataset is 48MB in size.

The "alpaca-gpt4" dataset's method of creation is what makes it unique. The "alpaca-gpt4" dataset makes use of GPT-4 as opposed to the original Alpaca dataset's text-davinci-003 for rapid completions, yielding results that are of higher quality and depth.

3 Splitting the data

Split the dataset into train, validation, and test set As a common practice, we split the dataset into:

Training set: Data used for training the model parameters. Validation set: Data used for hyperparameter tuning or early stopping to avoid overfitting. Test set: Data used for checking what performance we can expect on new data.

The choice of the sizes of the three datasets usually depends on many factors, such as the size of the whole dataset, how much time you want to spend training or evaluating your model, and how precise you want your model evaluations to be. Since we are training a quite big model on a free GPU from Kaggle, I decided to use small validation and test sets to speed things up. Despite this, we will see that the final model will be able to produce good answers.

4 Pre-Processing the data

Each input text receives the prefix to identify the task's nature. In this instance, it is set to "answer_question:" to give the model background information regarding the particular task it is carrying out. The input and target sequences' maximum lengths are set in these lines. They aid in preventing lengthy input and target sequences, which might cause memory problems during training. Sequences that are longer will be padded or shortened as necessary.

The input text is first preprocessed using the clean_text function. The text is tokenized into sentences, with sentences further divided based on newline characters, and empty or sentences without punctuation marks are filtered out. This text-cleaning phase guarantees that the input is well-structured and reduces text noise.

Why T5TokenizerFast over T5Tokenizer?

T5TokenizerFast is substantially more fast than T5Tokenizer thanks to its speed optimization. When working with huge datasets and during the training of models, when tokenization is a frequent operation, this might be quite important. T5TokenizerFast is made to use as little memory as possible. It employs an implementation that is more memory-friendly and is therefore appropriate for processing vast volumes of text without incurring memory problems.

T5TokenizerFast has identical capabilities to T5Tokenizer (such as tokenization, truncation, and padding), but it does so more quickly.

5 Evaluation Metrics for Fine-Tuning Question Answering Models

In the context of fine-tuning question answering models, the selection of appropriate evaluation metrics is crucial for assessing their performance effectively. This section discusses three distinct evaluation metrics that are commonly used in this domain: ROUGE (specifically ROUGE-1, ROUGE-2, and ROUGE-L), Exact Match, and BLEU. We will elaborate on how these metrics function and justify their relevance in evaluating question answering models.

5.1 ROUGE Metrics (ROUGE-1, ROUGE-2, and ROUGE-L)

The ROUGE metrics, including ROUGE-1, ROUGE-2, and ROUGE-L, are widely employed in natural language processing tasks to evaluate the quality of generated text. These metrics are primarily designed to measure the overlap and similarity between the generated answers and reference answers.

ROUGE metrics are well-suited for question answering tasks due to the following reasons:

- Precision in measuring overlap: ROUGE metrics emphasize precision by assessing the overlap of words or n-grams. In question answering, precision is vital as models should generate concise and accurate answers to questions.
- Contextual relevance: ROUGE-L, in particular, accounts for the order of words in answers, ensuring that the generated answers maintain contextual relevance to the reference answers.
- Customizable: ROUGE metrics offer flexibility by allowing the choice of n-grams. Depending on the specific requirements of the task, one can select ROUGE-1, ROUGE-2, or ROUGE-L to focus on different levels of linguistic analysis.

5.2 BLEU (Bilingual Evaluation Understudy)

BLEU is a metric originally designed for machine translation but is also applied to question answering. It measures the similarity between the generated answer and the reference answers by computing the precision of n-grams (typically up to 4-grams).

BLEU is employed in question answering for the following reasons:

- N-gram precision: BLEU evaluates the precision of n-grams, allowing a more flexible assessment of answer quality compared to exact match metrics. This flexibility is especially useful for assessing partial correctness.
- Multiple reference answers: BLEU accommodates multiple reference answers, which is common in question answering tasks where different phrasings of correct answers exist.

6 T5 Architecture for Question Answering

The Text-to-Text Transfer Transformer (T5) architecture has gained prominence in natural language processing due to its impressive performance across various tasks. The T5 architecture is rooted in the Transformer model, a deep learning architecture that revolutionized NLP. T5 inherits the core components of the Transformer, which include multi-head self-attention mechanisms, feed-forward neural networks, and layer normalization. These components collectively enable T5 to capture intricate linguistic patterns, dependencies, and long-range relationships within textual data.

A distinguishing feature of T5 is its innovative approach to framing NLP tasks as "text-to-text" problems. In this framework, both the input and output are treated as text sequences. This unification simplifies the design and training of models for a wide array of NLP tasks, including translation, summarization, and question answering. For question answering, the question and context are represented as text, and the answer is generated as text, fostering a unified, end-to-end solution.

6.1 Encoder-Decoder Architecture

T5 follows an encoder-decoder architecture, where the encoder processes the input text, and the decoder generates the output text. In question answering, the input typically consists of a question and a context, while the output is the answer. The encoder efficiently encodes both the question and context, while the decoder generates coherent answers based on the encoded information.

6.2 Advantages for Question Answering

T5 architecture offers several advantages for question answering tasks:

- **Contextual Understanding:** T5's pre-training equips it with a deep understanding of context, which is pivotal for accurate question answering.
- **Text Generation:** T5 can generate human-readable answers in natural language, ensuring that the responses are coherent and interpretable.
- **Transfer Learning:** The pre-trained T5 model serves as a strong foundation for fine-tuning, reducing the need for extensive task-specific labeled data.
- **Unified Framework:** T5's text-to-text framework provides a unified approach to NLP tasks, simplifying model development and deployment.

7 Results

We have trained two models on the alpaca-gpt4 data:

1. A model without Seq2SeqTrainer and with T5forQuestionAnswering.
2. A model with Seq2SeqTrainer and with T5forConditionalGeneration.

The first model achieved a train loss of 4.44 and a validation loss of 4.00. The second model achieved a train loss of 1.16 and a validation loss of 1.06.

The first model predicted the following answer for the question "Write a story on moon":

> **Model 1:** The Moonś Journey: A Memoir of a Man on the Edge of
>
> **Model 2:** Once upon a time, in a circle of dawn when the moon suddenly arrived, the moon was in and the moon was a mystical and powerful moon of the moon.

8 Conclusion

The second model (with Seq2SeqTrainer and T5forConditionalGenerator) achieved a better validation loss than the first model. This suggests that it may be a better model for generating text. However, it is important to note that the two models were trained on different metrics, so it is difficult to make a direct comparison.

The first model's prediction for the question is not very informative. The second model's prediction for the question is more creative, and generated more words. Though, T5forQuestionAnswring is specifically designed for Question Answering task, still we did not get good output from the model which suggests more experimentation is required.

Overall, both models need further improvement. We can try training them on more data, using different training parameters, or exploring different model architectures. We can try exploring other metric of evaluation as well.