ADVANCED LEARNING FOR TEXT AND GRAPH DATA

Lab session 4: NLP Frameworks

Lecture: Prof. Michalis Vazirgiannis
Lab: Dr. Guokan Shang and Hadi Abdine

November 07, 2023

This handout includes theoretical introductions, coding tasks and questions. Before the deadline, you should submit on moodle **or** here a **.zip** file named `Lab<x>_lastname_firstname.pdf` containing a `/code/` folder (itself containing your scripts with the gaps filled) and an answer sheet, following the template available here, and containing your answers to the questions. Your answers should be well constructed and well justified. They should not repeat the question or generalities in the handout. When relevant, you are welcome to include figures, equations and tables derived from your own computations, theoretical proofs or qualitative explanations. **One submission is required for each student. The deadline for this lab is November 14, 2023 11:59 PM**. No extension will be granted. Late policy is as follows: $]0, 24]$ hours late $\rightarrow$ -5 pts; $]24, 48]$ hours late $\rightarrow$ -10 pts; > 48 hours late $\rightarrow$ not graded (zero).

---

1 Notebook

https://colab.research.google.com/drive/1hw7M$_4UKHEFECbROyzY0hDE6AXDVi8_0?usp = sharing$

2 Introduction

Transformer models have been the predominant deep learning models used in NLP for the past several years, with well-known exemplars in GPT-3 [1] from OpenAI and its predecessors, the Bidirectional Encoder Representations from Transformers model (BERT)[3] developed by Google, XLNet [11] from Carnegie Mellon and Google, and many other models and variants besides. Following the paper "Attention is All You Need" [8] from 2017, the unofficial milestone marking the start of the "age of transformers", transformer models have gotten bigger, better, and much closer to generating text that can pass for human writing, as well as improving substantial on statistical loss metrics and standard benchmarks.
In order to pretrain and finetune transformer based language models, multiple frameworks and libraries have been introduced such as Fairseq[7] and HuggingFace's transformers [9].

Fairseq: fairseq is an open-source sequence modeling toolkit developed by Meta AI. Fairseq allows researchers and developers to train custom models for translation, summarization, language modeling, and other text generation tasks. The toolkit is based on PyTorch and supports distributed training across multiple GPUs and machines. Fairseq also supports fast mixed-precision training and inference

on modern GPUs.Fairseq provides researchers with smooth implementation of sequence to sequence models. It supports various models such as RoBERTa [6] and BART [5].

HuggingFace: Hugging Face is an NLP-focused startup with a large open-source community, in particular around the Transformers library. HuggingFace/Transformers is a python-based library that exposes an API to use many well-known transformer architectures, such as BERT, RoBERTa, GPT-2 or DistilBERT, that obtain state-of-the-art results on a variety of NLP tasks like text classification, information extraction, question answering, and text generation. HuggingFace also provides almost 2000 data sets (using the library *datasets*) and layered APIs, allowing programmers to easily interact with those models using almost 31 libraries. Most of them are deep learning, such as Pytorch, Tensorflow, Jax, ONNX, Fastai, Stable-Baseline 3, etc.

In this lab you will learn how to use Fairseq and HuggingFace transformers - The most used libraries by researchers and developers to pretrain and finetune language models - to finetune pretrained language models.

3 Installations

Use this installation guideline if you don't intend to use Google Colab.
To install the packages required in this lab we first recommend that you create a new conda environment to avoid any potential conflicts. For example you can do that by running the following command:

```
conda create -n lab_frameworks python=3.9
```

Then activate the new environment by running:

```
conda activate lab_frameworks
```

3.1 Installing Sentencepiece

In this lab we use the sentencepiece tokenizer[1]. Do install it for python you can simply use the following command.

```
pip install sentencepiece
```

Otherwise, if you want to use it in the command-line check the Github repository.

3.2 Installing Fairseq

We created a fork of the of the official Fairseq project with minor modification to take into considerations the architecture of the model that we used in this lab ($\text{Roberta}_{\text{small}}$). To install the framework run the commands below:

```
git clone https://github.com/hadi-abdine/fairseq.git
cd fairseq
pip install --editable ./
```

Installing from source will make it possible to edit the source code even if this not will be required in our lab. Fairseq and all its dependencies should now be installed on your device.

[1] https://github.com/google/sentencepiece

3.3 Installing HF Transformers

Use the following commands to install the editable package on your device:

```
git clone https://github.com/huggingface/transformers.git
cd transformers
pip install -e .
```

4 The $RoBERTa^{fr}_{SMALL}$ Model

Our first pretrained language model in this lab is based on RoBERTa[6]. It has four transformer's encoder layers and pretrained with the MLM (Masked Language Model) task for 10 epochs using two Nvidia RTX A6000 GPUs on 8M French documents (approximately 39 GB of French raw text) from the dataset mC4[10]. Our dictionary contains 32k tokens obtained by extracting the most 32k frequent French tokens from the dictionary of XLMR[2].

Question 1
Compute the number of parameters of the model manually.

Hints:

- For details about the architecture check `roberta_small_architecture()` in `fairseq/fairseq/models/roberta/model.py`
- You can omit the biases and the parameters of normalization layers.
- Don't count the parameters of the language model head.

Task 1
Compute the number of parameters using `torch` or `Fairseq` to verify your answer.

4.1 Finetuning using Fairseq

To train models using Fairseq you have to:

Task 2
Prepare the data for finetuning. To do so you have to:

1. Use the provided sentencepiece model to tokenize the text.
2. Binarize the data using `fairseq-preprocess` command or the `fairseq/fairseq_cli/preprocess.py` script.

Task 3
Using Fairseq, finetune the pretrained model on the dataset CLS-Books to perform binary text classification:

1. Use three different seeds.
2. For each seed, choose the checkpoint with best validation accuracy.
3. Report the average accuracy and its standard deviation on the test set.

Make sure to use the following hyper-parameters: *batch size*=8, *max number of epochs*: 5, *optimizer*: Adam, *max learning rate*: 1e-05, *warm up ratio*: 0.06, *learning rate scheduler*: linear.

Task 4
Finetune a random checkpoint of the model on CLS-Books and compare the result with the one from the previous task.
hint: in Fairseq to train a randomly initialized model, you can simply replace the value of `--restore-file` with a nonexistent path

4.2 Finetuning using HF transformers

In this section of the lab we will use HuggingFace's transformers to finetune the $RoBERTa^{fr}_{SMALL}$ using an already existing fintuning script `run_glue.py` [2].

Task 5
Repeat the experiment of Task 3 using HF transformers. Make sure to use the same hyper-parameters.

5 Large Language Models

A large language model (LLM) is a type of language model notable for its ability to achieve general-purpose language understanding and generation (e.g. BLOOM, GPT3, Llama). In this section of the lab, we will finetune BLOOM-560m on a Q/A datset. In order to finetune this model on a single 12GB GPU, we will use LoRA [4] and quantization using BitsAndBytes library to perform 4-bit quantization to compress the model.

Task 6
Fill the gap to get the bnb_config (BitsAndBytes configuration). Hint: https://huggingface.co/blog/4bit-transformers-bitsandbytes

Task 7
Fill the gap in the function `print_trainable_parameters` to get the number of trainable parameters.

Question 2
Define each parameter used in LoraConfig. Hint: https://www.anyscale.com/blog/fine-tuning-llms-lora-or-full-parameter-an-in-depth-analysis-with-llama-2

[2] https://github.com/huggingface/transformers/tree/main/examples/pytorch/text-classification

Task 8
Fill the gap in the function `generate_prompt` and in the prompts variables to transform the data into the prompting format of:
<human>: question?
<assistant>: response.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[2] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] 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. In *International Conference on Learning Representations*, 2022.

[5] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[6] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[7] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[9] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[10] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online, June 2021. Association for Computational Linguistics.

[11] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.