Fine-tune LLM for Language Translation

Fangyu Jiang Matric No.: 429844
RPTU Kaiserslautern, Department of Computer Science

Note: *This report contains a project documentation and reflection on the portfolio task submitted for the lecture Engineering with Generative AI in WiSe 2024-25. This report is an original work and will be scrutinised for plagiarism and potential LLM use.*

1 Portfolio documentation

Compile a comprehensive documentation of your project, including all the project phases. You will need to explain every choice you made during the project and your thoughts about the results you get. You will introduce the results in suitable visualisation. Furthermore, you will need to explain which criteria you follow to build your prompts and how they affect the results.

Students write the entire documentation with sections, sub-sections, diagrams, etc in this section. Please write as comprehensively as possible. Head to the document 1_documentation.tex. You are free to use as many subsections as required. We will not provide a template for documentation.

1.1 Research Phase

In the research phase, there are two selections I'm going to make: the dataset selection and the model selection.

1.1.1 [Dataset A] Selection

With regards to the dataset of this language translation task, the basic requirement is that it should come with at least 1,000 pairs of German-French translations. So the selected dataset should include german and french languages, at least 1,000 rows and preferably be used for translation tasks. I applied these 3 filters to huggingface datasets and it gave me 88 datasets. I went through some of them and tried to get to know 1)how the data format looks like; 2)what the dataset originally is used for; 3)how the translation quality is.

First, the dataset must be a translation between German and French. The google/wmt24pp dataset is a translation between German and English, and French and English, so I excluded this type of data and focused on finding a dataset of translation between German and French. This can filter out many datasets, and finally I focused on the Helsinki-NLP/opus-100 [**?**] [**?**] dataset, the Helsinki-NLP/opus_books dataset, and the Helsinki-NLP/europarl [**?**] dataset.

Second, this task does not limit translations to specific fields. The Helsinki-NLP/opus-100 dataset is an English-centered multilingual corpus. All training pairs contain English on the source or target side, and the official does not specify a specific field. The Helsinki-NLP/opus_books dataset is taken from a collection of copyright free books aligned by Andras Farkas. The Helsinki-NLP/europarl dataset comes from the European Parliament, and the

content mainly involves law and politics. Because the task is not limited, it makes me a little entangled. So, I decided to refer to the third point and randomly check the quality of the data.

Third, after comparison, I found that the Helsinki-NLP/opus_books dataset contains strange semantic translations, such as one data "de": "Den ich wandern muß, arms Waisenkind! Weshalb sandten sie mich so weit, so weit," "fr": "«Pourquoi m'ont-ils envoyé si seul et si loin, là où s'étendent les marécages, là où sont amoncelés les sombres rochers?" Even if I used Google Translate, I couldn't understand it, which would inevitably affect fine-tuning, and the dataset is biased towards the storyline, which I don't think is suitable for this fine-tuning, so I excluded the Helsinki-NLP/opus_books dataset. The data in Helsinki-NLP/opus-100 also has a similar situation. One of the sentences is "de": "Schweröle, ausgenommen Schmieröle für Uhrmacherei und dergleichen in kleinen Behältern mit einem Inhalt von bis zu 250 Gramm Öl netto", "fr": "27101931 à 27101999" This does not match at all, and most of them are short sentences, and the quality is not high. The data in Helsinki-NLP/europarl are mostly long sentences. In addition, the content of the Helsinki-NLP/europarl dataset focuses on parliament and has higher quality. So I finally decided to use Helsinki-NLP/europarl as my [Dataset A].

1.1.2 [Model A] Selection

As for Model A, the most basic requirement is a small pre-trained model that can be fine-tuned within a free Google Colab notebook (T4 GPU, 15 GB VRAM, 12.7 GB RAM). In addition, I also need to consider that this task is German-French translation, Model A needs to support multiple languages, and it is best to have a good community ecosystem and related documentation.

I heard about Llama in the practice class, and I went to learn about Llama and found that Llama 3.2 1B is a good choice. According to the official website [**?**], the 1B model requires at least 4GB VRAM, and may be more during training. It is suitable for inference and lightweight fine-tuning on Colab. 3B may require more than 16GB VRAM during training, and requires additional optimization when training on Colab, and it may not be enough. So I decided to use Llama 3.2 1B to try the generated translation effect. In addition, I specifically chose the Instruction-tuned version of llama so that the model can better understand from the instructions that the task type is translation.

1.2 Design Phase

1.2.1 Fine-tuning Approach

Considering (1) the limited computing environment on Google Colab (T4 GPU, 15GB VRAM, 12.7GB RAM), (2) Llama-3.2-1B has 1 billion parameters, and (3) Dataset A has only 1,000 pairs of German-French translations, LoRA (Low-Rank Adaptation) [**?**] could be an efficient fine-tuning method.

LoRA inserts a small number of trainable parameters into the model through low-rank decomposition without changing the parameters of the original model, significantly reducing the training parameters and saving VRAM. However, the complete model weights still need to be loaded into the VRAM.

Also, the PEFT (Parameter-Efficient Fine-Tuning) library [**?**] from huggingface provides these fine-tuning methods. Through the interface of the PEFT library, I can easily integrate the LoRA method into my training process and automatically quantize and fine-tune the parameters. Therefore I decided to use LoRA from the PERT library as my fine-tuning method.

1.2.2 Split Ratio within [Dataset A]

First, because a larger model will be used in the subsequent step to generate a synthetic data set to optimize the model using a cross-validation strategy, as required by the task, a validation set is not required.

Second, because I randomly selected 1,000 pairs of German-French translation data from Dataset A, and then divided these 1,000 pairs of German-French translations into training and test datasets. When the amount of data is small, a larger training set (such as 80% training set and 20% test set or 90% training set and 10% test set) is usually selected to enhance the learning ability of the model.

Finally, because the test set will be used to evaluate different versions of the model, a slightly larger test set of 20% will be more stable than a 10% test set, and the volatility of the evaluation results will be smaller.

So I chose to split [Dataset A] into 80% training set and 20% test set.

1.2.3 Prompt for Querying A Large Model

First, I chose DeepSeek-V3 [**?**], which has more than 32B parameters. The reasons are: (1) DeepSeek-V3 has no less than 32B parameters; (2) DeepSeek-V3 supports German and French; (3) DeepSeek-V3 training covers the text of the European Parliament; (4) DeepSeek-V3's API is cheaper than OpenAI's.

Second, when querying the large model to generate 1,600 pairs of German-French data sets, the prompts are divided into system prompts and user prompts. The system prompt is

"""You are a professional DE-FR corpus generator. Create batch translations with: 1. German: 20-30 words, formal EU parliamentary style 2. French: Accurate technical translations 3. Format: ["de": "German text", "fr": "French translation",...]""",

and the user prompt is

f"""Generate a batch of batch_size German-French translation pairs. Requirements: - Unique legislative contexts per pair - No repeated phrases - Valid JSON array format - No markdown formatting""".

Of course, I have used deepseek itself to help me optimize prompts. This is something I learned when doing the DIFY exercise, and the prompt can be further strengthened through large language models. And it is not difficult to understand from the prompts themselves. First, the two prompts are divided into system-level context prompts and user-level task prompts according to the principle of role separation. Secondly, the system-level context prompt tells the big model that its role is a professional German-French bilingual translation generator. It also tells it that the topic of the dataset to be generated is the German-French translation pairs related to the European Parliament, each German sentence has 20-30 words, and requires an accurate and professional French translation. Finally, the data format is standardized to

facilitate subsequent processing into a data structure. Third, the user task-level prompt tells the model how many pairs of translations are needed for a single batch, and they must be non-repetitive and preferably have different legislative backgrounds. It also emphasizes that the format should be JSON, not markdown, to facilitate subsequent data processing. Eventually, the final result was indeed in line with expectations, and [Dataset B] was synthesized smoothly.

1.2.4 Evaluation Metric

BLEU (Bilingual Evaluation Understudy) [**?**] is the most commonly used machine translation evaluation indicator and is suitable for most translation tasks. BLEU is suitable for evaluating the accuracy of short text translation, such as short sentences and vocabulary-level translation. Considering that this translation task is sentence-level translation and most of the test set are short sentences, I chose BLEU as the evaluation metric.

1.3 Implementation Phase

The implementation phase is divided into 12 steps according to the guidance of the task book.

1. Load [Dataset A] and split it according to the designed ratio. Because of the GenAI course, I started to progress from hearing about huggingface to actually using it. Not only are the datasets and large models found on huggingface open source, but the platform also provides many ready-made tool libraries needed for large models. The first step is to use the huggingface dataset library to load the europarl dataset, and randomly select 1000 pairs according to the fixed random seed value 123 to ensure the repeatability of the experiment. Then divide the training set and test set according to the 8:2 ratio planned in the design stage. However, I didn't quite understand why there was no validation set to assist in training at first, but I understood it later because a larger dataset will be generated to strengthen the data comparison.

2. Load your chosen pre-trained model [Model A]. The second step is to use the pipeline method provided by the huggingface library to load the model and the corresponding tokenizer. The pipeline highly abstracts the loading of large models and can make reasoning more convenient. But in fact, I struggled for a long time on this step and the choice of model. I even spent almost two days on deciding whether to use the ordinary version of llama or the instruction-tuned version of llama. Finally, by comparing and running the evaluation of the third step several times, I found that the instruction-tuned llama can indeed understand and obey instructions better than the ordinary version of llama, and the instruction-tuned version of llama also supports pipeline messages, while the ordinary version of llama does not support it, which makes me a little confused. Unfortunately, I don't have time to study it in detail.

3. Evaluate [Model A] on the test dataset [Dataset A: Test] using the chosen metric.

4. Fine-tune [Model A] on the training dataset [Dataset A: Train] to create [Model B].

5. Evaluate [Model B] on the test dataset [Dataset A: Test] using the chosen metric.

6. Use the designed prompt to generate a new synthesized dataset [Dataset B], twice the size of the training set [Dataset A: Train], using the selected larger model.

7. Fine-tune [Model A] on the synthesized dataset [Dataset B] to create [Model C].

8. Evaluate [Model C] on the test dataset [Dataset A: Test] using the chosen metric.

9. Combine [Dataset A: Train] and [Dataset B], shuffle them with suitable seeds, and create [Dataset C].
10. Fine-tune [Model A] on the combined dataset [Dataset C] to create [Model D].
11. Evaluate [Model D] on the test dataset [Dataset A: Test] using the chosen metric.
12. Plot the performance of all models using appropriate visualizations.

2 Reflection

In 3-5 pages, 1500-2000 words

The purpose of the reflection is to show that you can reflect and assess your own work and learning process critically.

This section needs to be adjusted to align with the reflection requirements specified in the selected task.

Note: You should address all the questions from your selected task. Please list each question and provide your answers in the following enumeration.

For example:

1. What was the most interesting thing that you learnt while working on the portfolio? What aspects did you find interesting or surprising?

   Answer: This is my answer...

2. Which part of the portfolio are you (most) proud of? Why? What were the challenges you faced, and how did you overcome them?

   Answer: This is my answer...

3. What adjustments to your design and implementation were necessary during the implementation phase? What would you change or do differently if you had to do the portfolio task a second time? What would be potential areas for future improvement?

   Answer: 1. 2. If I had the chance to do it again or allocate more time to the portfolio earlier, I would really like to figure out what the difference is between the regular version of llama and the instruction-tuned version of llama that causes the instruction-tuned version of llama to support pipeline messages while the regular version of llama does not. And when the implementation was really finetuned, I realized that the translation type is a seq2seq task, and using a large seq2seq model may have better results. However, llama is of the casual$_LMtype, soIhavetousetheinstructionversiontoconvertthetranslationtaskintoareasoningtaska$

4. Include a brief section on ethical considerations when using these models on language translation tasks.

   Answer: See Section **??** on page **??**.

5. From the lecture/course including guest lectures, what topic excited you the most? Why? What would you like to learn more about and why?

Answer: This is my answer...

6. How did you find working with DIFY platform during the course work? Would you recommend using DIFY in learning Generative AI technologies and why? What is the best start for learning Generative AI either by Python code or No-code platforms and why?

   Answer: This is my answer...

7. How did you find the assignments and exercises in the course and how they help you in portfolio exam?

   Answer: This is my answer...

3 Ethical Considerations

A brief section on ethical considerations when using these models on language translation tasks. Readings:

- ACL Ethics Policy (2023)
- EU AI Act (Article 10 on Translation Systems)
- ISO/IEC 24028:2020 (AI Trustworthiness)

All of the resources used by the student to complete the portfolio task should be organised in the references section. **Note that the Reference section does not count towards the number of pages of the report.** Example references are given below [**?**] [**?**] [**?**]. **If you are using a reference manager like Zotero, you can export your Zotero library as a .bib file and use it on Overleaf. As you cite the article/technology/library in your main text, the References section will automatically update accordingly.** Please include a full list of references found. If students are using Zotero for their research paper management, a bibTeX will help them during citation which automatically adds references to the report.

References

[1] Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. Improving massively multilingual neural machine translation and zero-shot translation. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online, July 2020. Association for Computational Linguistics.

[2] Jörg Tiedemann. Parallel data, tools and interfaces in OPUS. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA).

[3] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand, September 13-15 2005.

[4] Meta AI. Llama 3.2 requirements.

[5] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[6] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`, 2022.

[7] DeepSeek. Introducing deepseek-v3. [Online], December 2024.

[8] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics.

[9] Albert Einstein. Zur Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies]. *Annalen der Physik*, 322(10):891–921, 1905.

[10] Donald Knuth. Knuth: Computers and typesetting.

[11] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1993.