References

InstructGPT paper (https://arxiv.org/pdf/2203.02155.pdf)

From Language Model to Helpful Assistant: Instruction following

An LLM is trained in a text-completion task: predict the next token.

Through the Universal Model (NLP_Universal_Model.ipynb) we can transform many other tasks into instances of text-completion.

The human user must

- engineer a prompt
- so that the "next token" prediction
- is the desired response

For example, if you want an explanation of a topic

- Don't prompt the LLM with "Explain the Black Scholes pricing formula"
- Formulate the prompt in text-continuation form: "The Black Scholes pricing formula states"

To make LLM's more human-friendly, they can be *fine-tuned* to exhibit *Instruction Following* behavior

- the prompt is treated by the LLM as an *Instruction*:
    - a request to perform a task
    - rather than to predict the next token

Moreover: we often want to constrain the response to be

- helpful: follow the instructions
- truthful: answers are honest
- harmless: avoid bias or suggestions of dangerous behavior

The "LLM's" we have become familiar with are, in fact

- LLM's that have been fine-tuned to act as *Helpful Assistants*

Instruction following

Instruction following behavior consists of 3 components

- *Instruction*
    - describing the task to accomplish
- *Input* or *Context*
    - the input to the task
- *Response* or *Output*
    - the expected response (output)
    - given the Input
    - given the Instruction

For example

- Instruction: Tell me the word that is the opposite of the word that I input
- Input: Stop
- Response: Go

The Context can be empty (pure instruction)

- Instruction: Tell me a joke
- Context:
- Response: Why did the chicken cross the road ...

Let's explore ! A training set for Instruction Following

Here (https://huggingface.co/datasets/databricks/databricks-dolly-15k/viewer/default/train) is a dataset created to Fine-Tune an LLM to become a Helpful Assistant (demonstrate Instruction Following Behavior)

The examples are categorized as demonstrations of several types of tasks expected to be typical prompts, , for example

- `closed_qa`: Question Answering, conditioned on a context
- `open_qa`: Question Answering without a context
- `summarization`: Summarize the Context
- `information_extraction`: extract an Answer (from the Context) to the Question in the Instruction part
- `classification`: what class does the Instruction indicate ?
- `brainstorming`

Beyond Instruction Following

- we often want to Fine-Tune the Helpful Assistant to be conversational.
- the "Chat" in ChatGPT

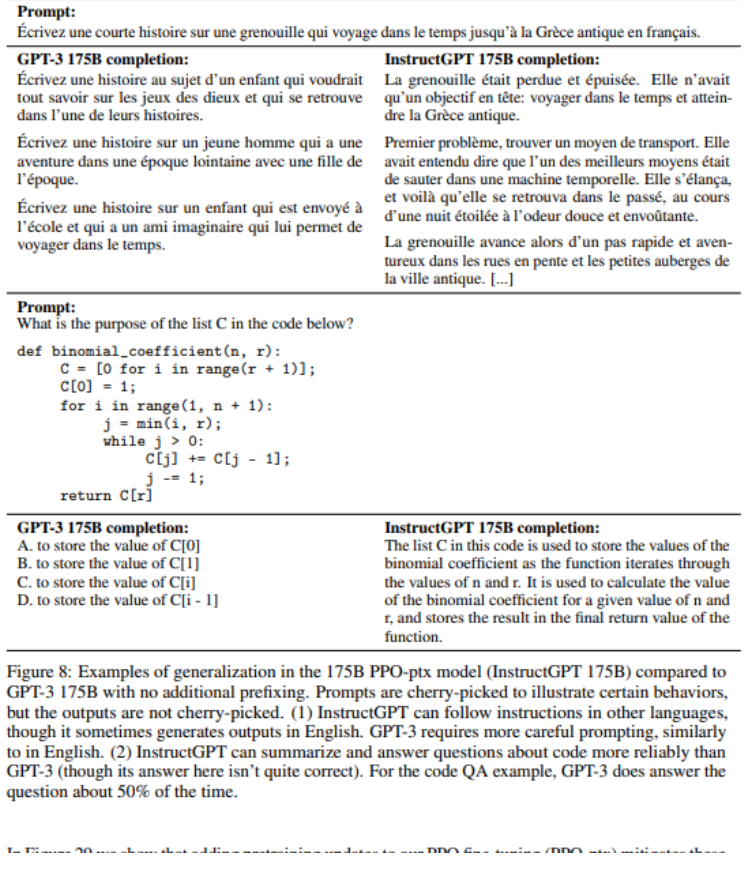
Here (https://huggingface.co/datasets/OpenAssistant/oasst2/viewer/default/train?f%5Blang%5D%5Bvalue%5D=%27en%27&row=37) is a dataset mutli-turn "conversations" between the user (role: `prompter`) and the assistant (role: `assistant`)

- chained together by `message_id` $\rightarrow$ `parent_id`

LLM's struggle with Instruction Following

The following examples illustrate the difference between

- Text Completion behavior (left column, labeled `GPT-3 175B completion`)
- and Instruction Following behavior (right column, labeled `InstructGPT 175B completion`)



Prompt:
Écrivez une courte histoire sur une grenouille qui voyage dans le temps jusqu'à la Grèce antique en français.

GPT-3 175B completion:	**InstructGPT 175B completion:**
Écrivez une histoire au sujet d'un enfant qui voudrait tout savoir sur les jeux des dieux et qui se retrouve dans l'une de leurs histoires. Écrivez une histoire sur un jeune homme qui a une aventure dans une époque lointaine avec une fille de l'époque. Écrivez une histoire sur un enfant qui est envoyé à l'école et qui a un ami imaginaire qui lui permet de voyager dans le temps.	La grenouille était perdue et épuisée. Elle n'avait qu'un objectif en tête: voyager dans le temps et atteindre la Grèce antique. Premier problème, trouver un moyen de transport. Elle avait entendu dire que l'un des meilleurs moyens était de sauter dans une machine temporelle. Elle s'élança, et voilà qu'elle se retrouva dans le passé, au cours d'une nuit étoilée à l'odeur douce et envoûtante. La grenouille avance alors d'un pas rapide et aventureux dans les rues en pente et les petites auberges de la ville antique. [...]

Prompt:
What is the purpose of the list C in the code below?

```
def binomial_coefficient(n, r):
    C = [0 for i in range(r + 1)];
    C[0] = 1;
    for i in range(1, n + 1):
        j = min(i, r);
        while j > 0:
            C[j] += C[j - 1];
            j -= 1;
    return C[r]
```

GPT-3 175B completion:	**InstructGPT 175B completion:**
A. to store the value of C[0] B. to store the value of C[1] C. to store the value of C[i] D. to store the value of C[i - 1]	The list C in this code is used to store the values of the binomial coefficient as the function iterates through the values of n and r. It is used to calculate the value of the binomial coefficient for a given value of n and r, and stores the result in the final return value of the function.

Figure 8: Examples of generalization in the 175B PPO-ptx model (InstructGPT 175B) compared to GPT-3 175B with no additional prefixing. Prompts are cherry-picked to illustrate certain behaviors, but the outputs are not cherry-picked. (1) InstructGPT can follow instructions in other languages, though it sometimes generates outputs in English. GPT-3 requires more careful prompting, similarly to in English. (2) InstructGPT can summarize and answer questions about code more reliably than GPT-3 (though its answer here isn't quite correct). For the code QA example, GPT-3 does answer the question about 50% of the time.

Attribution: https://arxiv.org/pdf/2203.02155.pdf#page=15 (https://arxiv.org/pdf/2203.02155.pdf#page=15)

In the first example, the prompt (in French) is the Instruction

```
Write a short story about a frog who travels back in time to ancient Greece in
French.
```

without any Context.

Even if you don't understand French, you can see the repetitive nature of the Text Completion response.

In the second example (explain a piece of code)

- the Instruction is a request to explain some code
- the Context is the code

The response in the Text Completion behavior

- is not even an answer to the request
- the prompt appears to have been interpreted as the context for of a multiple choice question
    - it is **not wrong**, just not aligned with the user's intent

Fine-tuning an LLM to demonstrate Instruction Following behavior

The way to extend a pre-trained model's behavior to encompass a new Target task is with Transfer Learning.

The *Unsupervised Pre-Training + Supervised Fine-Tuning paradigm* is a type of Tranfer Learning

- Adapting a Pre-Trained LLM
- By Fine-Tuning with a small number of examples from the Target task

To get a LLM to exhibit Instruction Following behavior, we need to have a dataset of examples that demonstrates Instruction Following.

The examples in this dataset will be pairs or triples

- an *Instruction* part
    - tag "Instruction: "
- an optional *Context* part
    - tag "Input: "
- the *Response* part: the target/label
    - tag "Output: "

For example

- an Instruction part + Context part " $\mathbf{x}$

```
Instruction: Given a word, find out its length and its number of vowels.
Input: Word = "hello"
```

- a Target Output part $\mathbf{y}$

  Output: Length = 5, Number of vowels = 2

InstructGPT (https://arxiv.org/pdf/2203.02155.pdf) is a pre-trained GPT-3 that has been Fine-Tuned on a dataset that demonstrations of Instruction Following.

The chart above demonstrating Instruction Following (i.e., the one with the prompt in French to write a story)

- compares the Instruction following of pre-trained GPT-3
- with a Fine-Tuned GPT-3

The results are more satisfying.

Where do the Instruction Following demonstration examples come from ?

The demonstration examples for Instruction Following

- were manually constructed by human labelers

This is a very labor-intensive process.

In a separate module

- we will describe several efforts
- to *generate* training examples for the Instruction Following behavior
- using an LLM !

That is: we use a non-Instruction Following LLM

- to create examples
- on which to train an LLM to follow instructions

In [2]:
```
print("Done")
```

```
Done
```