# EPFLearn: AI Student Mentor

**Antoine Bonnet**
antoine.bonnet@epfl.ch

**Silvia Romanato**
silvia.romanato@epfl.ch

**Alexander Sternfeld**
alexander.sternfeld@epfl.ch

**Team**: `syntax-sorcerers`
School of Computer Science
Ecole Polytechnique Fédérale de Lausanne

## Abstract

EPFLearn is a didactic chatbot designed to answer students' questions on specialized course material. Built on top of Google's pre-trained T5 Transformer as its foundation model, it underwent continual pre-training on the StackOverflow dataset and further fine-tuning on EPFL's new NLP4Education dataset. EPFLearn achieves promising results on closed-book question-answering tasks with a BLEU score of 40.89 and ROUGE scores of 67.47 (ROUGE-1), 43.45 (ROUGE-2), and 63.71 (ROUGE-L). Additionally, qualitative evaluation confirms EPFLearn's capacity to assimilate complex technical knowledge and provide accurate and reliable answers across a wide range of educational topics.

## 1 Introduction

**Educational chatbot**. This paper details the development of EPFLearn, an educational chatbot designed to address students' queries regarding courses offered at École Polytechnique Fédérale de Lausanne (EPFL). The primary objective of this research is to enhance the learning experience for EPFL students by providing them with a reliable and efficient tool for course-related inquiries. By employing EPFLearn, students can receive prompt and accurate responses to their questions, bypassing the need to consult professors, search through course materials, or access library resources. This not only saves time but also offers a personalized and readily available source of information, catering to the individual needs of students.

**Language model**. To achieve this, the chatbot was created by fine-tuning Google's T5 770 million parameters language model, enabling it to perform closed-book question answering. The training mixture incorporates publicly available question-and-answer data from the StackOverflow forum as well as EPFL's NLP4Education dataset, tailored specifically for educational content. This process involves pre-processing the relevant data, training the model on them, optimizing the model's response generation capabilities and finally evaluating the performance. Additionally, considerations such as model architecture, experimental methodology, and design choices are discussed.

**Structure**. The paper is organized as follows: Section 2 provides an overview of the existing literature on fine-tuning large language models (LLMs) and highlights the unique challenges addressed in this project. Section 3 details the methodology employed for training EPFLearn and fine-tuning procedures. Section 4 presents the data selection and preprocessing techniques. Section 5 discusses the experimental setup, the validation metrics and the performance analysis. Finally, Section 6 and Section 7 concludes the paper by analyzes and interprets the findings, discussing the implications and limitations of the EPFLearn and highlighting its significance in the context of educational chatbot research, and suggesting future research directions.

Submitted to *CS-552: Modern Natural Language Processing* at EPFL in June 2023.

## 2 Related Work

**Fine-tuning**. The idea of fine-tuning pre-trained generative models has gained significant traction in the field of natural language processing. In this section, we discuss the existing body of research related to fine-tuning pre-trained language models, along with the primary challenges associated with this approach.

**Pre-trained models**. Considerable efforts have been dedicated to the pre-training of language models (LMs) on vast amounts of data. Currently, a wide array of pre-trained Transformer models are available. Our focus will revolve around encoder-decoder models developed by Google, including BERT [1], its enhanced version RoBERTa [2] and the text-to-text T5 [3]. Although these models showcase remarkable out-of-the-box performance, fine-tuning them for specific tasks allows for adaptation to more specialized domains or tasks with limited labeled data.

**Challenge**. Let us discuss one challenge that has surfaced in the context of fine-tuning LLMs. When the available labeled data for fine-tuning is comparatively small in relation to the pre-training data, it may be challenging for the model to properly absorb the new data. Aggressively fine-tuning with limited data poses the risk of overfitting to the training data [4], which degrades the model's generalization abilities. One solution to this problem would be to gather more data, however, this may not be feasible. Our research relates to a second option, specifically *continual pre-training*.

**Continual pre-training**. The objective of continual pre-training is to tailor a language model (LM) to a specific domain, thereby enhancing its performance on the target task [5]. In this scenario we want to fine tune the model in order to provide answers to question about a specific domain and knowledge required. Initially, the model is pre-trained on a broad language corpus, establishing a foundation of language knowledge. Subsequently, the model can be further refined by training it on a dataset specifically curated on a Q&A task on a specific domain. The key advantage of continual pre-training, specifically of domain-related continual pre-training, is that it reduces the requirement for task-specific data, instead relying on data within the same domain, which is typically more readily accessible and improve end-task performances.

## 3 Approach

### 3.1 Language model

**Foundation model.** We selected Google's Text-To-Text Transfer Transformer (T5) model as the foundational language model for our chatbot. Specificially, we used its large version T5 Large which comprises an encoder-decoder architecture with 770 million parameters. The general goal of the T5 generative model is to provide a versatile and robust solution for a wide range of NLP tasks. Rather than designing separate models for individual NLP tasks such as translation, summarization, question answering and text classification, T5 takes a unified approach by treating all these tasks as text generation problems. By doing so, T5 aims to simplify the model architecture, promote transfer learning, and improve overall performance across various NLP tasks [3].

**Encoder-Decoder architecture.** The T5 model architecture is made of two parts, the encoder and decoder. The encoder consists of 24 successive self-attention layers designed to extract relevant contextual hierarchical information from the prompt. It learns to abstract complex patterns and relationships present in the input text into a 1,024-dimensional representation called a context vector. The decoder also consists of 24 self-attention layers, but now with cross-attention layers to attend to different parts of the encoded information so as to generate a coherent output sequence. To do so, it uses an auto-regressive approach by which it successively predicts the next token given the previously generated tokens. The decoder plays a crucial role in generating accurate and contextually appropriate answers during both pre-training and fine-tuning phases.

**Pre-training**. During its pre-training phase, the T5 model was trained with a masked-language modelling (MLM) objective on the Colossal Clean Crawled Corpus (C4), an extensive dataset spanning 700GB of text in multiple languages including English, German, French and Romanian. Its tokenizer vocabulary encompasses 32,128 tokens. Its multilingual abilities were an important motivation in choosing T5, considering the presence of both English and French content within the NLP4Education dataset. In its raw, pre-trained state, T5 is able to produce coherent responses to general queries, but it fails when prompted on specialized knowledge, technical course content or mathematical questions.

**Fine-tuning.** During fine-tuning, the model was trained using high-quality question-answer pairs from both sources (see Section 4.1). When fed a question, the model was trained to replicate the target answer using the cross-entropy loss, commonly adopted for language generation tasks. This loss function measures the dissimilarity between the predicted output and the true answer in the target sequence. It guides the model to generate more accurate and coherent outputs.

### 3.2 Reward Model

**Classification model**. To address the challenge of evaluating the quality of our chatbot's generated responses, we trained a reward model with the aim of assigning a scalar reward to a question-answer pair. We framed this task as a classification problem by training a classifier model to differentiate 'correct' from 'incorrect' QA pairs. To do so, we used the pre-trained encoder of the RoBERTa [2] base model with a custom classification head on top of the special [CLS] token embedding. This Transformer model is composed of a stack of 12 identical layers with 12 attention heads each which encode the input text into a 768-dimensional embedding vector. This base model was pre-trained with the MLM objective on 160GB of English data.

**Classification head**. The classification head applies on the 768-dimensional embedding and a first linear layer with sigmoid and 10% dropout to obtain a hidden layer of equal dimension 768. A second linear layer is added with output dimension 2 (one for each label; 0 for 'incorrect', 1 for 'correct') and softmax activation. Each class therefore receives an associated output logit, which can be interpreted a scalar reward in $[0, 1]$ range associated with the class probability of the input text.

**Fine-tuning**. During training, the reward model was fed concatenated question-answer pairs and trained to differentiate correct from incorrect answers. Due to major class imbalances in the datasets (most questions had fewer correct answers than incorrect answers), we used a weighted cross entropy loss by adjusting the loss based on the class frequency.

## 4 Experiments

### 4.1 Data

Fine-tuning a language model to respond accurately with specialized knowledge on technical content requires examples of high-quality answers for the model to absorb and learn to reproduce. Training the reward model to differentiate correct from incorrect QA pairs requires contrastive samples of both 'good' and 'bad' examples to learn from. We trained both models using pre-processed data from two sources. Both reward and generative models were trained first on the StackOverflow dataset, then fine-tuned on the NLP4Education dataset.

#### 4.1.1 EPFL's NLP4Education dataset

**ChatGPT interactions**. Distilled generations from 100B-parameter scale LLMs form a valuable source of data for fine-tuning smaller specialized models. Following this line of thought, the NLP4Education project at École Polytechnique Fédérale Lausanne (EPFL) provided us with 4450 unique questions and their official solutions derived from official course content in both French and English. To extend this course material with high-quality distilled data, students were each assigned a subset of 100 questions and were asked to manually collect interactions with ChatGPT. Every question was assigned to 3 different students and each of them was individually tasked with finding the best prompting strategy to answer these 100 questions. Students were also asked to assign a confidence score to the best answer provided by ChatGPT indicating its estimated answer quality on a scale from 1 to 5. Table 2 shows the distribution of the confidence scores and the distribution of the number of collected answers per question.

**Pre-processing**: In order to train a classifier reward model, it is essential to perform contrastive learning to distinguish between 'good' and 'bad' samples. In our specific case, each sample consists of a unique question-answer pair. For this purpose, we labeled official solutions and one-shot ChatGPT interactions with a confidence score of 5 as positive samples, while interactions with confidence scores between 1 and 3 were considered as negative responses. On the other hand, when it comes to fine-tuning a language model, it is crucial to have only high-quality 'good' samples in order to effectively learn the replication process. To achieve this, we retained only the official answers and the generated one-shot answers labeled with confidence levels 4 and 5. Consequently, the resulting dataset consists of 2,284 questions and 6,955 answers for reward modeling, and 4450 questions and 7,872 answers for language modeling.

#### 4.1.2 StackOverflow dataset

**Public Q&A forums**. The open-source StackOverflow dataset was scraped from a popular Q&A forum containing over 27 million answers on a wide range of topics. The dataset contains over 27 million answers across various topics. Each question posted on the forum can have at most one "accepted" answer, designated by the original poster, which serves as an indicator of answer quality. Although the number of upvotes could have been used as well to rate answers, we used answer acceptance as the only indicator of answer quality.

**Pre-processing**. To train our specialized chatbot, we chose questions from the StackOverflow dataset on topics which were aligned with the technical nature of the course taught at EPFL. The chosen topics were the following: computer science, computer science theory, data science, physics, chemistry, engineering, mechanics, software engineering, quantitative analysis and mathematics.

**Reward modelling**. For the reward model, we restricted our training data to the first 3 topics. We only retained questions with at least one (and up to 30) non-accepted answers per question and used weighted cross-entropy to balance out the loss across classes. QA pairs were concatenated into a single string formatted as *Human: [Question] Assistant: [Answer]*. We obtained a total of 8,086 questions and 22,441 corresponding answers for the reward model.

**Text generation**. To train the generative model, we restricted the fine-tuning mixture to only 'positive' interactions by keeping only questions with an accepted answer. We included all 10 topics to ensure a broad knowledge base, but filtered answers with a minimal number of up-votes to ensure answer quality using different thresholds to balance out the topic distribution (see Table 3) for a total of 95,041 unique QA pairs.

### 4.2 Evaluation Method

**Text generation metrics**. Evaluating the quality of generated text is not straightforward; a common proxy is to assess the similarity of the model's generated answers to the actual target answers. To do so, we rely on BLEU [6] and ROUGE [7] scores, two popular metrics used to in natural language generation. ROUGE scores measure the recall of n-grams between the predicted and target text; ROUGE-1 for unigrams, ROUGE-2 for bigrams, and ROUGE-L for the longest common subsequence. On the other hand, the BLEU score measures the precision of matching n-grams between the generated and target answers. Both metrics provide valuable insights into the degree of textual overlap between reference and prediction and offer valuable albeit limited insights on the quality and accuracy of generated answers across different models.

**Qualitative evaluation**. To provide a comprehensive evaluation of the models, we also assess the textual quality of generated answers. Through manual inspection of answers given to randomly selected prompts, we assess factors such as correctness, coherence, knowledge on technical content, diversity of vocabulary and fluency. This qualitative analysis provides valuable insights into the strengths and weaknesses of the model beyond the quantitative metrics, offering a holistic evaluation of their ability to produce valuable educational content.

**Baselines**. To evaluate the impact of fine-tuning on the model's ability to generate meaningful text, we establish a baseline using a similar text to text model. However, the pre-trained T5 model could not be used as a baseline because it was only trained on open-book QA, where the model extracts the answer from a given context. As an alternative, we use the `T5-large-ssm-nq` model as baseline. It is a pre-trained T5 model which has undergone additional pre-training on Wikipedia and fine-tuning on Natural Questions for the closed-book QA task [3]. We compare this baseline against the performance of our fine-tuned models, including an intermediary checkpoint (EPFLearn-V1) trained on StackOverflow and the final checkpoint (EPFLearn-V2) trained on both StackOverflow and EPFL datasets. Through this comparison, we assess the effectiveness of our fine-tuning approach and measure improvements resulting from specialized training on educational data.

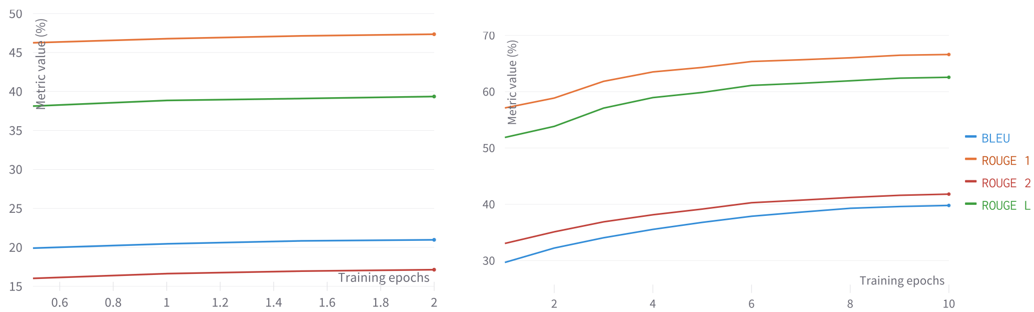


Figure 1: Evaluated ROUGE and BLEU metrics during fine-tuning on StackOverflow dataset (left) and EPFL dataset (right) on their respective validation sets.

### 4.3 Experimental details

**Validation sets**. During the training process, each dataset was split into training, validation and test sets with a ratio of 80%, 10% and 10% respectively. To prevent data leakage, questions were grouped together in the same split. After each training epoch, we evaluated the model on the validation set using the perplexity, ROUGE and BLEU metrics on text generated using the greedy maximum likelihood estimate (MLE). At the end of training, we evaluated the model's performance on the reserved test set with our custom decoding configuration.

**Training parameters**. Our generative model was trained and evaluated on batches of 8 question-answer pairs, where each textual samples was truncated and padded to 128 tokens. Training occurred on a single GPU with early stopping, gradient checkpoint and 4 gradient accumulation steps for computational optimization reasons. The T5 model was first trained on the StackOverflow dataset with an initial learning rate of $2 \cdot 10^{-4}$ decaying linearly over two epochs of training. Note that to save computational resources, the dataset was randomly subsampled by 50%. An intermediary checkpoint (EPFLearn-V1) was then saved. Further fine-tuning on the EPFL dataset ensued with a smaller initial learning rate of $5 \cdot 10^{-5}$ with linear decay over 10 epochs. More careful optimization was adopted for the final round of training to avoid unlearning previously acquired knowledge. The BLEU and ROUGE validation curves obtained during training on the EPFL and StackOverflow dataset are shown in Figure 1.

**Generation parameters**. Once our model was fine-tuned, we evaluated its ability to generate coherent answers using a carefully tuned customized decoding strategy. Beam search with sampling was used with 8 beams on the top-50 highest probability tokens with cumulative probability 0.95, with a temperature of 1.2 to encourage generation diversity and a repetition penalty of 1.3 to discourage excessive repetition in the generated text. We manually forbid the model to generate links, as we found that it had a tendency to produce random links because of its StackOverflow training.[1]

### 4.4 Results

**Comparative analysis**. The final evaluation on the reserved test sets of our models EPFLearn-V1, EPFLearn-V2 and the baseline mdoel are presented in Table 1. We highlight that, even if the baseline was trained on closed book question answering, it only produced short answers without explanations, which might bias the resulting evaluation metrics even if the short answer matches the reference. Examples of model outputs are shown in Table 2.

**Performance on StackOverflow dataset**. On the StackOverflow test set, all metrics for the intermediary version indicate a moderate degree of similarity between the generated responses and the reference answers. The EPFLearn-V2 performance is slightly lower than EPFLearn-V1 over all metrics on StackOverflow. This was expected, as some degree of knowledge leaks when incorporating new knowledge into the model. Given that the decline in performance is only around 2 points,

[1]Note that extensive details on the reward model's training procedure, experimental results and analysis are provided in Section 7.2 of the Appendix.

Table 1: Evaluation metrics of EPFLearn-V1, EPFLearn-V2 and T5-large-ssm-nq baseline

| | StackOverflow dataset | | | EPFL dataset | | |
|---|---|---|---|---|---|---|
| Metric | Baseline | EPFLearn-V1 | EPFLearn-V2 | Baseline | EPFLearn-V1 | EPFLearn-V2 |
| BLEU | 1.62 | 21.01 | 19.79 | 1.72 | 23.38 | 40.88 |
| ROUGE-1 | 17.0 | 47.46 | 45.40 | 18.9 | 51.21 | 67.47 |
| ROUGE-2 | 1.60 | 17.05 | 15.53 | 3.30 | 26.89 | 43.45 |
| ROUGE-L | 13.27 | 39.43 | 37.10 | 15.3 | 45.54 | 63.7 |

Table 2: Comparison of generated outputs by EPFLearn-V1, EPFLearn-V2 and T5-large-ssm-nq

| | |
|---|---|
| | **Example 1** |
| **Question** | What is the difference between genetics and epigenetics? Select one of the following choices:<br>1. Genetic characteristics are heritable, but not epigenetic one<br>2. Epigenetic characteristics vary between cell types, but genetic characteristics do not<br>3. Differences in epigenetics do not influence gene expression whereas genetic variability does<br>4. All of the above. |
| `T5-large-ssm-nq` | Genetic variability within species |
| `EPFLearn-V1` | Epigenetics and genetics are the two different ways of looking at gene expression. Genetics is the study of the mechanism of how genes are regulated, and epigenetics the study of how genes change over time. It's the same as epigenetic research. The difference between genetics and epigenomics is that epigenetic variation does not affect gene expression; whereas genetic variability does. Genetics and epigeneticism are the two ways of studying gene expression: if you've got a gene that's regulated, you'll probably want to understand what it means to say |
| `EPFLearn-V2` | The difference between genetics and epigenetics is that epigenetic changes can influence gene expression, but genetic variability can influence gene transcription. Genetic variation is the ability of a cell to express or suppress certain genes, but it is not heritable. Epigenetic changes are the changes in DNA methylation patterns that can alter gene expression. Epigenes can affect gene expression by altering the DNA methyl groups in a cell, but they are not directly related to genetic variation. Genetic variability is the ability for a cell or tissue type to express a gene without changing its DNA methylated with |
| **Solution** | Epigenetic characteristics vary between cell types, but genetic characteristics do not |
| | **Example 2** |
| **Question** | What is the general relation between the entanglement of a state and the violation of a Bell inequality? Select one of the following answers:<br>1. Entanglement is necessary but not sufficient for the violation of a Bell inequality,<br>2. Entanglement is sufficient but not necessary for the violation of a Bell inequality,<br>3. Entanglement is necessary and sufficient for the violation of a Bell inequality |
| `T5-large-ssm-nq` | Entanglement |
| `EPFLearn-V1` | Suppose that x_1, y_2, z_3, z_4,..., z_7, z.d are two different states, and x_1 < y_2. Then x_i ; y > z_6 . . . x_i is a state that is entangled with y_1 - y_4 - z_5, which is the same as x_2 - |
| `EPFLearn-V2` | Entanglement is necessary but not sufficient for the violation of a Bell inequality. To violate a Bell equality, we need to show that there is a state with a higher probability of being entangled with another state than the one in which we are entanglement. This means that we need to find the state with the highest probability of being in a state of entangled states. If there is entanglement between two states with the same probability of being together, then we have violated a Bell equation. However, if there is no entanglements between two states, then |
| **Solution** | Entanglement is sufficient but not necessary for the violation of a Bell inequality |

evidently catastrophic forgetting did not occur, and most of the knowledge acquired during continual pre-training on the StackOverflow dataset was retained by the final chatbot.

**Performance on EPFL dataset**. The chatbot's performance on the NLP4Education dataset was our primary concern. We observe that the final chatbot significantly outperforms the intermediary one across all evaluation metrics. Indeed, EPFLearn-V2 exhibits much higher scores compared to the intermediary model with increases between 16.3 and 18.2 points, indicating a greater degree of generation accuracy. These results show that further fine-tuning the model on the EPFL dataset enhances its ability to generate relevant and accurate responses specifically tailored to EPFL students.

## 5 Analysis

**Qualitative evaluation**. Following the completion of the quantitative evaluation in Section 4, which provides some insights into the quality of the generated answers, we proceed with a comprehensive qualitative analysis where we compare outputs by the final version of EPFLearn, the intermediate version, and the baseline model on two questions from the EPFL NLP4Education test set, as shown in Table 2. We assess the technical accuracy and coherence of the answer, but also the completeness of the explanations accompanying the correct answers.

**Baseline**. Evidently, the baseline T5-large-ssm-nq model struggles to generate comprehensive answers. These responses often contain repetitive but contextually relevant terms, failing to address the question adequately and typically generating short answers. This may be caused by the training objective used by Google, which might have favored shorter answers. However, in the case of an educational chatbot, producing didactic explanations along the correct answer is a desired part of the model's response and the baseline model fails crudely in this regard.

**First example**. When comparing the performance of EPFLearn V1 and V2 on Question 1, further insights emerge. Both models generate pertinent text that is aligned with the adopted vocabulary. However, the intermediate version offers an answer that is coherent with the question but lacks a direct response to the multiple-choice task. Instead, it provides a more generic explanation with redundant concepts and incorrect statements. In contrast, EPFLearn-V2 provides a correct answer, uses sophisticated vocabulary and shows advanced conceptual understanding. Nevertheless, it falls short in presenting the user with a clear choice among the provided options.

**Second example**. The second example provided additional insights into the model's performance. The answer generated by EPFLearn-V1 was incomplete and did not directly respond to the question. Moreover, it did not offer a choice among the available options. In contrast, the final model performed correctly by providing a clear choice among the options. It correctly stated that entanglement is necessary but not sufficient for violating a Bell inequality and provided a reasonable and accurate explanation. The text flowed well and exhibited coherence, with generally correct grammar.

**Analysis**. In conclusion, continual pre-training on the StackOverflow dataset significantly improved the model's understanding of the content compared to the baseline model. The intermediate version of EPFLearn demonstrated a basic understanding of the question's topic but failed to provide a direct answer, resulting in confused and often incorrect text. In contrast, the final version of EPFLearn showed notable improvement, displaying a better grasp of the subject matter and utilizing more accurate and sophisticated language indicating that the fine-tuning of the NLP4Education dataset was effective.

## 6 Conclusion

**Contributions**. Our study focused on fine-tuning a generative model for educational question answering using the T5 model architecture. We conducted experiments involving continual pre-training on the StackOverflow question and answers dataset, followed by fine-tuning on the NLP4Education dataset. This was done in order to widen the technical understanding of the model and then fine-tuning on the final education content of EPFL. We managed to address the issue of catastrophic forgetting because the model demonstrated good performance also on the StackOverflow datasets even after fine-tuning.

**Performance**. The evaluation of our model demonstrated promising results. We achieved a BLEU score of 40.89, along with Rouge-1, Rouge-2, and Rouge-L scores of 67.47, 43.45, and 63.71, respectively. These metrics indicate the effectiveness of our model in generating relevant and accurate answers. Furthermore, we also observed qualitatively the performance of the model which provided accurate and correct answers with nuanced and specific vocabulary. However, the generated answers of the EPFLearn final version sometimes appeared to be too generic, which suggests some room for improvement. In summary, our study contributes to the advancement of generative models for educational question answering, showcasing the effectiveness of fine-tuning the T5 model architecture.

**Limitations**. Although our study yielded valuable insights, it is important to acknowledge its limitations. Due to computational constraints, we were only able to train the model with a limited number of parameters for a limited number of epochs. This constraint may have impacted the model's performance, and future work should consider allocating additional computational resources and conducting more extensive pre-training on StackOverflow. Furthermore, given the success of our model, exploring the potential advantages of utilizing a larger model with an increased number of parameters, such as the t5-3b model with its 3 billion parameters, could potentially lead to a more accurate and robust chatbot. Furthermore, BLEU and ROUGE are not completely reliable metrics when evaluating performance, and using other metrics such as BERTscore or LSA might provide additional insight.

**Future work**. A clear direction for future developments is the incorporation of the reward model we developed in the fine-tuning process using reinforcement learning with human feedback (RLHF). This would enable the model to further optimize its performance based on specific objectives and feedback, leading to enhanced QA capabilities in educational settings captured by the reward model which was developed.

## References


[1] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. May 2019. DOI: 10.48550/arXiv.1810.04805. URL: http://arxiv.org/abs/1810.04805 (visited on June 12, 2023).

[2] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. July 2019. DOI: 10.48550/arXiv.1907.11692. URL: http://arxiv.org/abs/1907.11692 (visited on May 12, 2023).

[3] Colin Raffel et al. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. July 2020. DOI: 10.48550/arXiv.1910.10683. URL: http://arxiv.org/abs/1910.10683 (visited on June 14, 2023).

[4] Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. "Mixout: Effective Regularization to Finetune Large-scale Pretrained Language Models". en. In: Dec. 2019. URL: https://openreview.net/forum?id=HkgaETNtDB (visited on June 12, 2023).

[5] Zixuan Ke et al. *Continual Pre-training of Language Models*. Apr. 2023. DOI: 10.48550/arXiv.2302.03241. URL: http://arxiv.org/abs/2302.03241 (visited on June 16, 2023).

[6] Kishore Papineni et al. *BLEU: A Method for Automatic Evaluation of Machine Translation*. Philadelphia, Pennsylvania, 2002. DOI: 10.3115/1073083.1073135. URL: https://doi.org/10.3115/1073083.1073135.

[7] Chin-Yew Lin. *ROUGE: A Package for Automatic Evaluation of Summaries*. Barcelona, Spain, July 2004. URL: https://aclanthology.org/W04-1013.

# 7 Appendix

## 7.1 Data

This appendix section contains additional information on the data distribution of both datasets. For the EPFL dataset, the Table 1 lists the number of distinct ChatGPT interactions per question, as well as the distribution of confidence scores. As for the StackOverflow dataset, Table 2 lists the upvotes threshold used to filter answers for the text generation model training, as well as the final number of accepted answers for each topic.

Table 3: EPFL dataset: Statistics for the manually collected interactions with ChatGPT.

| Number of distinct chats per question | | Confidence score | |
|---|---|---|---|
| Value | Percentage of the data | Value | Percentage of the data |
| 1 | 21.4% | 0 | 1.2% |
| 2 | 19.2% | 1 | 5.4% |
| 3 | 54.2% | 2 | 8.1% |
| 4 | 5.1% | 3 | 16.5% |
| 5 | 0.02% | 4 | 26.8% |
| 6 | 0.09% | 5 | 41.9% |

Table 4: Upvote threshold for each StackOverflow topic with final number of accepted answers.

| | Upvotes threshold | Number of answers |
|---|---|---|
| Mathematics | 12 | 22495 |
| Physics | 5 | 20337 |
| Software engineering | 5 | 17714 |
| Computer science | 3 | 9803 |
| Chemistry | 5 | 7173 |
| Mechanics | 3 | 4842 |
| Computer science theory | 3 | 4645 |
| Data science | 3 | 3372 |
| Quantitative | 3 | 2706 |
| Engineering | 3 | 1954 |

## 7.2 Reward model

In this appendix section, we detail the reward model's training procedure, results and analysis.

**Classification metrics**. The performance of our reward model was measured using the usual metrics for binary classifiers; namely precision, recall and specificity. In the imbalanced context with fewer correct than incorrect answers, our evaluation mainly relied on the *F1-score* and *ROC-AUC* as they provide accurate insight into the model's ability to differentiate correct from incorrect answers.

**Training procedure**. For the training of the reward model, we will split the training data into a training set and a validation set. The validation set can then be used to evaluate the performance of the reward model. We split each dataset into training, validation and test sets using a 70/10/20 split with shuffling. We first trained the model on the StackOverflow dataset as a form of pre-training over 20 epochs. We then trained our model for another 20 epochs on the EPFL dataset as a final fine-tuning round. After every epoch, the model was evaluated on diverse metrics over a reserved validation set for the current dataset trained on, as shown in Figure 2. The best model was loaded after every dataset training from the epoch at which its weighted cross entropy loss on the validation set was minimized.

**Training hyperparameters**. We used the AdamW optimizer, with batches of 16 samples with weight decay 10%. Linear decay was used with initial learning rates of $10^{-5}$ for StackOverflow and $10^{-6}$ for EPFL. Mixed precision training and gradient accumulation over 4 batches were used to speed up training to 5 hours on a single GPU.

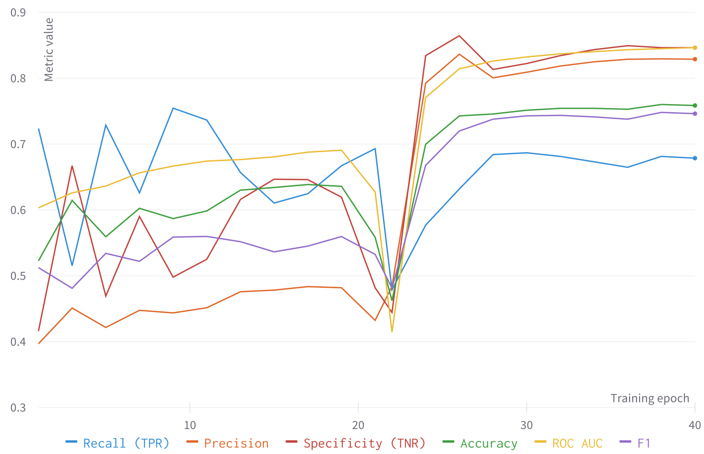


Figure 2: Classifier reward model validation metrics on current dataset during training

**Results**. We evaluated the model on the reserved test sets of both datasets after each training round, as shown in Table 4. We logged all validation metrics on the current dataset during training after every epoch, during the first 20 epochs of training on the StackOverflow dataset, most metrics increased slightly except Recall, which varied unstably. From 20 epochs onwards, the EPFL dataset saw a high increase in validation metrics in the first epochs, then increased marginally but stably until the end of training. We can also see that all metrics had significant improvements on the EPFL after being fine-tuned on it; it reached strong F1-score of 72.1% and ROC AUC of 0.818. We reach decreased final performance on the StackOverflow dataset, with a F1-score of 40.3% and a ROC AUC of 0.557, however this is less important for the desired use of our chatbot.

Table 5: Reward model evaluation after each round of training

| Metric | Round 1 | | Round 2 | |
|---|---|---|---|---|
| | StackOverflow | EPFL | StackOverflow | EPFL |
| Accuracy | 55.8% | 46.2% | 57.4% | 74.0% |
| Specificity | 48.2% | 44.4% | 67.6% | 84.2% |
| Recall | 69.3% | 47.9% | 39.7% | 64.5% |
| Precision | 43.2% | 48.4% | 41.1% | 81.7% |
| F1-score | 53.2% | 48.1% | 40.3% | 72.1% |
| ROC AUC | 0.627 | 0.414 | 0.557 | 0.818 |

**Analysis**. This low performance may explained by the fact that StackOverflow answers are hard to classify; differentiating accepted from non-accepted answers is complex, even for human annotators, because the original poster can only accept a single answer as correct, while multiple answers may well be considered as correct as well. One way to mitigate this bias would be to use the user upvotes of each answers to train the model to identify answers with many upvotes as correct.

Another reason for the low performance on the StackOverflow dataset may also be that training on the EPFL dataset hurts performance on the StackOverflow dataset. As shown in Table 4, the F1-score goes from 53.2% to 40.3% and the ROC AUC goes from 0.627 to 0.557 after training on the EPFL dataset. This means that the model generalizes poorly, and is best-suited to the last dataset it was trained on because it has short-term memory. A way to mitigate these undesired effects would be to use smaller learning rates, especially for the second dataset, and to increase the capacity of the classification head by increasing the number of layers to improve its memory capacity and generalization ability.

### 7.3 Contributions

The respective contributions of each team member were as follows:

**Antoine Bonnet**: Technical implementation of reward and generative models. Fine-tuning the reward and generative models and hyperparameter tuning. Report redaction.

**Silvia Romanato**: Google Cloud installation with Alexander. Pre-processing the EPFL data. Helping Antoine with debugging the model code. Helping train the generative model, tune training hyperparameters and evaluation of the model. Report redaction.

**Alexander Sternfeld**: Google Cloud installation with Silvia. Pre-processing the StackOverflow data for usage. Helping train the generative model and tune generation hyperparameters. Report redaction.