An overview of mathematical foundations of Transformer analysis

Mishka (Michael Bukatin)

github:anhinga

Dataflow Matrix Machines project

Hopf algebra seminar led by András Kornai

November 27, 2023

Structure of the talk

- Technical part
  - Transformer code
  - Tokenization
  - Autoregressive model
  - Text as a matrix
  - Residual stream
  - Part of “A Mathematical Framework for Transformer Circuits”
- Overview part

GPT-2 pedagogical implementations

It's important to read some implementation of Transformers and to play with it a bit.

For example:

- `https://github.com/karpathy/minGPT` - more pedagogical
- `https://github.com/karpathy/nanoGPT` - more professional

Tokenization

Tokens: frequent sequences of letters including single letters

Words, fragments of words, words with attached whitespace, ...

Tokens are mapped to vectors

Do we want a more linguistic-friendly tokenization?

```
config.vocab_size = 50257 # GPT-2 model vocabulary
```

Autoregressive decoder-only model

Given tokens $t_1, \dots, t_n$ model makes predictions:

$t_1 \Rightarrow t_2^{new}$
$t_1, t_2 \Rightarrow t_3^{new}$
$\dots$
$t_1, \dots, t_n \Rightarrow t_{n+1}^{new}$

t_i^{new} is a probability distribution on all tokens, $t^{name} \mapsto \rho_i(t^{name})$

During training: loss $= \sum_{i=1}^{n} -log(\rho_{i+1}(t_{i+1}^{correct}))$

During autoregressive inference: sample t_{n+1} from t_{n+1}^{new} probability distribution[1] and pass $t_1, \dots, t_n, t_{n+1}$ to the input of the model to predict t_{n+2}^{new}, etc. **This is a recurrent machine.**

[1] all predictions except for the last one are ignored

Text as a matrix

The embedding maps each token to a vector

It maps a sequence of tokens to a matrix

Matrix rows are vectors corresponding to tokens

The number of rows is the number of tokens in the sequence

Transformer layers **update** this matrix:

```
def forward(self, x):
    x = x + self.attn(self.ln_1(x))
    x = x + self.mlpf(self.ln_2(x))
    return x
```

The stream of changing x is called **residual stream**

"A Mathematical Framework for Transformer Circuits"

I cover only the part Adam Nemecek is relying upon[2]

The rest belongs to the overview part

https://transformer-circuits.pub/2021/framework/index.html

Neel Nanda, "A Walkthrough of A Mathematical Framework for Transformer Circuits", 2 hours 50 minutes:
https://www.youtube.com/watch?v=KV5gbOmHbjU

Tensor product of matrices (matrix direct product):
https://en.wikipedia.org/wiki/Kronecker_product

https://en.wikipedia.org/wiki/Vectorization_(mathematics)#Compatibility_with_Kronecker_products

[2]the links in the blue are the links I am planning to visit during the talk

Main ideas

Papers usually describe implementations written for efficiency

Rewrite without thinking about efficiency in order to understand

Heads are independent from each other:

- Low-rank bottlenecks are ubiquitous
- Can compose heads from different layers

X is a rectangular matrix $M \times N$:

- AX is well-formed, A is a square matrix $M \times M$
- But the XW_{OV} for the output-value circuit

Neel Nanda: use of tensor product is optional

Structure of the overview part

This is a selection from a huge field:

- more on induction heads and such
- more on low-rank motifs
- interplay of Transformers and autoencoders
- grokking and phase transitions
- in-context learning and trying to improve Transformers
- autoregressive models as simulators
- GPT-4-related topic
  - Mixture-of-Experts and Transformers
  - Impressions from base-GPT-4
  - state of advanced fine-tuning
- misc

more on induction heads

Continue reading this paper and/or watching Neel Nanda video

`https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html`

`https://transformer-circuits.pub/2021/exercises/index.html`

`https://transformer-circuits.pub/2021/videos/index.html`

more on low-rank motifs

The main method of **lightweight fine-tuning** instead of honest fine-tuning: Edward Hu et al. “LoRA: Low-Rank Adaptation of Large Language Models”, `https://arxiv.org/abs/2106.096851`

Used everywhere for all kinds of applications, including

“LoRA Fine-tuning Efficiently Undoes Safety Training in Llama 2-Chat 70B”, `https://arxiv.org/abs/2310.20624`

“MultiLoRA: Democratizing LoRA for Better Multi-Task Learning”, `https://arxiv.org/abs/2311.11501`

People should play with really large number of attention heads, i.e. with very low-dimensional heads, see what happens.

interplay of Transformers and autoencoders 1

The most recent paper on `https://transformer-circuits.pub/` :

"Towards Monosemanticity: Decomposing Language Models With Dictionary Learning", Oct 2023,
`https://transformer-circuits.pub/2023/monosemantic-features/index.html`

Extracted features are linear combinations of neurons;
obtained by connecting a Transformer and a sparse autoencoder

interplay of Transformers and autoencoders 2

Representation learning:

"AdaVAE: Exploring Adaptive GPT-2s in Variational Auto-Encoders for Language Modeling", https://arxiv.org/abs/2205.05862

The following Oct 2023 work by John David Pressman is built on top of that: "Revealing Intentionality In Language Models Through AdaVAE Guided Sampling":

https://www.lesswrong.com/posts/4Hnso8NMAeeYs8Cta/revealing-intentionality-in-language-models-through-adavae

https://github.com/JD-P/minihf/tree/adavae-moe

grokking and phase transitions

My notes:

`https://github.com/anhinga/2022-notes/tree/main/Grokking-is-solved`

Discovered in 2021 by an OpenAI team, Alethea Power et al., "Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets", `https://arxiv.org/abs/2201.02177`

Solved by Neel Nanda et al. in 2022 (see my notes).

At least, *that's my position*, however, there are a bunch of proposed alternative approaches to understanding the Grokking phenomenon, including even this generalization: "Grokking Beyond Neural Networks: An Empirical Exploration with Model Complexity", `https://arxiv.org/abs/2310.17247`

in-context learning and trying to improve Transformers 1

Many people looking from different angles independently discovered that **attention layers perform "large-size steps of gradient descent" during one inference pass**.

For example, "you can think of softmax attention as implementing a single, big gradient step of some energy function and that training transformers is akin to meta-learning how to best tune a stack of attention and feed-forward modules to perform well on some auxiliary (meta-)task(s)"
writes Matthias Bal in

`https://mcbal.github.io/post/deep-implicit-attention-a-mean-field-theory-perspective-on-attention-mechanisms/`

See also his other blog format articles on "Transformers Are Secretly Collectives of Spin Systems" and such at
`https://mcbal.github.io/`

in-context learning and trying to improve Transformers 2

Let's look at a couple of papers.

It is technically convenient to consider **linear attention**, that is attention without **softmax**.

Then one's has associativity here $(QK^T)V = Q(K^T V)$ which allows one to do various interesting things.

Note that linear-attention-only Transformer is still not a linear model, but a polynomial one, hence sufficiently expressive.

First, look at "Exploring the Space of Key-Value-Query Models with Intention" by Garnelo and Czarnecki, `https://arxiv.org/abs/2305.10203`

in-context learning and trying to improve Transformers 3

"Our goal is to determine whether there are any other stackable models in KVQ Space that Attention cannot efficiently approximate, which we can implement with our current deep learning toolbox and that solve problems that are interesting to the community. Maybe surprisingly, **the solution to the standard least squares problem satisfies these properties.** A neural network module that is able to compute this solution not only enriches the set of computations that a neural network can represent but is also provably a strict generalisation of Linear Attention. Even more surprisingly the computational complexity of this module is exactly the same as that of Attention, making it a suitable drop in replacement."

"the solution to a standard regularised least squares minimisation problem": $Q[KK^T + \alpha I]^{-1} K^T V$.

(cf. also "Multiplicative Interactions and Where to Find Them" (ICLR 2020), `https://openreview.net/forum?id=rylnK6VtDH`)

in-context learning and trying to improve Transformers 4

Garnelo-Czarnecki paper has exactly one reference to it:

von Oswald et al., "Uncovering mesa-optimization algorithms in Transformers", `https://arxiv.org/abs/2309.05858`

A very important, but not well-written paper, see also `https://twitter.com/oswaldjoh/status/1701873029100241241` and `https://gonzoml.substack.com/p/uncovering-mesa-optimization-algorithms`

It has plenty of references to other independent discoveries that attention layers perform "large-size steps of gradient descent" during one inference pass.

But then it is trying to explain how earlier attention layers set up an optimization task on the fly, and subsequent attention layers solve it.

in-context learning and trying to improve Transformers 5

And it also explores replacing an attention layer by a (different) least-squares solver.

Page 2 contains the summary; read it, it is well written, explains the high level: `https://arxiv.org/pdf/2309.05858.pdf`

In-context/few-shot learning was first discovered in GPT-3, but the claim here is that this can work even in **really small models.**

See also my notes at `https://dmm.dreamwidth.org/76107.html`

autoregressive models as simulators 1

Even with GPT-2, I had a feeling that GPT-2 sampled an interlocutor for me from a distribution which depended on my initial prompt, and then I was having a chat with that interlocutor.

In September 2022, "Janus" published a theory which made this much more precise.

I took extensive notes on that theory:

`https://github.com/anhinga/2022-notes/tree/main/Generative-autoregressive-models-are-similators`

Rough approximation of what's going on: "These models are simulators, they generate various virtual entities (which tend to be rather short-lived at the moment), and users are interacting with those virtual entities."

autoregressive models as simulators 2

Inference runs are simulations.

Simulations and virtual characters have widely differing properties from run to run.

The art of “prompt engineering” is the art of tuning the simulation, so that it has the properties one currently wants.

This understanding might be the most important conceptual breakthrough of 2022.

On Nov 8 Murray Shanahan and “Janus” co-authored a paper in *Nature*: “Role play with large language models”, https://www.nature.com/articles/s41586-023-06647-8

Mixture-of-Experts and Transformers (cf. GPT-4 rumors)

Romors are that GPT-4 is a **mixture-of-experts**:
`https://en.wikipedia.org/wiki/Mixture_of_experts`

Some people think this means that it's 8 GPT-3's working in parallel, with some of them running each time depending on the context, but I don't think so.

Mixture-of-experts just means these days that a part of the overall model is activated during each inference run, basically, a **dynamically computed sparse mask** is applied on each run, otherwise the architecture can be rather arbitrary.

Huge diversity of hybrids between Mixture-of-Experts approach and Transformers. This is a field of science by itself:
`https://github.com/XueFuzhao/awesome-mixture-of-experts`

Impressions from base-GPT-4

It is difficult to obtain access to base-GPT-4 (the one before RLHF with all its trade-offs has been applied), but there is a procedure for interested researchers to apply.

base-GPT-4 is a very rich powerful model in the right hands, and it is much more versatile than GPT-4, but there is no handholding.

"Janus" kindly shared the impressions from their experience with **base-GPT-4** in the answers here:

https://www.lesswrong.com/posts/tbJdxJMAiehewGpq2/impressions-from-base-gpt-4

state of advanced fine-tuning (GPT-3.5 vs. GPT-4)

Everybody is trying to sell lightweight fine-tuning, something inexpensive like LoRA (Low-Rank Adaptation), but in such a fashion that the quality is high.

The reports for available GPT-3.5 fine-tuning were glowing ("GPT-4-like performance on the narrow area you fine-tune for").

However, GPT-4 fine-tuning currently is experiencing difficulties similar to early GPT-3 fine-tuning difficulties.

"Preliminary results indicate that GPT-4 fine-tuning requires more work to achieve meaningful improvements over the base model compared to the substantial gains realized with GPT-3.5 fine-tuning."

misc

A Swiss paper claiming to be able to do it well without skip connections:

"Simplifying Transformer Blocks", `https://arxiv.org/abs/2311.01906` and `https://github.com/bobby-he/simplified_transformers`

Claim: "a family of white-box transformer-like deep network architectures, named CRATE (Coding RAte reduction TransformEr), which are mathematically fully interpretable":

"White-Box Transformers via Sparse Rate Reduction: Compression Is All There Is?", `https://arxiv.org/abs/2311.13110` and `https://github.com/Ma-Lab-Berkeley/CRATE`