Natural Language Processing Large Language Models Usage and Evaluation Patterns

Felipe Bravo-Marquez

November 23, 2023

Introduction

- Since the inception of Large Language Models, various patterns of use and evaluation of this technology have emerged.
- In this talk, we will try to organize these patterns and give a general overview of them.

Source:

`https://www.masayume.it/img/masayume/Large-Language-Models.jpg`

Recap: What is an LLM

- An autoregressive language model trained with a Transformer neural network on a large corpus (hundreds of bullions of tokens) and a large parameter space (billions) to predict the next word.
- It is usually later aligned to work as a user assistant using techniques such as Reinforcement Learning From Human Feedback [Ouyang et al., 2022] or supervised fine-tuning.
- Some are private (access via API or web browser): Google Bard, ChatGPT, etc.
- Others are open (model's weights can be downloaded): Llama, LLama2, Falcon, etc.
- The most remarkable feature of these models is their few-shot, one-shot, zero-shot learning capabilities.
- This means that they can learn new tasks without large amounts of human-annotated data.

Zero-shot, One-shot, and Few-shot Learning

The three settings we explore for in-context learning

Zero-shot

```
1  Translate English to French:      ← task description
2  cheese =>                         ← prompt
```

One-shot

```
1  Translate English to French:      ← task description
2  sea otter => loutre de mer        ← example
3  cheese =>                         ← prompt
```

Few-shot

```
1  Translate English to French:      ← task description
2  sea otter => loutre de mer        ← examples
3  peppermint => menthe poivrée      ←
4  plush girafe => girafe peluche    ←
5  cheese =>                         ← prompt
```

Traditional fine-tuning (not used for GPT-3)

Fine-tuning

```
1  sea otter => loutre de mer        ← example #1
```

↓ gradient update ↓

```
1  peppermint => menthe poivrée      ← example #2
```

↓ gradient update ↓ ... ↓

```
1  plush giraffe => girafe peluche   ← example #N
```

↓ gradient update ↓

```
1  cheese =>                         ← prompt
```

Talk Overview

- Despite the recency of this technology, its adoption has been tremendous in many areas.
- Below, we propose a simple categorization of the ways in which LLMs are used and evaluated.
- These patterns will serve as the narrative backbone of this presentation.

Usage Patterns

1. Fixed-knowledge Assistant
2. Knowledge-augmented Assistant
3. LLM-based Applications

Evaluation Patterns

- MTBench
- LLM Arena

Usage Pattern 1: Fixed-Knowledge Assistant

- In this pattern a user interacts with the LLM proving prompts as input and receiving a text as answer.
- The knowledge the LLM has access is limited to the corpus on which it was trained and the context given in the prompt.

Prompting

- Prompt Engineering : https://twitter.com/IntuitMachine/status/1727079666001870877
- Roles
- JSON outputs

```
{"role": "system", "content": "You are a helpful assistant desi
 {"role": "user", "content": "Who won the world series in 2020
```

- Chain of thought Prompting

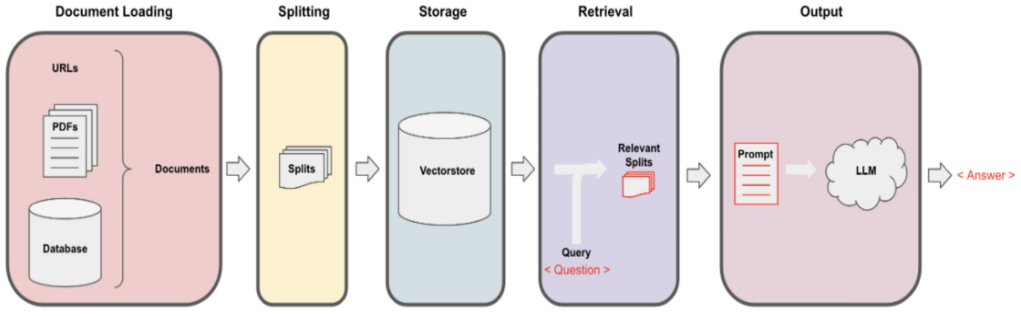
Knowledge-augmented Assistant

- Idea incorporate domain-scpefific knowledge not included during training.
- There two main patterns to achieve this:
  1. Retrieval-Augmented Generation (Vector Databases)
  2. Instruction Fine-Tuning

Setting the style, tone, format, or other qualitative aspects Improving reliability at producing a desired output Correcting failures to follow complex prompts Handling many edge cases in specific ways Performing a new skill or task that's hard to articulate in a prompt

Retrieval-Augmented Generation

- Rely on a Vector Database embed queries, retrieve relevant documents, append them into the prompt [Lewis et al., 2021].
- Many ideas of Informatrion Retrieval are employed here.
- `https://www.infoworld.com/article/3709912/vector-databases-in-llms-and-search.html`
- `https://learn.deeplearning.ai/vector-databases-embeddings-applications/lesson/1/introduction`
- `https://stackoverflow.blog/2023/10/09/from-prototype-to-production-vector-databases-in-generative-ai-ap`



Instruction Fine-Tuning

- Idea: instead of training the LM with raw text with next token prediction, train it with pairs of prompts and user-aligned answers.
- Paid Fine-Tuning (GPT-4??)
- OpenAI offers many more specific gpts: `https://openai.com/blog/introducing-gpts`
- Alpaca, Vicuna, Llama, Llama2
- https://blog.gopenai.com/paper-review-qlora-efficient-finetuning-of-quantized-llms-a3c857cd0cca

Datasets for Instruction Fine-Tuning

- Standford Alpaca Dataset (Vicuna)
- ShareGPT (Alpaca)
- Dolly-15K
- Orca Dataset

Parameter Efficient Fine Tuning

- Lora, QLora
- https://blog.gopenai.com/paper-review-qlora-efficient-finetuning-of-quantized-llms-a3c857cd0cca

Token-Incrementation

- Lora, QLora
- https://blog.gopenai.com/paper-review-qlora-efficient-finetuning-of-quantized-llms-a3c857cd0cca

Applications

- LLMs can be embbeded into any software via API calls. For example a Search Engine (you.com)
- https://gptstore.ai/

Autonomous Agents

- Agents are a special kind of LLMs application in which the LLM serves as the reasoning and planning component of the software.
- agent in the sense of perceiving an environment and taking actions to achieve goals.

LLMBench and LLm Arena

- Standard NLP evaluation: human annotated gold-labels and metrics.
- LLMS are intrinsically multi-task and not easily evaluated with this approach.
- Machines evaluating machines??
- MT-bench (categories)
- HuggingFace Open LLM Leaderboard
- LLM Arena

Questions?

Thanks for your Attention!

References I


Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. (2021).
Retrieval-augmented generation for knowledge-intensive nlp tasks.


Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022).
Training language models to follow instructions with human feedback.
Advances in Neural Information Processing Systems, 35:27730–27744.