Large Language Models

Jindřich Libovický

April 24, 2024
Outline

Decoder-only models

Model scaling

Introduction-base Fine-tuning

Societal impact of LLMs
Decoder-only models
1. **Embed input words.**
Get a sequence of continuous vectors.

2. **Contextualize input.**
Apply a sequence processing architecture and get contextual representation.

3. **Get some output.**
Typically classification or labeling.
From Sequence Labeling to Decoding

Transformers for sequence labeling

What if we labeled the sequence with what the next word is?
We need to modify the attention **not to attend the right context**

LMs estimate probability of a text: \( P(\text{“I am the Walrus”}) = \)

\[
P(\text{“I”}|[\text{BOS}]) \cdot P(\text{“am”}|[\text{BOS} \text{ I am}]) \cdot P(\text{“the”}|[\text{BOS} \text{ I am}) \cdot P(\text{“Walrus”}|[\text{BOS} \text{ I am the}) \cdot P(\text{“[EOS]”}} | [\text{BOS} \text{ I am the Walrus}])
\]
Reminder: Transformer Architecture

• Several layers (original paper 6)
• Each layer 2 sub-layers: self-attention and feed-forward layer
• Everything inter-connected with residual connections

**Feedforward-layer**

\[
F(X) = W_2 \text{ReLU}(W_1 X + b_1) + b_2
\]
Reminder: Multi-head scaled dot-product attention

Scaled dot-production attention

\[ Q = (q_1, \ldots, q_n): \text{queries}, \; K: \text{keys}, \; V: \text{values} \]

\[
\text{Attn}(Q, K, V) = \text{softmax} \left( \frac{QK^\top}{\sqrt{d}} \right) V
\]

Multi-head setup

\[
\text{Multihead}(Q, V) = \left( H_1 \oplus \cdots \oplus H_h \right) W^O
\]

\[ H_i = \text{Attn}(QW_i^Q, VW_i^K, VW_i^V) \]

\[ W_i^Q, W_i^K, W_iV \text{ head-specific projections} \]
Triangular Mask to Make Training Work

- Target is known at training: don’t need to wait until it’s generated
- Self attention can be parallelized via matrix multiplication
- Prevent attending the future using a mask
LM itself only computes probability

\[ P(w_{n+1}|w_0, \ldots, w_n) \]

We need an inference algorithm.

- In machine translation: beam search to search for maximum probable target
- With LMs random sampling — problem with exposure bias
  I.e., LM would behave strange if it sampled something improbable
- Solution: top-\(k\) sampling or nucleus sampling (= top \(x\) probability mass)
Inference

<\text{s}>

\begin{align*}
  & \text{sample} \\
  & \text{softmax} \\
  & P(w_0|<\text{s}>) \\
  \text{Transformer} \\
  \text{embed} \\
\end{align*}

\begin{align*}
  & \text{sample} \\
  & \text{softmax} \\
  & P(w_1|...) \\
  \text{Transformer} \\
  \text{embed} \\
\end{align*}

\begin{align*}
  & \text{sample} \\
  & \text{softmax} \\
  & P(w_2|...) \\
  \text{Transformer} \\
  \text{embed} \\
\end{align*}

\begin{align*}
  & \text{sample} \\
  & \text{softmax} \\
  & P(w_3|...) \\
  \text{Transformer} \\
  \text{embed} \\
\end{align*}

\ldots

\text{Large Language Models}

Decoder-only models  Model scaling  Introduction-base Fine-tuning  Societal impact of LLMs
Model scaling
**Timeline 2017–2018**

**06/2017**  Transformer Architecture


**02/2018**  ELMo – LM pre-training with RNNs for finetuning: 93.6M params.


**06/2018**  GPT-1 – Generative LM, in this scale for finetuning only, worse than *BERT*

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018
**Timeline 2017–2018**


**02/2019** GPT-2 – generative model 1.5B parameters

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019

**04/2020** GPT-3 – 175B parameters


**12/2022** ChatGPT
The gray text is **model input**, the black text is **continuation**.

<table>
<thead>
<tr>
<th>Poor English input</th>
<th>Good English output</th>
</tr>
</thead>
<tbody>
<tr>
<td>I eated the purple berries.</td>
<td>I ate the purple berries.</td>
</tr>
<tr>
<td>Thank you for picking me as your designer. I'd appreciate it.</td>
<td>Thank you for choosing me as your designer. I appreciate it.</td>
</tr>
<tr>
<td>The mentioned changes have done. or I did the alteration that you requested. or I changed things you wanted and did the modifications.</td>
<td>The requested changes have been made. or I made the alteration that you requested. or I changed things you wanted and made the modifications.</td>
</tr>
<tr>
<td>I'd be more than happy to work with you in another project.</td>
<td>I'd be more than happy to work with you on another project.</td>
</tr>
<tr>
<td>Please provide me with a short brief of the design you’re looking for and that’d be nice if you could share some examples or project you did before.</td>
<td>Please provide me with a brief description of the design you’re looking for and that would be nice if you could share some examples or projects you have done before.</td>
</tr>
<tr>
<td>The patient was died.</td>
<td>The patient died.</td>
</tr>
</tbody>
</table>

Source: GPT-3 Paper, Brown et al., 2020, Fig 3.17

- No update of the parameters
- The LM just continues in the text in the same style
Few shot learning success depend on how the **task is formulated**. (e.g., number of few-shot, examples needed, task description might be enough)

Art of finding the correct prompt = **Prompt Engineering**
Later discovered that smaller LMs are also few-shot learners...


Just replace in-context learning with clever finetuning, only consider predefined set of answers.
Figure 2: Eight examples of emergence in the few-shot prompting setting. Each point is a separate model. The ability to perform a task via few-shot prompting is emergent when a language model achieves random performance until a certain scale, after which performance significantly increases to well-above random. Note that models that used more training compute also typically have more parameters—hence, we show an analogous figure with number of model parameters instead of training FLOPs as the x-axis in Figure 1.


Finally, Figure 2H shows the Word in Context (WiC) benchmark (Pilehvar & Camacho-Collados, 2019), which is a semantic understanding benchmark. Notably, GPT-3 and Chinchilla fail to achieve one-shot performance of better than random, even when scaled to their largest model size of $\sim 5 \cdot 10^{23}$ FLOPs. Although these results so far may suggest that scaling alone may not enable models to solve WiC, above-random performance eventually emerged when PaLM was scaled to $2.5 \cdot 10^{24}$ FLOPs (540B parameters), which was much larger than GPT-3 and Chinchilla.
DeepMind's Chinchila experiments: longer training can compensate for parameter count.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. arXiv preprint arXiv:2203.15556, 2022, Figure 1

This trick used in Meta's LLAMA: GPT-3 quality with 30B parameters.

Emergent Properties of LMs (2)

- New abilities of LMs **emerge with size** – must be **discovered**
- Counterargument: Retrospectively with a **continuous metric**, everything is **gradual**

Introduction-base Fine-tuning
Finding the right prompt for a given task is alchemy...

What if we finetuned the LM for instruction-like prompts?

⇒ Finetuning on instructions, reinforcement learning with human feedback
   Started with InstructGPT, ChatGPT, …

Three steps of InstructGPT

**Step 1**
Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

**Step 2**
Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

**Step 3**
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is updated to update the policy using PPO.
Supervised Finetuning

- **Annotators write scripts** of conversation with the assistant
- Scripts are used for **direct finetuning**
- $10^5$–$10^6$ conversations are needed in this stage
Reinforcement Learning

Agent with parameters $\theta$, gets a distribution over possible actions $\pi_\theta$

- $a_t$ — Action taken in step $t$
- $r_t$ — Reward in step $t$

Problem with training: We cannot compute gradient $\partial r_t / \partial \pi_\theta$ because we sample a single action.
Words sampled from the model. I.e., the inference algorithm is a part of the training.

Simulated human feedback. A neural network prediction how would human annotators like the output.
Simulating Human Feedback

- Sample multiple $K$ answers for a prompt
- Annotators rank them $\Rightarrow \binom{K}{2}$ pair-wise comparisons
- Fine-tune BERT-like model to predict the comparison

Loss function over dataset of prompts $x$:

$$-\sum_{\text{prompt } x} \frac{1}{\binom{K}{2}} \sum_{y_{w}, y_{l}} \log \left[ \sigma(r(x, y_{w}) - r(x, y_{l})) \right]$$

where $w$ better than $l$
Learning from the Reward Function (Sketch)

- **Goal**: Optimize expected reward
- **We can approximate the gradient of reward with respect to the action distribution** (policy gradient algorithms):

\[
\frac{\partial}{\partial \pi_\theta} \mathbb{E}_{a \sim \pi_\theta} r(x, a) \approx \nabla \log \pi_\theta(a|x) \tilde{A}
\]

\(\tilde{A}\) is advantage (estimate reward gain = reward shifted by a smart constant)

- **Dark magic**: Construct a differentiable function that has a gradient like this + some good properties
- **InstructGPT uses Proximal Policy Optimization**
  

- **Additionally**: Minimize KL-Divergence from the supervised finetuned model
Intuition: Analogy to the Derivative of Cross-Entropy

\[
\frac{\partial L(P_y, y^*)}{\partial l} = - \frac{\partial}{\partial l} \log \frac{\exp l_y^*}{\sum_j \exp l_j} = 1_{y^*} - P_y(y^*)
\]

We want to have similar gradient but with **selected action and reward**
(i.e., positive advantage when reward is better than expectation)
Training Overview

Prompt from dataset/logs → Language model → Sampled answer → Proximal Policy Optim. → Reward model

Large Language Models
Decoder-only models  Model scaling  Introduction-base Fine-tuning  Societal impact of LLMs
It is not only ChatGPT

- Alpaca, Vicuna: Finetuned Meta’s LLAMA at Standford
- OpenAssistant: crowd-sourced dataset, models based on LLAMA and Pythia

...in May 2023 for most NLP tasks task-specific models are by large margin better.

Societal impact of LLMs
Problems with LLMs

• Changes in labour productivity will have economical consequences
• Labour productivity increases only somewhere:
  Only available in languages of Global North
• Biases against already underprivileged social groups
Training Data Might be Problematic

Crawling the Internet — not representative, people with extreme/wird opinions write more texts than the rest of society

Crowd-sourcing — using cheep labour, so-called gig economy – precarization of labour
Mary L Gray and Siddharth Suri. Ghost work: How to stop Silicon Valley from building a new global underclass. Eamon Dolan Books, 2019

Mining existing databases — unpaid labour, nontransparent “payment” for “free services”
Nick Couldry and Ulises A Mejias. The costs of connection: How data is colonizing human life and appropriating it for capitalism. Stanford University Press, 2020
The most **famous criticism** of LLMs (already from 2021)

- **Carbon footprint**: Different people benefit from the technology and different people carry the consequences
- Documented examples as **gender and racial bias** (in dubious applications in the US)
- **Nontransparent data curation**: authors decides about values in the text without much outside control

Narrated as a consequence of LMs **not capturing meaning**.
Assumption: Meaning is a (mathematical) relation on $M \subset I \times E$

$e \in E$ expression, $i \in I$ intent

Thought experiment: An octopus listens to human conversation over line and then simulates human conversation

- It never saw the world above the sea
- In only sees the expressions, no clue what intentions might be about

$\implies$ no way of understanding this type of meaning

Without meaning, (ethical) reasoning is impossible $\implies$ LMs inherently harmful
Arguments against the Octopus

- Alternative **established theories of meaning** that do not have this problem
- Empirical evidence rebuke the main points (beliefs and communication intents can be identified via probing)

Summary

1. Generating text = sampling from a language model
2. Scaling causes LMs to gain more capabilities
3. Instruction following is learned via reinforcement learning
4. Ethical problems: Reinforcing already existing societal issues

http://ufal.cz/courses/npfl1124