LLM from deepLearning.ai

Saturday, May 18, 2024 1:34 PM

Size of LLMs

1. Transformers are the basis for most LLMs.
    a. Transformers help in parallelly compute the importance of each word vis-a-vis each word in the input
2. GPT is a decoder-only model.
3. Uses of LLMs
    a. Summarization
    b. Translation
    c. Named entity recognition
4. Small LLMs can be fine-tuned to perform well on focused tasks.

In Context Learning

1. Adding examples in the prompt often improves the performance - including the format in which the output is expected.

If 5-6 examples are not enough for the task at hand - it is recommended that we fine-tune the LLM.

Sampling in the output layer: This set of techniques determine how tokens are chosen by the decoder.

1. Greedy: In this approach, the word with the highest probability is picked. A problem with this approach is that it lacks creativity and may not sound very natural.
2. Random sampling: Here the words are picked at random from the softmax layer - weighted by their probabilities.
    a. Top-k: This is a hyperparameter that forces the model to pick randomly from the topK tokens from the softmax layer.
    b. Top-p: This is a parameter which forces the model to pick randomly from the top K tokens where the cumulative probability of K tokens is P.

Temperature: This is a hyperparameter that changes the softmax layer. Lower temperatures have a more peaked shape (concentrated probability in the top tokens) and is usually less creative.

LLM Pre-training

1. Petabytes of unstructured text is used to train LLMs.
2. To improve the quality of data, remove bias, or remove harmful content, usually a "Data Quality Filter" is applied to the training data which usually filters out 97-99% of the tokens.
3. **Encoder Only LLMs / Autoencoding models:**
    a. Tokens are masked and the model's objective is to correctly predict the masked token (denoising objective). For example, The teacher <MASK> the student. should be converted to The teacher teaches the student.
    b. The models creates a bidirectional context for the masked token to get the full picture.
    c. These models are usually good at:
        i. Sentiment analysis
        ii. Named entity recognition
        iii. Word classification
4. **Decoder Only LLMs / Autoregressive models**
    a. The training objective is to predict the next token based on the previous sequence of tokens (full language modeling).
    b. Good for:
        i. Text generation
5. **Sequence-to-Sequence models**
    a. The training objective usually varies. In T5, the training happens using span corruption where a random chunk of tokens is masked and replaced with another unique and new token (sentinel token). The objective of T5 is to map the sentinel token to the original chunk of tokens.
    b. Good for:
        i. Translation
        ii. Summarization
        iii. Question - Answering

Memory Optimization

Additional GPU RAM needed to train 1B parameters

	Bytes per parameter
Model Parameters (Weights)	4 bytes per parameter
Adam optimizer (2 states)	+8 bytes per parameter
Gradients	+4 bytes per parameter
Activations and temp memory (variable size)	+8 bytes per parameter (high-end estimate)
TOTAL	=4 bytes per parameter +20 extra bytes per parameter

Quantization is a method to statistically map 32-bit floating values (4 bytes) to 16-bit floating values (e.g. FP16, BFLOAT 16) or integers.

If the model cannot be loaded onto a single GPU, Fully Sharded Data Parallel (FSDP) is used to shard the model onto different GPUs. This technique offers almost linear memory reduction with the increase in the number of GPUs (sharding across 64 GPUs will result in 2^{-6}x memory use reduction. This however introduces communication between GPUs which leads to performance overheads.

Chinchilla paper argues that it is possible that LLMs can perform similarly with fewer parameters and longer training. The paper shows that compute-optimal models (a model that has the correct parameter size to dataset size ratio) perform better than other models such as GPT-3.
It also hints that we should have 20x the amount of data per parameter.
This paper has motivated teams to train smaller models for longer to get better performance.

Week 2

Finetuning

1. Disadvantages of in-context learning:
    a. Does not always provide good results, especially for smaller models.
    b. Uses up valuable context space
2. Developers have developed *prompt templates* that can convert large datasets into instruction prompt datasets for fine-tuning.
3. Models can be fine-tuned on specific tasks which leads to
    a. Better results for the specific task
    b. *Catastrophic forgetting*: As the parameters of the model changes, it becomes worse at other downstream tasks. For example, a model fine-tuned for sentiment analysis may forget named entity recognition.
        i. To maintain (if needed) accuracy across multiple tasks, the model should be fine-tuned on all those tasks (50k-100k data points per task).
        ii. Or, perform *Parameter Efficient Fine-tuning (PEFT)*, where most layers are frozen and only a few layers are tuned for the task.

FLAN: Fine-tuned Language Net

1. A specific set of instructions to perform instruction finetuning. For example, FLAN + PALM = FLAN-PALM.
2. Finetuned on 473 datasets across 146 task categories such as summarization.

Sample FLAN-T5 prompt templates

```
"samsum": [
    ("{dialogue}\n\nBriefly summarize that dialogue.", "{summary}"),
     ("Here is a dialogue:\n{dialogue}\n\nWrite a short summary!",
      "{summary}"),
     ("Dialogue:\n{dialogue}\n\nWhat is a summary of this dialogue?",
      "{summary}"),
     ("{dialogue}\n\nWhat was that dialogue about, in two sentences or less?",
      "{summary}"),
     ("Here is a dialogue:\n{dialogue}\n\nWhat were they talking about?",
      "{summary}"),
     ("Dialogue:\n{dialogue}\nWhat were the main points in that "
      "conversation?", "{summary}"),
     ("Dialogue:\n{dialogue}\nWhat was going on in that conversation?",
      "{summary}"),
]
```

Model Evaluation

1. Evaluating LLMs is challenging as the output is probabilistic.
2. *ROUGE* and *BLEU SCORE* are the top metrics that are used to evaluate the accuracy of LLMs.
    a. Rouge is used for text summarization. Here, the metric compares the output summary to one or more reference summaries.
        i. Compute the longest common subsequence (LCS) between the prediction and the label.
        ii. Compute Rouge-L F1 where recall = LCS / (number of unigrams in reference)
        iii. There are other implementations of the Rouge score.
    b. Bleu is used for text translation.
        i. Computes the average precision across a range of n-gram sizes

Benchmarks

1. Need to choose datasets that is largely different from the training data.
2. The dataset should be particular to the task that is being tested.
3. Examples: GLUE, SuperGlue, HELM, and so on.

Parameter efficient fine-tuning

1. Freeze most layers and update a small subset of parameters. Some also freeze all original layers and train new layers or components.
2. There are 3 types of PEFT:
    a. **Selective:** Fine-tune a subset of initial LLM parameters to fine-tune
        i. The results are mixed.
        ii. Significant tradeoff between parameter efficiency and compute efficiency.
    b. **Reparameterize:**
        i. **LoRA**: Low-Rank Adaptation of Large Language Models (LoRA)
            1) Freeze most of the original LLM weights.
            2) Inject 2 rank decomposition matrices (A&B).
            3) Train the weights of the small matrices
            4) During Inference:
                a) Weights = weights + B x A
            5) This is done usually in the self-attention layer.
    c. **Additive:** Freeze original layers and add new layers
        i. **Adapters:** Add new layers to the architecture of the model - typically inside the encoder or decoder components.
        ii. **Soft prompts (Prompt tuning):** Manipulate the inputs
            1) Add trainable tokens to the prompt that are learnt.

Concrete example using base Transformer as reference

Use the base Transformer model presented by Vaswani et al. 2017:

- Transformer weights have dimensions $d \times k = 512 \times 64$
- So $512 \times 64 = 32{,}768$ trainable parameters

In LoRA with rank $r = 8$:

- A has dimensions $r \times k = 8 \times 64 = 512$ parameters
- B has dimension $d \times r = 512 \times 8 = 4{,}096$ trainable parameters
- **86% reduction in parameters to train!**

Reinforcement learning from human feedback (RLHF)

1. Fine-tuned model based on human feedback writes better summaries than the base model, base human-level performance, initial fine-tuning and so on.
2. Define a criterion for the human labelers to evaluate the model outputs.
3. Once all the possibilities are labeled, we need to train a reward model which learns the total ordering of the completions.
4. Proximal policy optimization is used to update the weights
5. Reward hacking: Learn methods to improve the reward but deviate from the main objective (eg. adding irrelevant words)
    a. The fix is to compute KL Divergence Shift Penalty with the original model and add it to the reward model.
        i. KL is a mathematical function that computes the divergence of the probabilities from some baseline.
    b. To save space, PPO can be used with PEFT so that the base model can be reused.

Constitutional AI

1. Set a bunch of rules that the LLM can use to self-critique its responses.
2. It can help disambiguate competing instructions such as helpfulness vs. legality and so on.

LLM Optimization Techniques

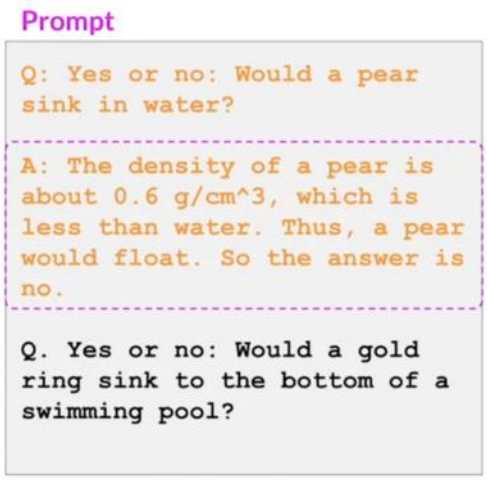
1. **Distillation**
    a. A smaller (student model) will learn from a larger fine-tuned model (teacher model).
    b. Typically works with encoder models but not with decoder models.
    c. Visit (https://snorkel.ai/llm-distillation-demystified-a-complete-guide/) for more details.
2. **Quantization**
3. **Pruning**
    a. Remove weights that are close to 0.

RAG: Retrieval Augmented Generation

1. Encode the query and use the encoded query to search a database.
2. Need to make sure the retrieved document fits the context window

Chain of Thought Prompting

Here, you teach the LLM how to reason in order to reach your answer. Look at the example prompt below that includes one example.



LLMs are still not very good at math. This limitation can be overcome using Program Assisted LLM (PAL) where the example in the prompt not only provides a chain of thought but a script to be generated. For the new problem, the LLM generates a similar script which is then passed to an interpreter for better answer.

LangChains can be used to template and memory in pipelines that use strategies and multiple interactions with LLMs and external APIs to fetch the best results.

ReAct prompting can be used to request LLMs strategize the above workflow.