CS-552 Project Plan

Antoine Magron, Mathieu Desponds, Mekhron Bobokhonov

May 2023

The CS-552 project consists of building an assistant similar to ChatGPT to answer student's question related to EPFL courses. The project consist of two sequential parts :

- Building a reward model that will learn what a good answer consists of
- Fine tune a Large Language Model (LLM) using the obtained reward model to turn it into the EPFL assistant

We will describe the two parts here independently, the final Figure 1 contains the entire structure of the project.

1 Reward Model

We will start by training a reward model. This model will give a score to a demonstration produced by our assistant. The model will be trained to give better score to good demonstration and bad score to poor ones.

1.1 Dataset

This part is described in the top part of Figure 1.

The model will essential tries to maximize the difference in reward for a good answer Y_+ and a bad answer $Y_-^{(i)}$. We then need to build a dataset consisting of tuples $(Q, Y_+, Y_-^{(i)})$ where Q is a question. To do so, we will use the set of question associated given during **Milestone 1**. We will start by filtering out answers with a low confidence. This will give a set of the form (Q, Y_+) considering the answers as the good answer. If the number of questions is less than 1k, we will collect questions and answers from exams of EPFL courses from previous years. Such data are available on the webpage of courses or on GitHub.

Then we will use again the provided `GPTWrapper` to query ChatGPT purposely bad prompts to produce a wide variety of bad answers. This method will yield a set $\{Y_-^{(i)}\}_{i=1}^k$ for each question. These prompts could be :

- "Give the final answer to this question"
- "Give the answer to the question and explain why it is the good answer"
- "Develop each of the answers and explain why they are true or false and finally give the right answer"

Using this, we would have a wide range of variety in so-called "bad answers". This will also help us augment our dataset in order to have a dataset big enough to train our reward model.

We thus end up with a dataset of tuples $(Q, Y_+, Y_-^{(i)})$.

1.2 Model

For our reward model, we want to use top performing transformer model that can take text as input and output a number. For this purpose, we will try to use RoBERTa-large. However, if it turns out to be too demanding in terms of resources, we might use RoBERTa-base.

This model can yield good performance on regression task by embedding properly the semantic of the question.

1.3 Fine-tuning

This part is described in the middle part of Figure 1. As said previously, the model will be trained to maximize the difference in reward between bad and good answered. The usual loss function for this task is :

$$\mathcal{L}_{RL} = -log(\sigma(R(Y_+) - R(Y_-^{(i)})))$$

Where $R(Y)$ is the model output for answer Y. We will train the model this way using a 80% split of the created dataset. The rest will be used for evaluation. For this part, we will calculate $(R(Y_+), R(Y_-^{(1)}), ..., R(Y_-^{(k)}))$ where answers worsen with k. We can then apply standard metrics of ranking (DCG NDCG, `Precision@n`, `NDCG@n`, etc.)

2 Final Model

In this part, we use the previously trained Reward Model to train a LLM to become an EPFL assistant. This part is described in the bottom third of Figure 1.

2.1 Dataset

To train the model, we will use the same dataset as built before, but we don't need the bad answers $Y_-^{(i)}$ and keep only a set of pair (Q, Y_+).

2.2 Model

The goal of this part is to start from a basic generative model, to input a question from our dataset and train it to output an answer that has the same structure as provided Y_+.

We simply need to choose a LLM for text generation. We chose to reduce this down to either OpenAI's GPT-3 or Google T5. Comparing OpenAI's GPT-3 and Google's Flan-T5 is not a straightforward task, as they are both state-of-the-art language models but serve different purposes and have different strengths.

GPT-3 is a massive, generative language model with over 175 billion parameters. It can perform a variety of tasks such as language translation, question-answering, and text completion with high accuracy. However, it requires a lot of computational resources to run and is not optimized for specific NLP tasks. Flan-T5, on the other hand, is a smaller, fine-tuned version of the T5

architecture, designed for specific NLP tasks such as text classification, summarization, and translation. It is more computationally efficient than GPT-3 and can perform these tasks with high accuracy, but it may not be as versatile as GPT-3 in generating human-like text.

2.3 Fine-tuning

We now fine-tune the previously chosen model to answer the EPFL's student questions. For the training, we will give the model one of the dataset's question Q for which we know the gold reference Y_+. The model will return its answer : $\hat{A}$. We can now proceed in two possible ways. Either, we use the usual LLM fine-tune and thus using the Cross Entropy Loss between $\hat{A}$ and Y_+ to train the model. Or, we use the state-of-the-art method of Reinforcement Learning (RL) to build train the model better. This method change the loss function to account for the reward of the previous build reward model. The reward model has learned to give a high score to good answers and low score to bad ones. The loss function will then be :

$$\mathcal{L} = -R(\hat{A})CE(\hat{A}, A) = -R(\hat{A}) \sum log(\mathbf{P}(\hat{a}_i|q^*; \{\hat{a}_j\}_{j=1}^{i-1}))$$

Where $R(\hat{A})$ is the reward attributed to the generated answer and CE is the Cross entropy between the generated answer and the gold reference.

To evaluate our model, we will use our reward model to measure the quality of the text generated by our fine-tuned model on a previously selected test set. We can use standard regression metrics (MSE, MAE, etc.) to compare the rewards of generated answers and gold answers.

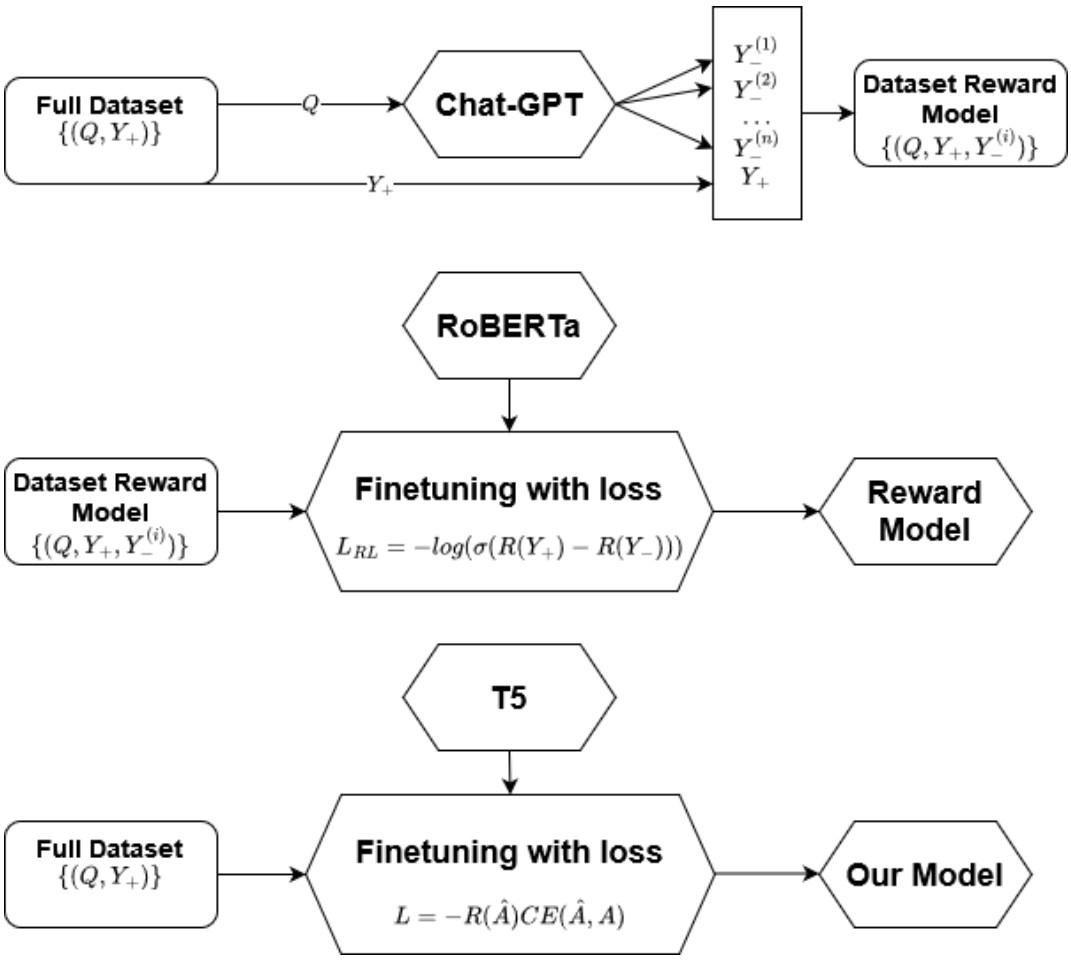

Figure 1: Project Plan