Large language models

Herman Kamper

2024-01, CC BY-SA 4.0

to-do TOC

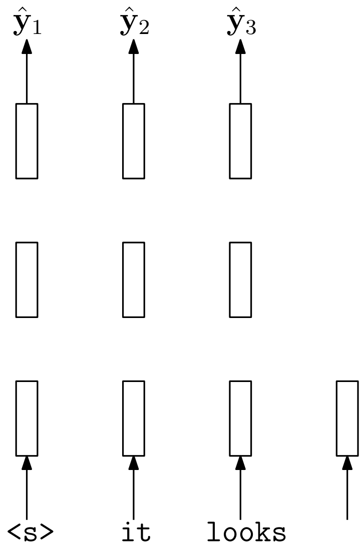
High-level introduction

Intro to large language models by Andrej Karpathy [slides]

GPT is just a transformer language model

Inference: Predicts one word at a time

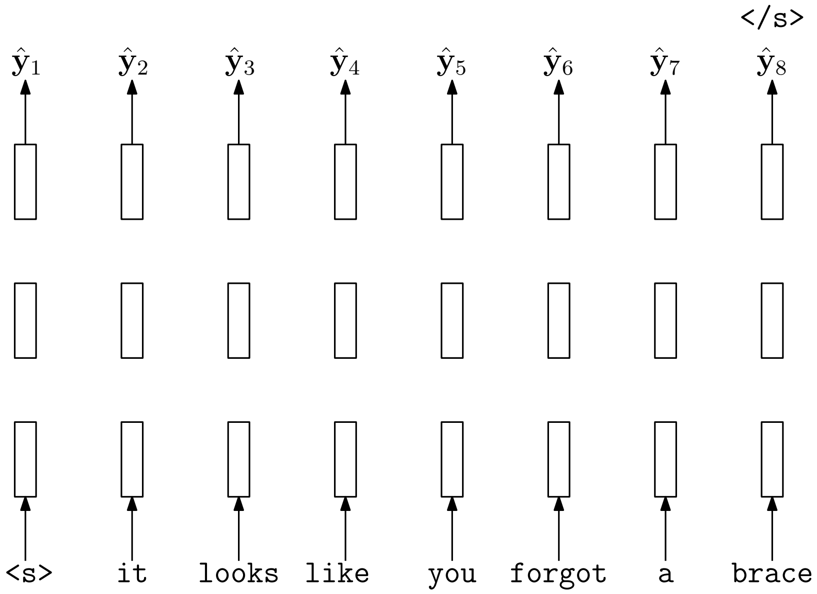
here: Don't do 12pt.



The model outputs the next word and then feeds that in autoregressively as its own input at the next time step.

There are different ways of sampling the output word from the predicted output distribution.

Training time: Predicts all the next words together



</s>
$\hat{\mathbf{y}}_1$ $\hat{\mathbf{y}}_2$ $\hat{\mathbf{y}}_3$ $\hat{\mathbf{y}}_4$ $\hat{\mathbf{y}}_5$ $\hat{\mathbf{y}}_6$ $\hat{\mathbf{y}}_7$ $\hat{\mathbf{y}}_8$
<s> it looks like you forgot a brace

Uses masking to make sure that what happens during training matches what happens during inference.

This is not an encoder-decoder model: it is just a decoder.

The basic architecture, I think, is still the one from (Radford et al., 2018).

What is different from what came before?

- Size of model
- Size of data: Compression of the internet (Karpathy)

From language model to assistant

Three steps to get to ChatGPT:

1. Pretraining: Standard next-word prediction task as above
2. Finetuning
3. Reinforcement learning from human feedback (RLHF)

2. Finetuning

Get high-quality data where we add `<prompt>` and `<answer>` tags:

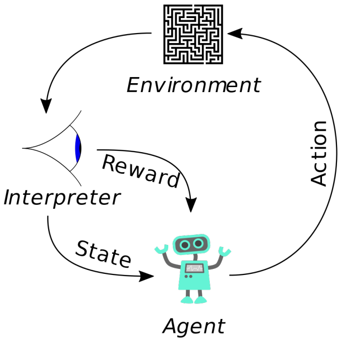
```
<prompt>
Can you help me with this code? It seems like there is
a bug.
print("hello world)
</prompt>

<answer>
It looks like you forgot to close the string passed to
the function print. You have to add a closing quote to
properly terminate the string. Here is the corrected
function:

print("hello world")

Let me know if I can help with anything else!
</answer>
```

You can think of this as a supervised task, but really it is still exactly the same task as the one used during pretraining. So the dataset is just swapped out, and training is continued.

It is a little bit amazing that all of the knowledge gained during pretraining isn't wiped out.

3. Reinforcement learning from human feedback (RLHF)

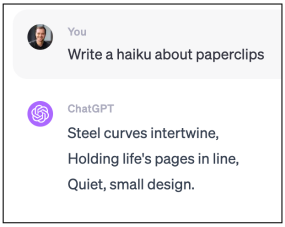
Reinforcement learning:[1]



What reward function should we use?

We learn a reward function by asking humans to rank possible outputs:[2]

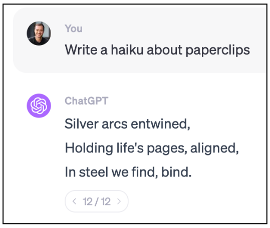
Simple example: it's much easier to spot a good haiku than it is to generate one:


You

Write a haiku about paperclips

ChatGPT

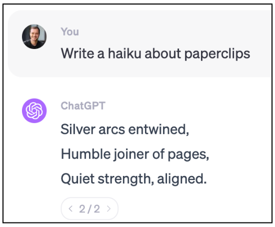
Steel curves intertwine,
Holding life's pages in line,
Quiet, small design.


You

Write a haiku about paperclips

ChatGPT

Silver arcs entwined,
Holding life's pages, aligned,
In steel we find, bind.


You

Write a haiku about paperclips

ChatGPT

Silver arcs entwined,
Humble joiner of pages,
Quiet strength, aligned.

In this way they get even more assistant-like behaviour from the model (Ziegler et al., 2020).

But Rafailov et al. (2023) and others say that it isn't really necessary to use reinforcement learning to update the model with the learned reward function: you can do it directly through supervised learning.

[1]Figure from Wikipedia.
[2]Figure by Andrej Karpathy.

PPO links

Acknowledgments

. . . .

Further reading

. . .

References

A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training", *OpenAI*, 2018.

R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," in *NeurIPS*, 2023.

D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving, "Fine-tuning language models from human preferences," *arXiv*, 2020.