1 Model, 1 GPU, 1 Day: Language Model Efficiency Challenge

1 Introduction

Large Language Models (LLMs) have transformed natural language processing, achieving state-of-the-art (SoTA) performance across virtually all tasks. Yet, their large size and high computational requirements pose challenges for practical use, especially in resource-limited settings. Why are models so large? It has been observed that language model performance improves *steadily* as we increase the model size, dataset size, and amount of compute used for training. For optimal performance, all three factors must be scaled up in tandem.

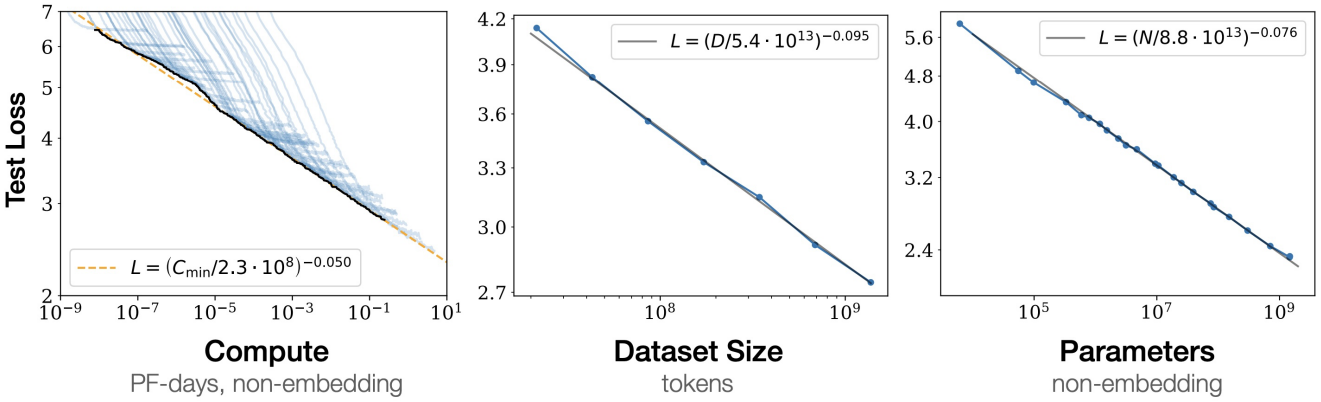


Figure 1: *Language modeling performance improves smoothly as we increase the model size, dataset size, and amount of compute used for training. PF-days refers to PetaFLOP days. "One petaFLOPS-day is the number of computations that could be performed in one day by a computer capable of calculating a thousand trillion floating point operations (like adding or multiplying two numbers) per second. For comparison, a standard laptop would need about a year to reach one petaFLOPS-day. That laptop would need several millennia to reach the 3640 petaFLOPS-days it took to train GPT-3." (Source)*

This "blessing of scale" has led to a steady, competitive race to develop larger language models (see Figure 2). As a result, the costs of storing, fine-tuning, and querying LLMs have become massive. SoTA models require thousands of GPU hours, terabytes of memory, and can easily run into millions of dollars. Consequently, access to high-performance LLMs is primarily limited to large-scale AI companies. For context, consider a model with 7 billion parameters, which is tiny compared to models like GPT-3. Full-parameter fine-tuning of a 7B model requires approximately 168 GB of GPU memory. Given that an NVIDIA H100 GPU (80GB) is priced at over 25000 euros, just acquiring the necessary hardware to play around with this model could cost upwards of 50000 euros.

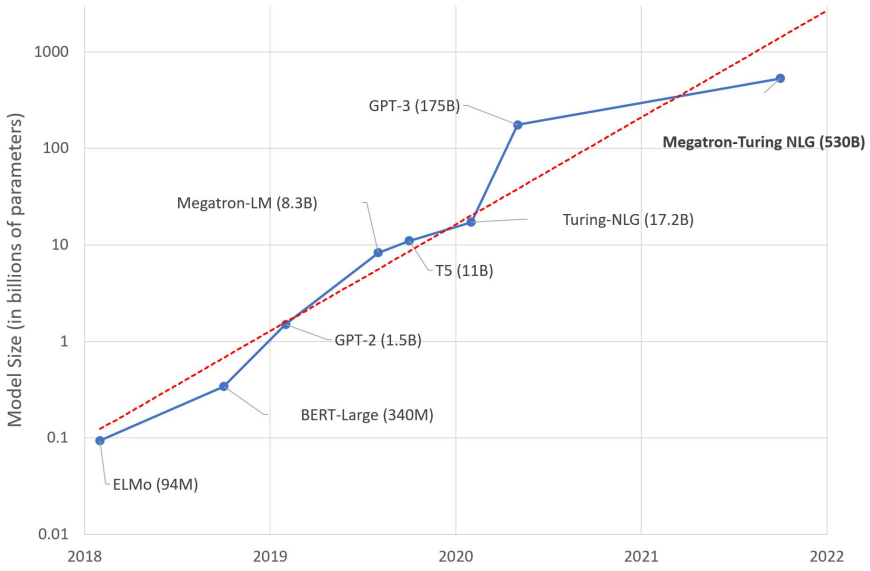


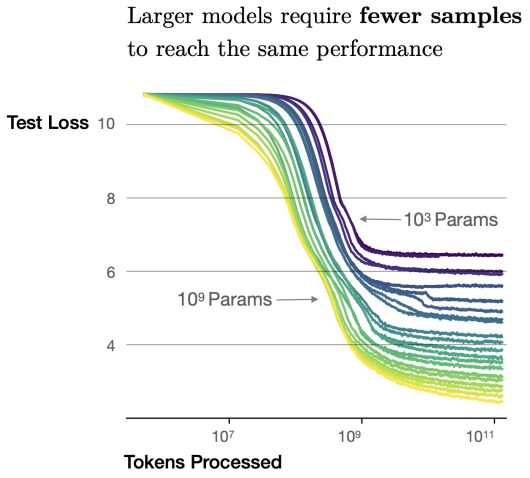
Figure 2: Every year, model size increases by 10x.

To address the computational challenge of training LLMs, a range of efficient LLM training techniques have been developed. These methods focus on optimizing memory and computational efficiency while preserving high performance, during training and inference. Among the most prominent methods are Parameter-Efficient Fine-Tuning (PEFT) and model compression techniques.

In this project, you will tackle the challenge of training and evaluating an LLM under tight computational constraints that mirror real-world limitations. Specifically, *your task is to fine-tune a pre-trained language model on a specific NLP benchmark, using a single GPU and a maximum 24 hours of training.* ***One LLM, one GPU, one day of training.*** This will force you to navigate the trade-off between staying within computational boundaries and achieving the best possible performance. To be clear: you have a maximum of 24 hours to train your *final* model. You are encouraged to try out a few different training schemes beforehand for some hours. But once you commit to your final approach, your training must be completed within the 24-hour limit, under hardware constraints that will be specified later in this document.

The typical training cycle of an LLM involves three stages: pre-training on vast, diverse corpora to learn general language patterns, supervised fine-tuning (SFT) on task-specific datasets to specialize the model, and alignment through techniques like RLHF or DPO, to ensure the model's outputs align with human preferences or safety guidelines. We will bypass the pretraining phase entirely. Your focus will be on performing supervised fine-tuning (SFT) and alignment efficiently *and* effectively. You will need to do the following:

1. **Choose a pre-trained base model to work with.** Larger models offer better performance and are also more sample-efficient, meaning they can achieve higher accuracy with less training time:



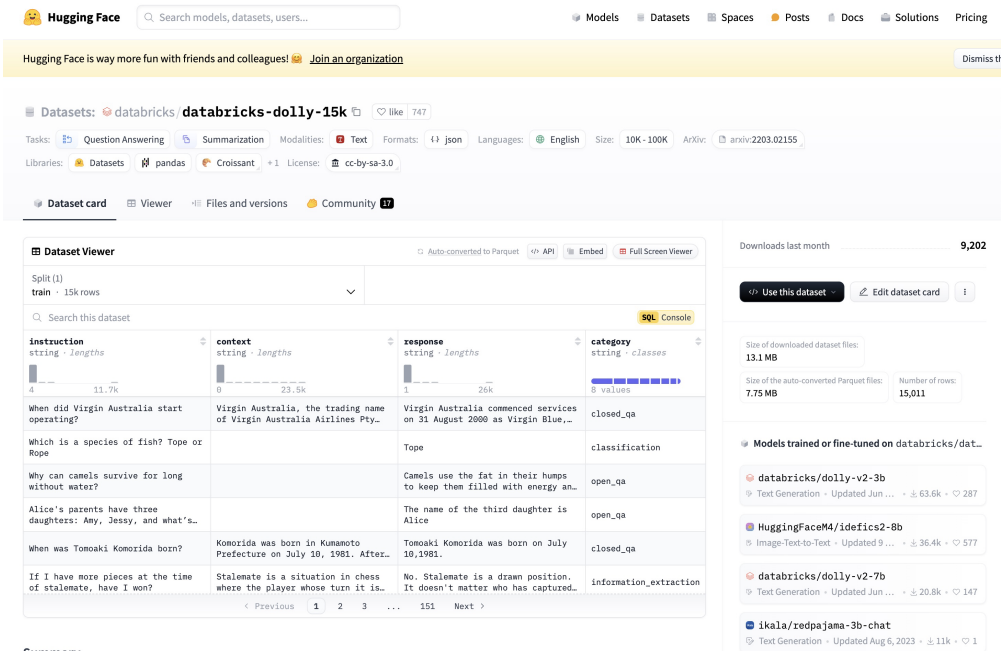
However, since you only have access to a limited GPU (a single Nvidia 4070 Ti with 12 GB of RAM), you need to balance the size of the model with available resources. The key is to select a model that is as large as possible without exceeding the memory or computational limits of your hardware, ensuring that you can make the most of your GPU and the 24 hours available for training. **The starting model for the competition should be a base model without instruction-tuning**. Here are some suggested models to get you started, but do feel free to explore other models:

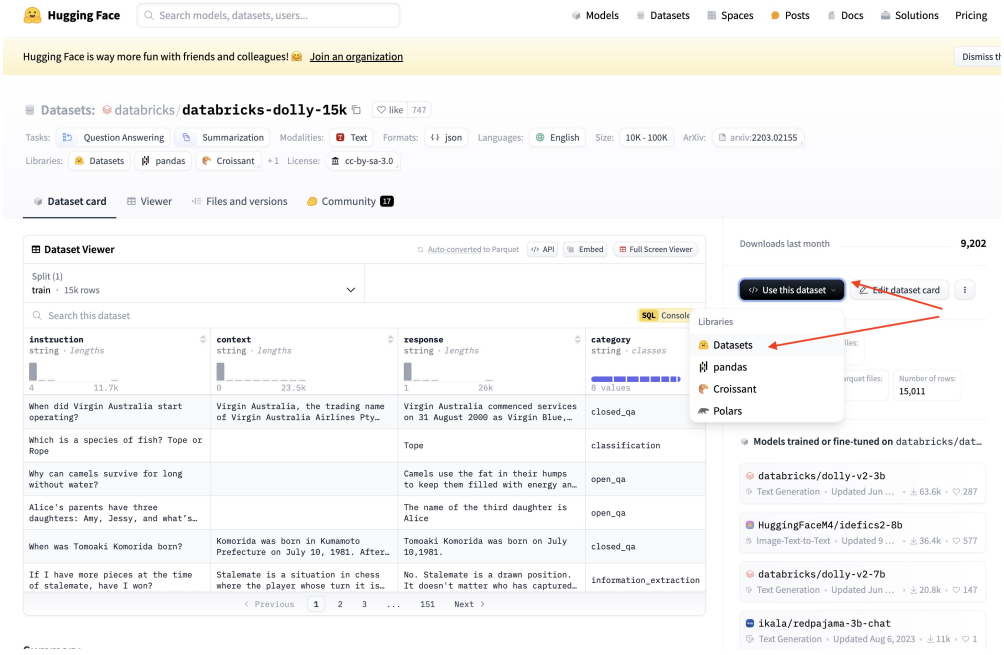
- LLama-3.1-8B
- Mistral 7B v0.3
- Qwen2.5-7B

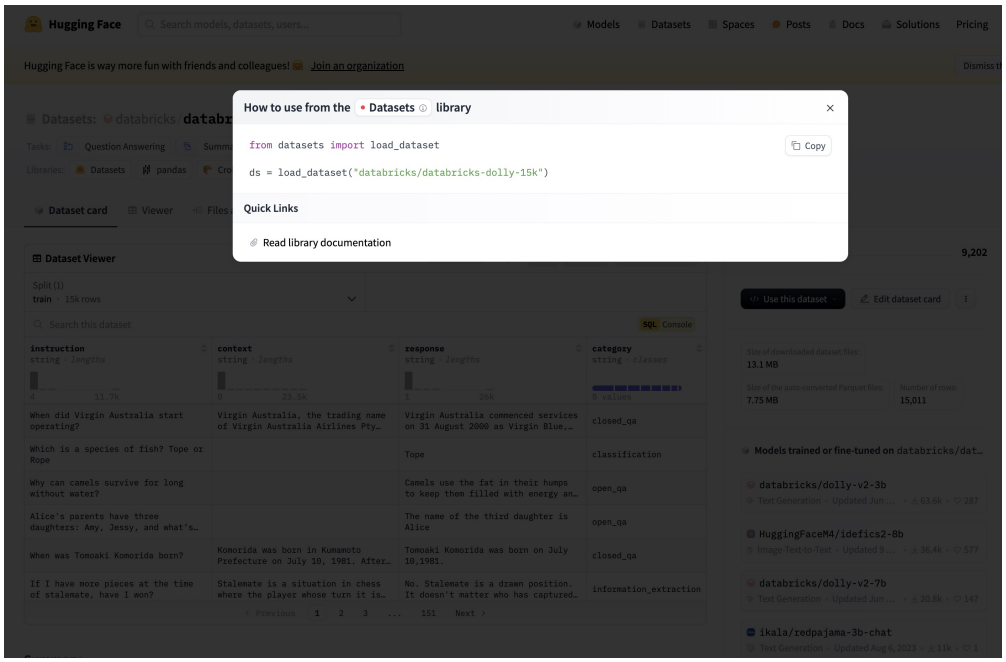
2. **Choose datasets to fine-tune your model**. For smaller models, the quality of the dataset becomes more important. **A high-quality dataset that is well-matched to your task will help your model learn faster and more effectively**, compensating for the limitations in model size and training time. Here are some suggested datasets to help you get started. However, part of your task is to select an appropriate combination of datasets for your specific model and task requirements:

   - Databricks Dolly 15K
   - OpenAssistant Conversations Dataset (OASST1)
   - LIMA: Less Is More for Alignment
   - Ultrafeedback binarized (preference data)

   If you use Hugging Face, you can easily check out and find the code needed to load the datasets:


Hugging Face
Search models, datasets, users...
Models
Datasets
Spaces
Posts
Docs
Solutions
Pricing
Hugging Face is way more fun with friends and colleagues! Join an organization
Datasets: databricks / databricks-dolly-15k
like 747
Tasks: Question Answering
Summarization
Modalities: Text
Formats: json
Languages: English
Size: 10K - 100K
ArXiv: arxiv:2203.02155
Libraries: Datasets
pandas
Croissant
+1
License: cc-by-sa-3.0
Dataset card
Viewer
Files and versions
Community 17
Dataset Viewer
Auto-converted to Parquet
API
Embed
Full Screen Viewer
Split (1)
train · 15k rows
Search this dataset
SQL Console
instruction
string · lengths
context
string · lengths
response
string · lengths
category
string · classes
8 values
When did Virgin Australia start operating?
Virgin Australia, the trading name of Virgin Australia Airlines Pty…
Virgin Australia commenced services on 31 August 2000 as Virgin Blue,…
closed_qa
Which is a species of fish? Tope or Rope
Tope
classification
Why can camels survive for long without water?
Camels use the fat in their humps to keep them filled with energy an…
open_qa
Alice's parents have three daughters: Amy, Jessy, and what's…
The name of the third daughter is Alice
open_qa
When was Tomoaki Komorida born?
Komorida was born in Kumamoto Prefecture on July 10, 1981. After…
Tomoaki Komorida was born on July 10,1981.
closed_qa
If I have more pieces at the time of stalemate, have I won?
Stalemate is a situation in chess where the player whose turn it is…
No. Stalemate is a drawn position. It doesn't matter who has captured…
information_extraction
Previous
1
2
3
...
151
Next
Downloads last month
9,202
Use this dataset
Edit dataset card
Size of downloaded dataset files:
13.1 MB
Size of the auto-converted Parquet files:
7.75 MB
Number of rows:
15,011
Models trained or fine-tuned on databricks/dat…
databricks/dolly-v2-3b
Text Generation · Updated Jun … · 63.6k · 287
HuggingFaceM4/idefics2-8b
Image-Text-to-Text · Updated 9 … · 36.4k · 577
databricks/dolly-v2-7b
Text Generation · Updated Jun … · 20.8k · 147
ikala/redpajama-3b-chat
Text Generation · Updated Aug 6, 2023 · 11k · 1


Hugging Face
Search models, datasets, users...
Models
Datasets
Spaces
Posts
Docs
Solutions
Pricing
Hugging Face is way more fun with friends and colleagues! Join an organization
Datasets: databricks / databricks-dolly-15k
like 747
Tasks: Question Answering
Summarization
Modalities: Text
Formats: json
Languages: English
Size: 10K - 100K
ArXiv: arxiv:2203.02155
Libraries: Datasets
pandas
Croissant
+1
License: cc-by-sa-3.0
Dataset card
Viewer
Files and versions
Community 17
Dataset Viewer
Auto-converted to Parquet
API
Embed
Full Screen Viewer
Split (1)
train · 15k rows
Search this dataset
SQL Console
instruction
string · lengths
context
string · lengths
response
string · lengths
category
string · classes
8 values
When did Virgin Australia start operating?
Virgin Australia, the trading name of Virgin Australia Airlines Pty…
Virgin Australia commenced services on 31 August 2000 as Virgin Blue,…
closed_qa
Which is a species of fish? Tope or Rope
Tope
classification
Why can camels survive for long without water?
Camels use the fat in their humps to keep them filled with energy an…
open_qa
Alice's parents have three daughters: Amy, Jessy, and what's…
The name of the third daughter is Alice
open_qa
When was Tomoaki Komorida born?
Komorida was born in Kumamoto Prefecture on July 10, 1981. After…
Tomoaki Komorida was born on July 10,1981.
closed_qa
If I have more pieces at the time of stalemate, have I won?
Stalemate is a situation in chess where the player whose turn it is…
No. Stalemate is a drawn position. It doesn't matter who has captured…
information_extraction
Previous
1
2
3
...
151
Next
Downloads last month
9,202
Use this dataset
Edit dataset card
Libraries
Datasets
pandas
Croissant
Polars
Number of rows:
15,011
Models trained or fine-tuned on databricks/dat…
databricks/dolly-v2-3b
Text Generation · Updated Jun … · 63.6k · 287
HuggingFaceM4/idefics2-8b
Image-Text-to-Text · Updated 9 … · 36.4k · 577
databricks/dolly-v2-7b
Text Generation · Updated Jun … · 20.8k · 147
ikala/redpajama-3b-chat
Text Generation · Updated Aug 6, 2023 · 11k · 1



3. **Choose your training scheme:** You will need to select a combination of fine-tuning techniques, alignment techniques, model compression techniques, and hyper-parameters. Remember, you have limited GPU resources and limited training time. So **you need to carefully plan how many parameters you will be training, and how much memory they will require during training**. Consider how model compression techniques (e.g., pruning, quantization) can reduce memory usage. Don't just experiment blindly and hope for the best, or you will repeatedly encounter out of memory errors or kernel crashes. Perform calculations on the number of trainable parameters and estimate memory usage under different techniques. **You will need to report your calculations and document your experiments and comparisons in a final report**. Hyper-parameter selection will be important too; be mindful of memory-hungry settings like batch size. You can try out hyperparameter optimization tools like Optuna to look for good hyper-parameter settings. If you choose to do alignment with techniques like RLHF or Direct Preference Optimization (DPO), take into account training speed and convergence rates. RLHF can be more resource-intensive than DPO and much trickier to get right, so choose carefully based on how much training time you have left and how fast you can expect convergence. You don't need to use all techniques covered in class. Also, you are allowed to use techniques beyond what was covered in class. However, you must show a clear understanding of how they work by explaining them thoroughly in both your report and final presentation.

2 Rules

1. Submissions must be fully reproducible, from the initial model through fine-tuning and inference. All models, code, and data must be open-source and included as part of the submission. We will replicate the training process to verify that models perform closely to the reported evaluation scores.

2. Submissions must not use any copyrighted or proprietary data, code, or closed-source content.
3. Teams are encouraged to use any open-source code and libraries, provided that proper attribution is given. The code must be based on PyTorch, Hugging Face, or any other PyTorch-compatible package or framework.
4. The fine-tuning process for the submission must take less than 24 hours to complete on a single Nvidia 4070 Ti GPU. You can try out training with different hardware. However, the final training must be done in *Lab IA*'s computers, to ensure that training is standardized across submissions. We will assign a separate computer to each team.
5. Each team must consist of 2 students. Teams must be formed and their members notified to the professors by **October 30th**. Each team must send an email to the professors indicating the team name and members. Use the mail title "NLP 2 Final project: team details".
6. The model of each team will be scored both quantitatively and qualitatively. The quantitative evaluation will be based in a benchmark for instruction following, such as *IFEval*, and head-to-head evaluation benchmarks, either using human judges or LLM-based judges (e.g., *AlpacaEval*). Model size and latency will also be assessed. The qualitative evaluation via human evaluators will be conducted by your classmates and your professors in one of the last class sessions.

3 Submission requirements

Your submission must include the following components:

1. **Model**:
   - Submit the final fine-tuned model, including all saved weights and checkpoints. The model must be fully reproducible, and you should include any necessary files for inference.
   - **Note**: Keep in mind that the computers in the *Lab IA* are turned off at 3AM. Therefore, you will need to use model checkpoints. Checkpoints save the state of a model during training, allowing you to resume training from a specific point in time, without starting over. If you use the `Hugging Face` class `transformers.TrainingArguments`, there is an argument called `save_steps`, which controls the number of steps between checkpoints. Choose an appropriate number so that you get checkpoints regularly (e.g., every 20 or 30 minutes) to avoid losing progress.
   - **Note:** Keep in mind that *Lab IA* computers may still be accessed by students from outside this course. Therefore, do not wait until the last day for your final training session, as it could be interrupted. Once you've finished training, remember to save the latest checkpoint to your personal computer or a cloud service.
2. **Dataset**:
   - Submit the dataset(s) you used for training and evaluation. Provide clear instructions on how to access the datasets. Ensure that any data preprocessing or data sampling steps are documented.

3. **Report**: The report should provide a detailed overview of your work and must include the following sections:

   (a) **Base model selection**: Explain the pre-trained model you selected and the rationale behind your choice. Discuss the trade-offs between different models and why this model was the best fit for your task and resource constraints.

   (b) **Dataset selection**: Describe the dataset(s) you used, including the reasoning behind the selection. Detail any preprocessing steps or modifications applied to the data.

   (c) **Training scheme**: Provide an in-depth explanation of the training process, including all techniques used, such as parameter-efficient fine-tuning, model compression, and alignment methods. Clearly specify the hyperparameters used (learning rate, batch size, etc.) and the reasoning behind their selection. If you used hyper-parameter search/optimization, describe the methods applied.

   (d) **Trainable parameters**: Include a calculation of the number of trainable parameters, especially if you used methods like LoRA or adapters that affect the number of parameters.

   (e) **Considered approaches**: Document the different strategies you considered during the project. Which methods worked well, and which ones did not? Do you have an idea why this happened? Add a section with insights you have extracted from the training process. Explain why you chose the final approach, particularly in relation to resource limitations (e.g., memory, time, GPU constraints).

   (f) **Resource management**: Discuss how you ensured that the training process stayed within the hardware and time constraints. Explain any considerations, such as memory usage or model compression, that helped you stay within the GPU's limits.

   (g) **Training Process**: Describe the actual training process in detail. This should include:
   - Total training time and how you utilized the 24-hour window.
   - Plots of training and evaluation results (e.g., loss over time, for training and validation).

   (h) **Evaluation process**: Report the obtained IFEval scores of your model. Beware that it may take about two hours to run the evaluation. Moreover, report the latency of the model and the model size, both total and trainable parameters. Low latency and smaller sizes will be considered positively.

The report, along with the oral presentation based on it, will be a key component of the project evaluation. Showcase your understanding of the techniques applied and your ability to manage the available resources effectively.

4 Timeline

- **October 30th**: Teams notified by email. All teams must be formed and their composition submitted to the professors by this date.
- **November 29th**: Submission of the complete team project. This includes the model, dataset(s), and the detailed report as specified in the submission requirements.

- **December 3rd and 5th**: Oral presentations of the team projects. Each team will present their work to the class and discuss their approach, results, and any challenges they encountered. We will discuss the presentation dates with you and, if preferred, we can schedule them a week earlier to avoid coinciding with the exam period. Be aware that the submission date would be moved to November 25th.

5 Starter Kit

In this starter kit, you will find useful code files for fine-tuning a model using Parameter-Efficient Fine-Tuning (PEFT) and model compression techniques. The starter kit includes scripts and notebooks that demonstrate how to:

- Implement PEFT techniques to fine-tune large models efficiently.
- Apply model compression methods such as quantization to reduce memory usage.
- Calculate model size and latency.
- Generate predictions and evaluate a model's performance on instruction-following tasks with the IFEval dataset.

6 Useful links

1. Hugging Face authentication
2. Hugging Face user access token