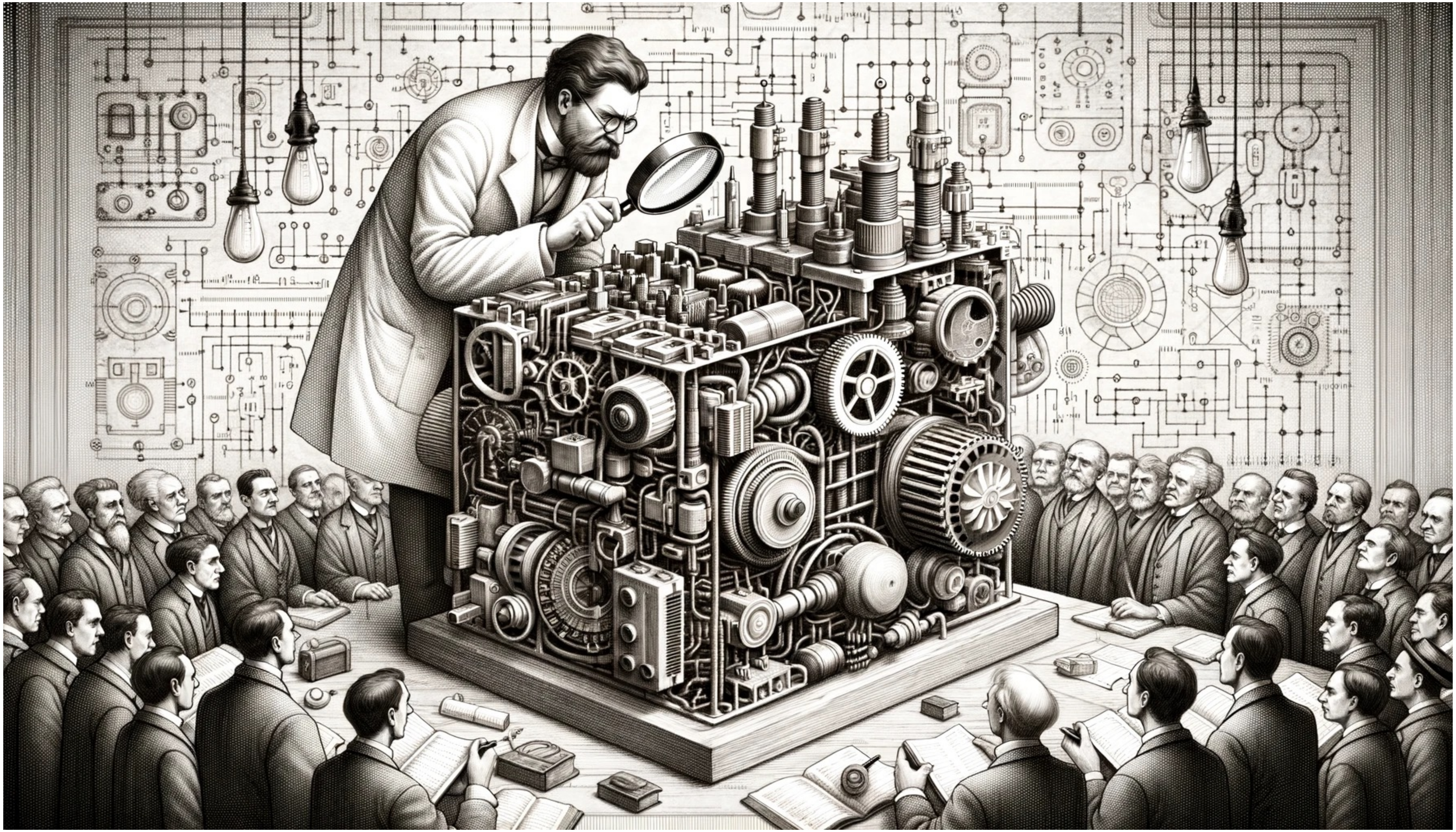
IOS
INSTITUT FÜR OPTISCHE SYSTEME

HTWG Hochschule Konstanz
Technik, Wirtschaft und Gestaltung



LLMs a Deep Dive into the mechanic

Brown Bag Seminar IOS

Oliver Dürr

Principles of ChatGPT

Describe your technology in one sentence.

I am a Generative Language model based on the transformer architecture that uses self-attention mechanisms and is trained with maximum likelihood principle to predict the next token in a sequence of tokens and generate new text.

Token = Part(s) of Word

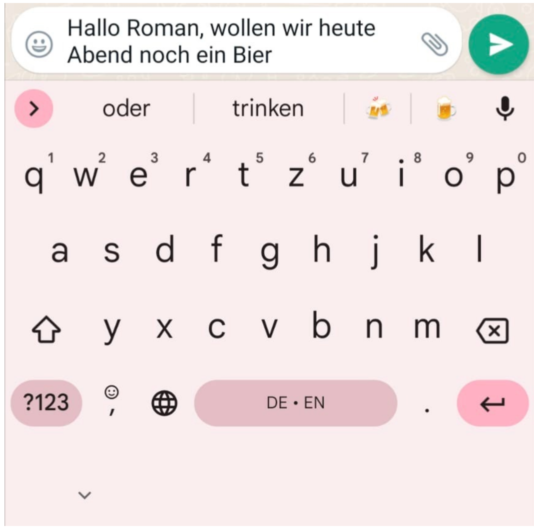
The Task*

Describe your technology in one sentence.

I am a Generative Language model based on the transformer architecture that uses self-attention mechanisms and is trained with maximum likelihood principle to predict the next token in a sequence of tokens and generate new text.

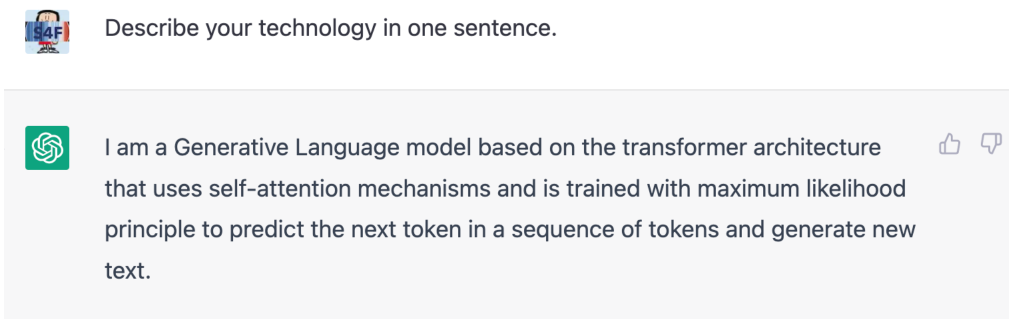
*Tokens later.

Generative Language Models: “predict next token”



Generative Language models:

Like you pressing (most) probable suggestion



Steps 1: Describe your technology in one sentence. → I
Steps 2: Describe your technology in one sentence. I → am
Steps 3: Describe your technology in one sentence. I am → a
Steps 4: Describe your technology in one sentence. I am a → generative

Step 36 Describe your technology in one sentence. I am a … new text. → END

Sampling Repeated

Prompt

The weather is really nice today. I'm thinking about going for a

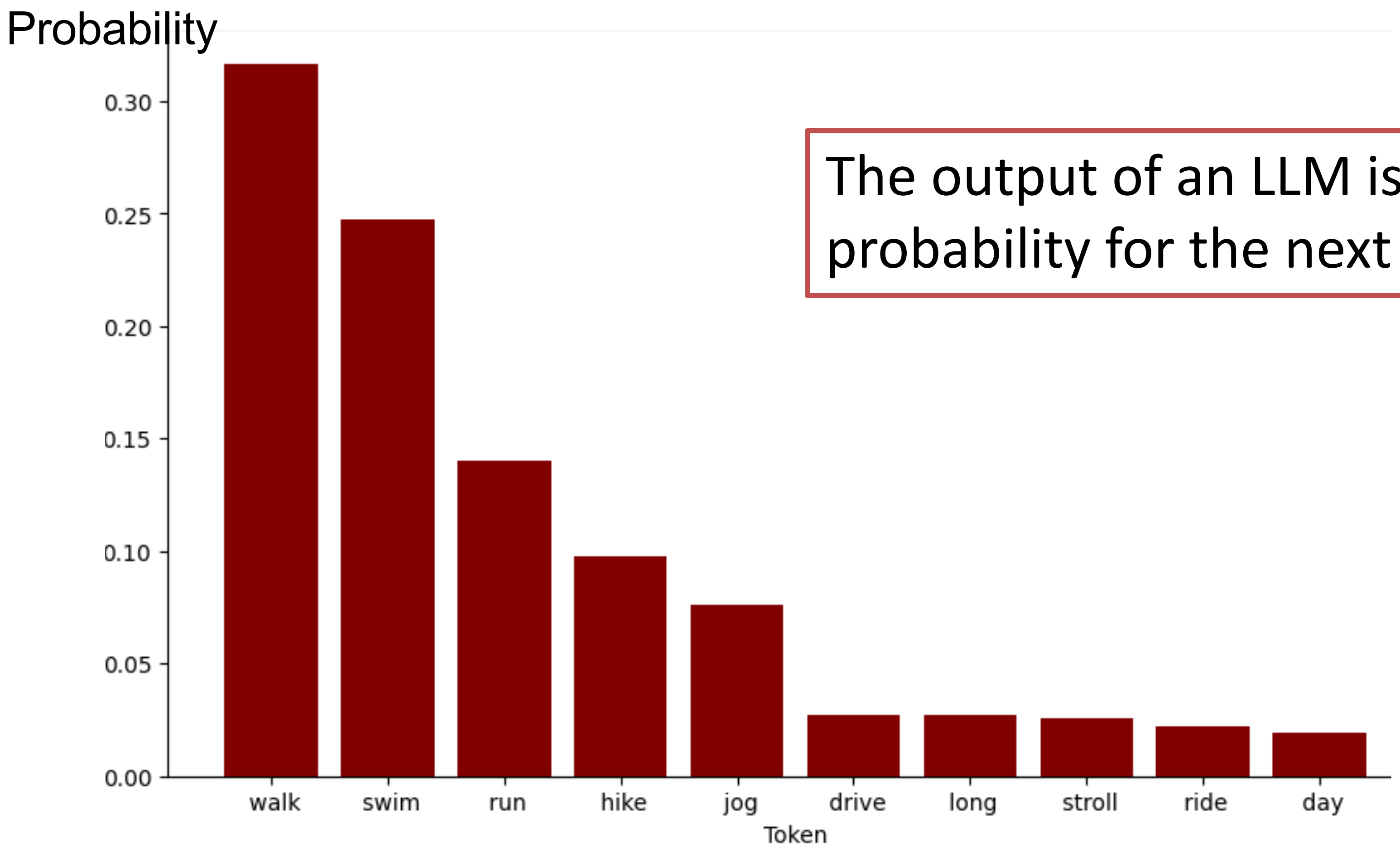
```
' day ...',
' walk ....',
' good ...',
' swim ...",
' swim ...',
' jog ...',
' hike ...',
' run ...',
' swim ...',
' jog ...',
```

10 Calls to ChatGPT, resulting in different results.

https://github.com/oduerr/llm_playground/tree/main/hugging_face/sampling_from_LLM

Probabilities for next word

Prompt
The weather is really nice today. I'm thinking about going for a



The output of an LLM is the probability for the next word

https://github.com/oduerr/llm_playground/tree/main/hugging_face/sampling_from_LLM

Training / Maximum Likelihood Prinziple

Describe your technology in one sentence.

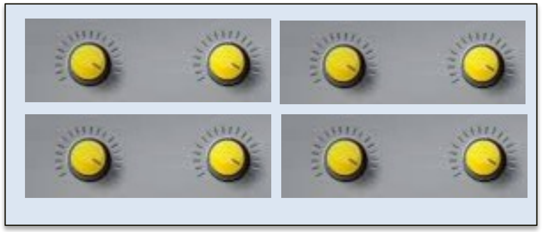
I am a Generative Language model based on the transformer architecture that uses self-attention mechanisms and is trained with maximum likelihood principle to predict the next token in a sequence of tokens and generate new text.

LLM are probabilistic models

The weather is really nice today. I'm thinking about going for a

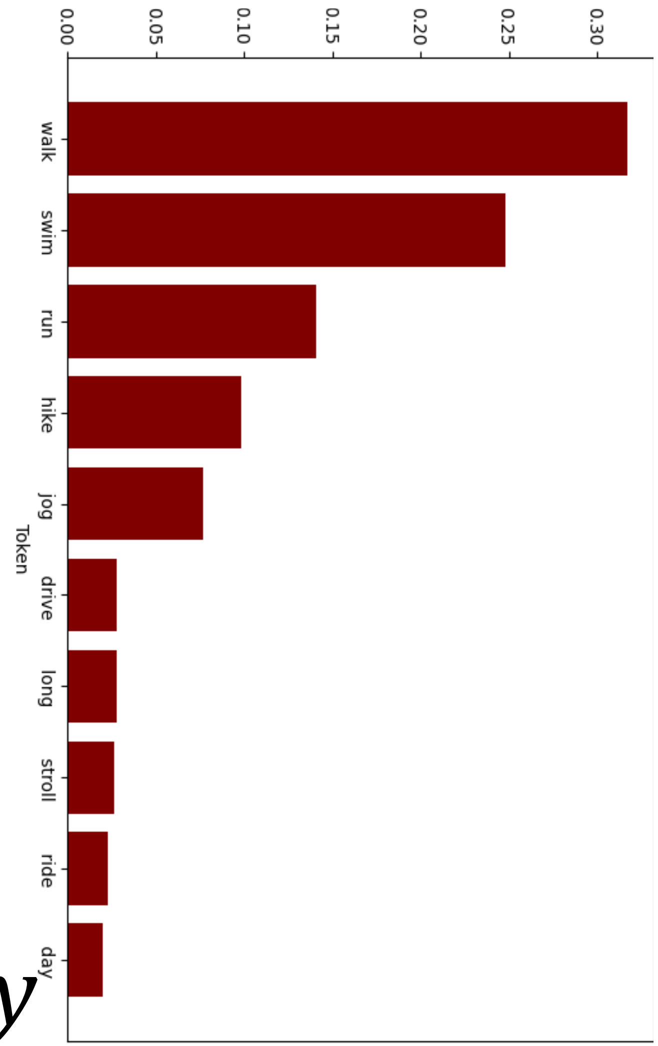
$\boldsymbol{x}$

LLM



A model with parameters $\boldsymbol{\theta}$

Probability for y



y

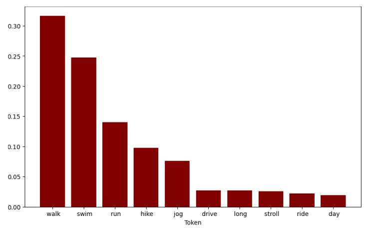
Quiz: Number of parameters in GTP-3.5?

Quiz: Write this in math

$$p_{\boldsymbol{\theta}}(y|\boldsymbol{x})$$

Training: To predict the next token

- Training data:
  - “Whole Internet”: arXiv + StackOverflow + …
    - LLaMMa*: 1T (1E12) Token
- Take samples
  - Take a text example where to know the answer
    - The weather is really nice today. I'm thinking about going for a walk
  - Use input x=“The weather is really nice today. I'm thinking about going for a”
  - Observed value y=“walk”
  - Output of model $p_{\boldsymbol{\theta}}(y|\boldsymbol{x})$



  - Tune the model so that $p_{\boldsymbol{\theta}}(\text{“walk”}|\boldsymbol{x})$ is high

*LLaMA arXiv:2302.13971

Training: Loss Function = Likelihood of Data

Quiz:

- What is the worst model? What is $p_{\boldsymbol{\theta}}(y = \text{“}walk\text{”}|\boldsymbol{x}) =???$
  - $p_{\boldsymbol{\theta}}(y = \text{“}walk\text{”}|\boldsymbol{x}_i) = 0$
  - $\log 0_+ = -\infty$
- What is the best model (for that single example)?
  - $p_{\boldsymbol{\theta}}(y = \text{“}walk\text{”}|\boldsymbol{x}_{\boldsymbol{i}}) = 1$
  - $\log 1 = 0$
- For that single example with observation y_i the following is a good loss
  - $-\log p_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_{\boldsymbol{i}})$
- Minimization of that loss averaged over a batch of 4M tokens (negative log likelihood, NLL)
  - minimizing the NLL = maximizing the likelihood

Tokenization and Embedding

Describe your technology in one sentence.

I am a Generative Language model based on the transformer architecture that uses self-attention mechanisms and is trained with maximum likelihood principle to predict the next token in a sequence of tokens and generate new text.

Tokenization Deep Learning need numbers

- Deep Learning system need numbers
  - x_i=“The weather is really nice today. I'm thinking about going for a”

- Simple Tokenization(ASCII)

```
cook@pop-os:~$ ascii -d
    0 NUL    16 DLE    32        48 0    64 @    80 P    96 `   112 p
    1 SOH    17 DC1    33 !      49 1    65 A    81 Q    97 a   113 q
    2 STX    18 DC2    34 "      50 2    66 B    82 R    98 b   114 r
    3 ETX    19 DC3    35 #      51 3    67 C    83 S    99 c   115 s
    4 EOT    20 DC4    36 $      52 4    68 D    84 T   100 d   116 t
    5 ENQ    21 NAK    37 %      53 5    69 E    85 U   101 e   117 u
    6 ACK    22 SYN    38 &      54 6    70 F    86 V   102 f   118 v
    7 BEL    23 ETB    39 '      55 7    71 G    87 W   103 g   119 w
    8 BS     24 CAN    40 (      56 8    72 H    88 X   104 h   120 x
    9 HT     25 EM     41 )      57 9    73 I    89 Y   105 i   121 y
   10 LF     26 SUB    42 *      58 :    74 J    90 Z   106 j   122 z
   11 VT     27 ESC    43 +      59 ;    75 K    91 [   107 k   123 {
   12 FF     28 FS     44 ,      60 <    76 L    92 \   108 l   124 |
   13 CR     29 GS     45 -      61 =    77 M    93 ]   109 m   125 }
   14 SO     30 RS     46 .      62 >    78 N    94 ^   110 n   126 ~
   15 SI     31 US     47 /      63 ?    79 O    95 _   111 o   127 DEL
```

  - x_i=“The weather is really nice today. I'm thinking about going for a”
  - x_i = (84, 104, 101,32, 119,...,32,97)

- And what with “Köche präferieren süße Schokolädentörtchen.”

Tokens in LLM

ChatGPT works with tokens not with words. It has 50257 different tokens.

https://platform.openai.com/tokenizer

Tokens are common sequences of characters found in text.

[30400, 527, 4279, 24630, 315, 5885, 1766, 304, 1495, 13]

Mein Luftkissenfahrzeug ist voller Aale

Mein Luftkissenfahrzeug ist voller Aale

Emergence or stochastical parrot 🦜?

Emergence or stochastical parrot ���?

差不多

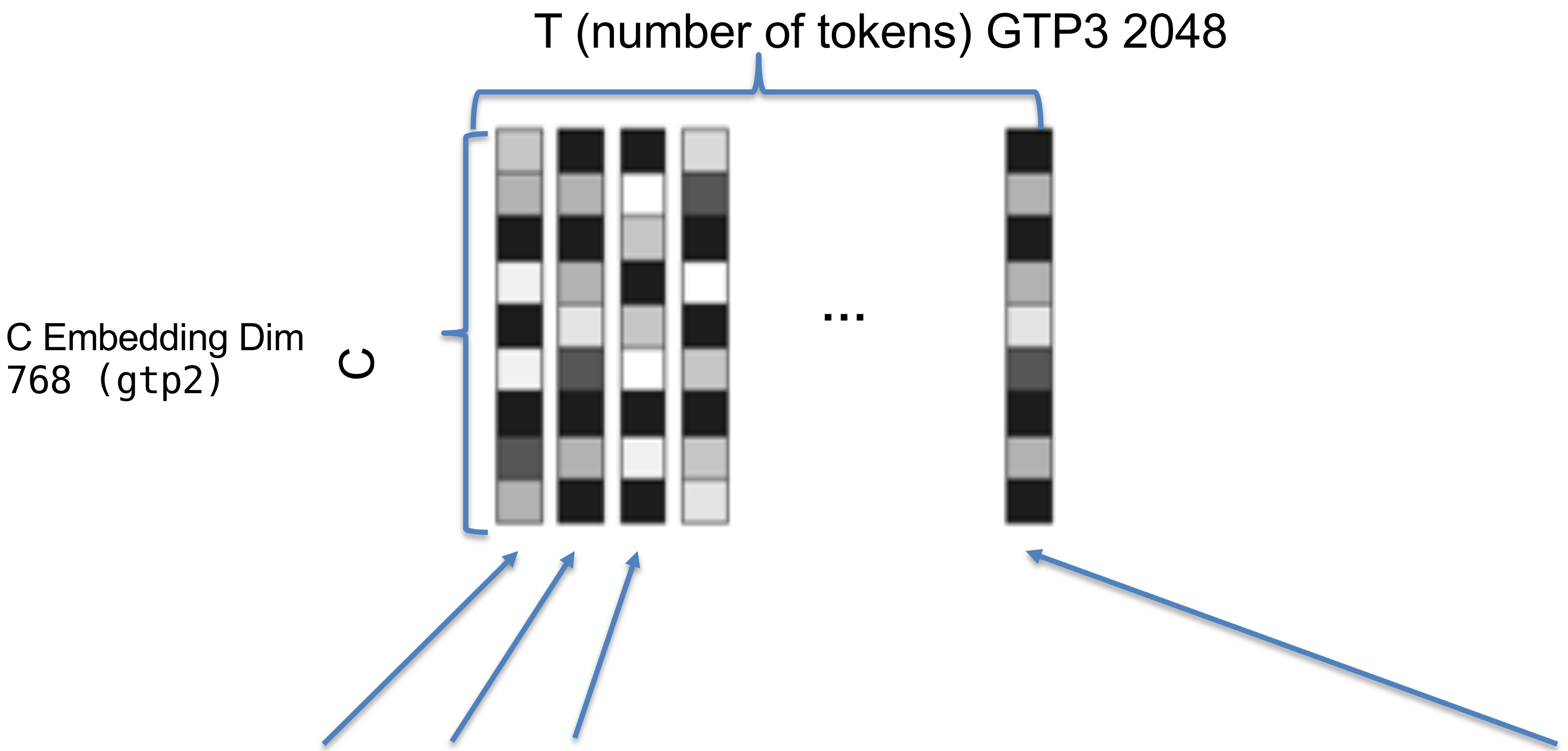
��不多

[17597, 106, 16937, 43240]

English is coded shorter (since most of text is in English for BPE Algorithm)
Fun Fact: "spell lollipop backwards" used to trick ChatGPT(before 25 Sep Version)

Understand Text as a Sequence of Vectors

- After Tokenization and Embedding Text looks like



[791, 9282, 374, 2216, 6555, 3432, 13, 358, 2846, 7422, 922, 2133, 369, 264, 198]

`Tokens are common sequences of characters found in text.`

In DL we usually take batches the primary object of interest
a tensors of size 3 with (B, T, C) dimensions

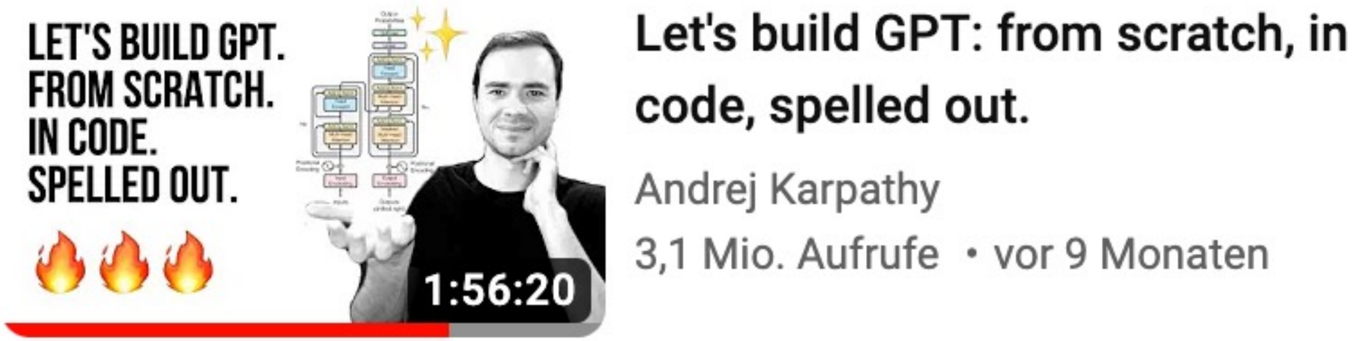
The architecture

Describe your technology in one sentence.

I am a Generative Language model based on the transformer architecture that uses self-attention mechanisms and is trained with maximum likelihood principle to predict the next token in a sequence of tokens and generate new text.

Sources for Transformer

- Live Codeing (and explanation) Andrej Karparthy nano-GPT https://youtu.be/kCc8FmEb1nY

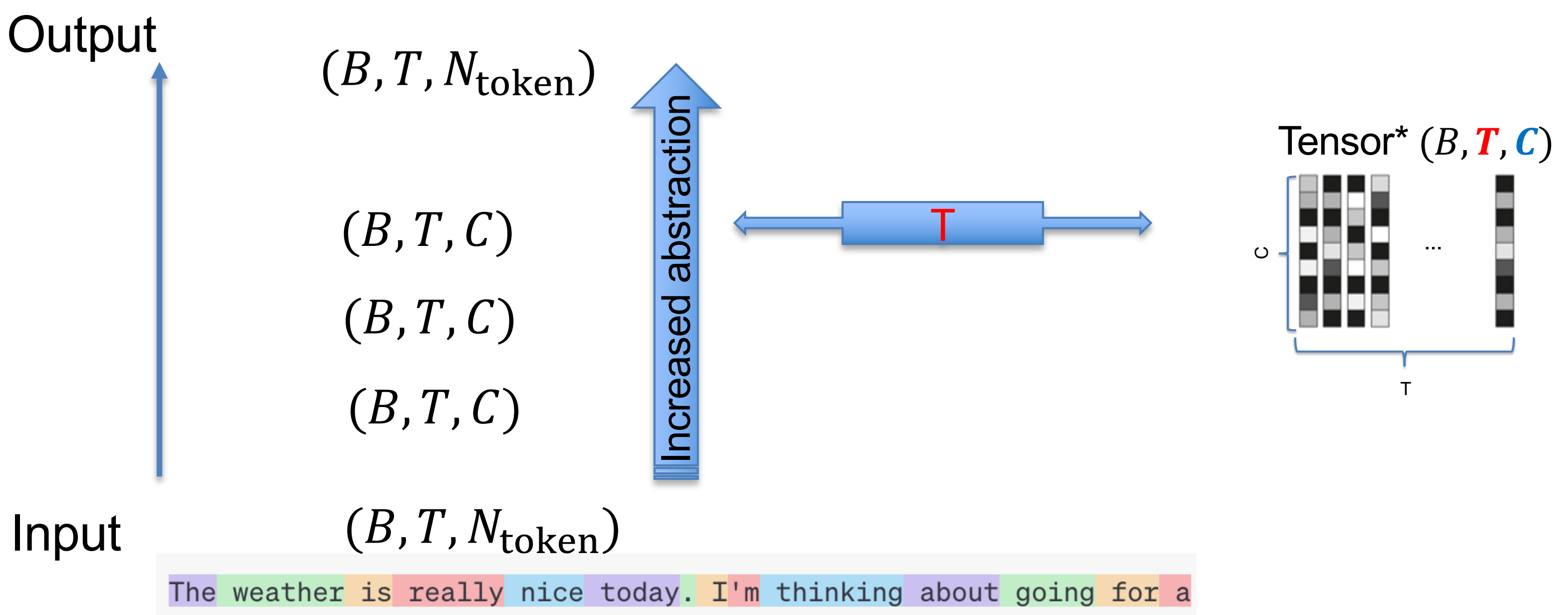


Further Resources

- CS25 Andrej Karparty Introduction to Transformers
  - Nice intuition with message parsing https://www.youtube.com/watch?v=XfpMkf4rD6E
- Lennart Svensson: https://youtu.be/0SmNEp4zTpc
  - Good intuition (weighted average, mathematical sound)
- Ava Soleimany: https://www.youtube.com/watch?v=QvkQ1B3FBqA
  - Good intuition (Search query idea)
- https://jalammar.github.io/illustrated-transformer/
  - Nice illustrations
- CS25 Andrej Karparty Introduction to Transformers
  - Nice intuition with message parsing https://www.youtube.com/watch?v=XfpMkf4rD6E

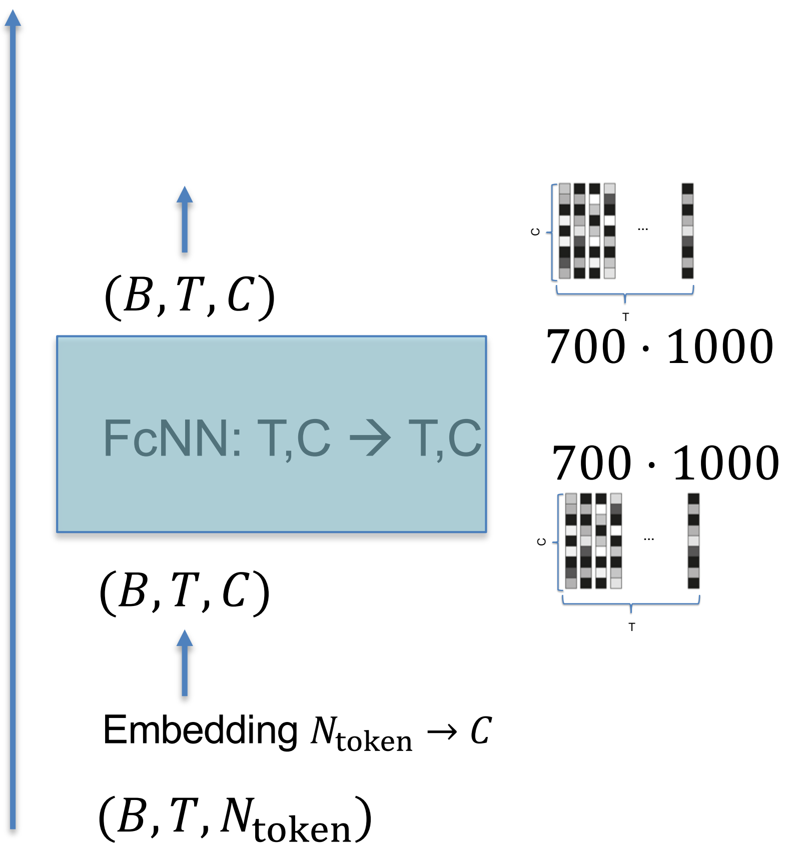
In Language: need for context

- Example
  - <u>Server</u>, can I have the check?
  - Looks like I just crashed the <u>server</u>
- We need the context, to understand the meaning of server.
  - <u>Server</u>, can I have the **check**?
  - Looks like I just **crashed** the **<u>server</u>**
- We need to encode the positional surrounding



*Strictly speaking this is transposed

A naïve approach



One Block would have
$(700 \cdot 1000)^2 \approx 490\text{B}$ parameter

Too much for a single layer

Other architectures have been developed in the past.
RNNs and LSTMs

Transformer

.03762v5 [cs.CL] 6 Dec 2017

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including

https://arxiv.org/abs/1706.03762

- Introduced for translation
- Basis for Language models (GPT-2, GPT-3, Chat-GPT, Bert)

Google Scholar

Articles

1 result (**0.02** sec)

Lookup

Attention is all you need

[PDF] neurips.cc

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

The dominant sequence transduction models are based on complex recurrent orconvolutional neural networks in an encoder and decoder configuration. The best performing such models also connect the encoder and decoder through an attentionm echanisms. We propose a novel, simple network architecture based solely onan attention mechanism, dispe nd convolutions entirely. Experiments on two machine translatio asks show these m dels to be superiorin quality while being more …

Save Cite Cited by 62992 R ated articles All 46 versions

The proposed network

Every talk has to show this figure!

Encoder Decoder

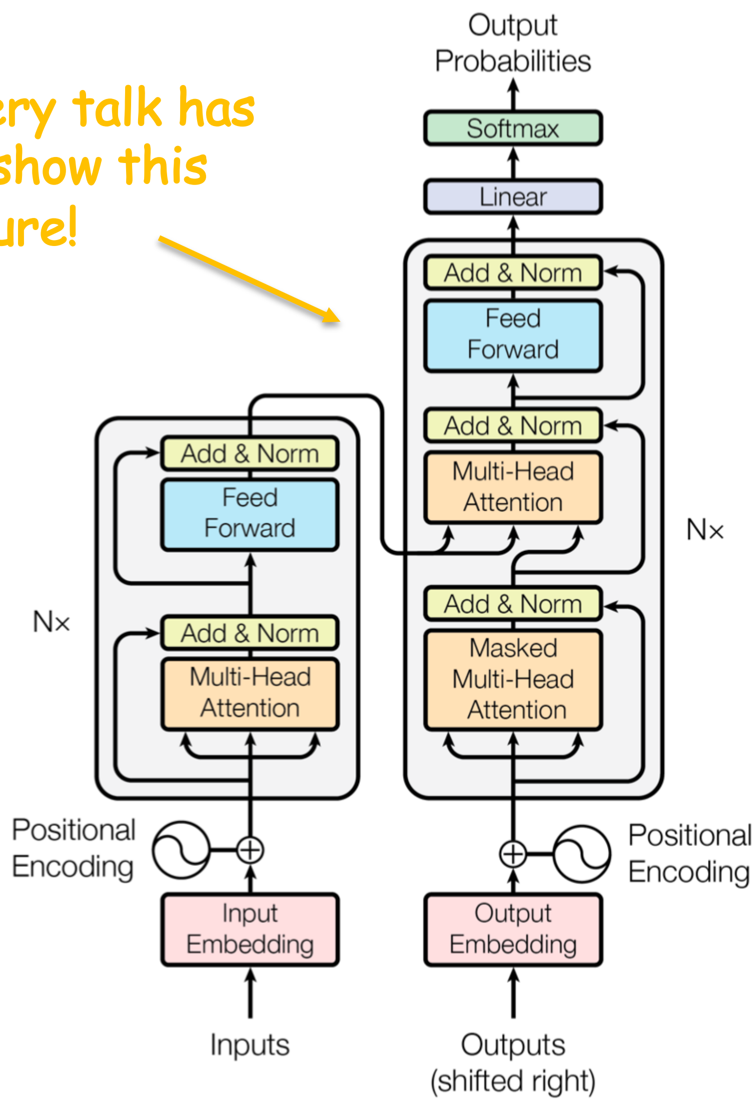


Figure 1: The Transformer - model architecture.

The attention blocks are the novel components: “Attention is all you need”.

Proposed for translation:

- Encoder summarizes the input in a context vector
- Decoder generates the output sequence, from context vector and previous output.

Many language models like GPT (Llama) just use the decoder part some like Bert the encoder.

GTP-Like Transformer

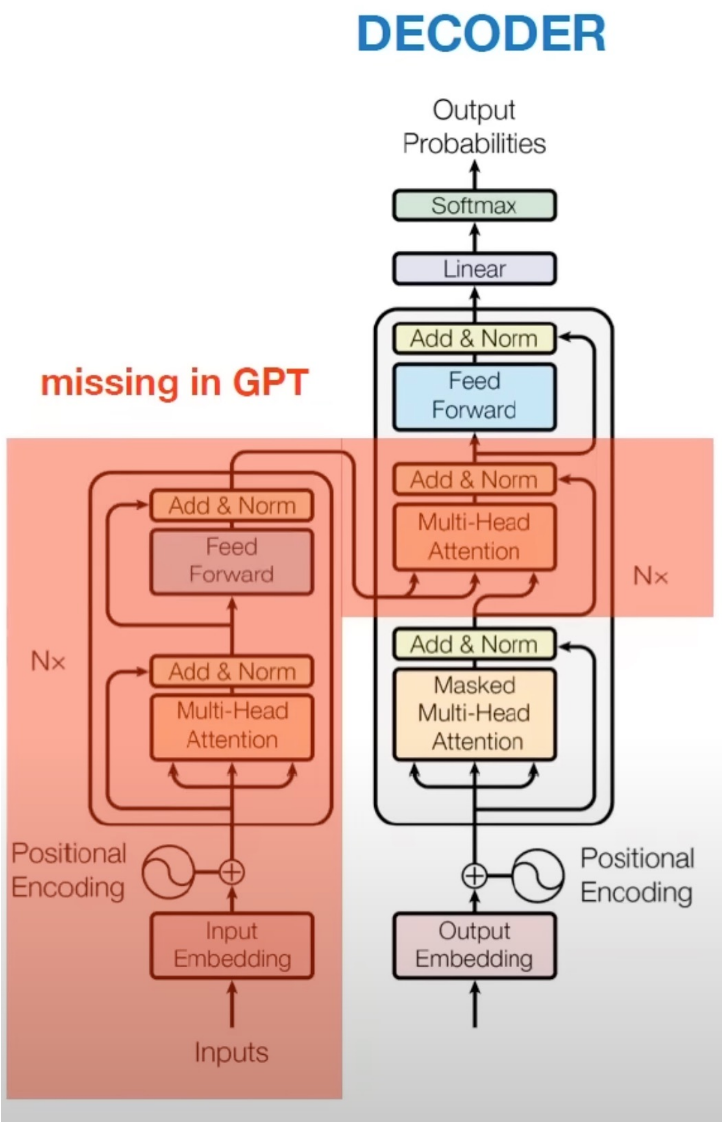
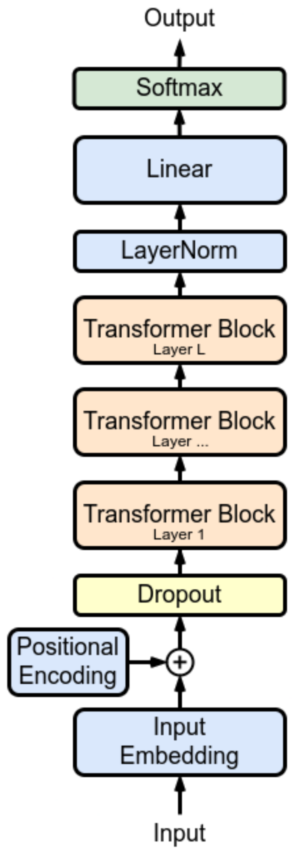


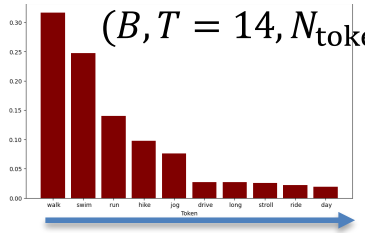
Figure 1: The Transformer - model architecture.

The decoder only implementation.
As in GTP



Follow the forward pass

$(B, T = 14, N_{\text{token}})$



N_{token}

(B, T, N_{token})

(B, T, N_{token})

(B, T, C)

(B, T, C)

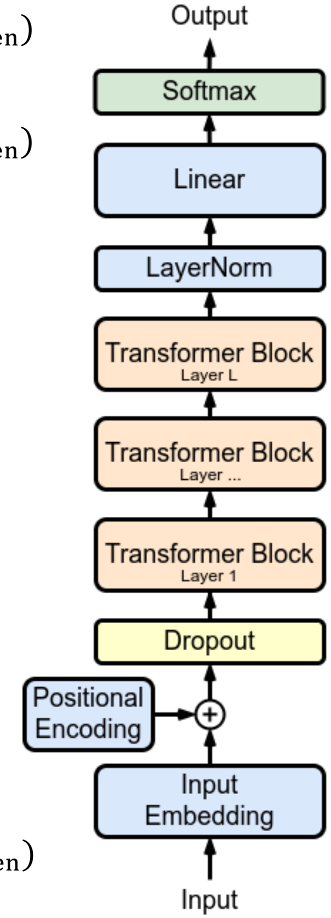
(B, T, C)

(B, T, C)

(B, T, C)

(B, T, C)

(B, T, N_{token})



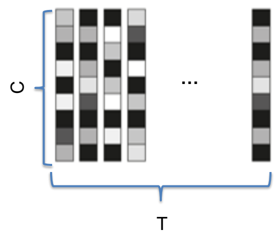
[791, 9282, 374, 2216, 6555, 3432, 13, 358, 2846, 7422, 922, 2133, 369, 264, 198]

The weather is really nice today. I'm thinking about going for a

$\text{softmax}(B, T, N_{token})$

$\text{LayerNorm}(B, T, C)$

(B, T, C) Tensor



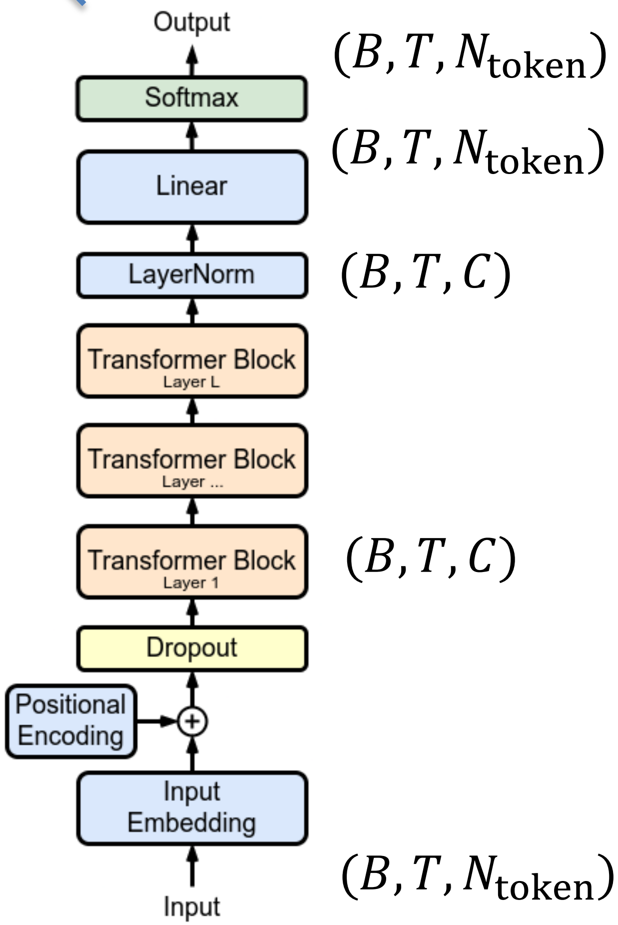
The dimension of the tensor (B, T, C) stays, it is just *transformed.*
Mostly along C dimension.

Training parallelism

Shift input one token to the left

~~The~~ weather is really nice today. I'm thinking about going for a walk

[~~791~~, 9282, 374, 2216, 6555, 3432, 13, 358, 2846, 7422, 922, 2133, 369, 264, 4321]



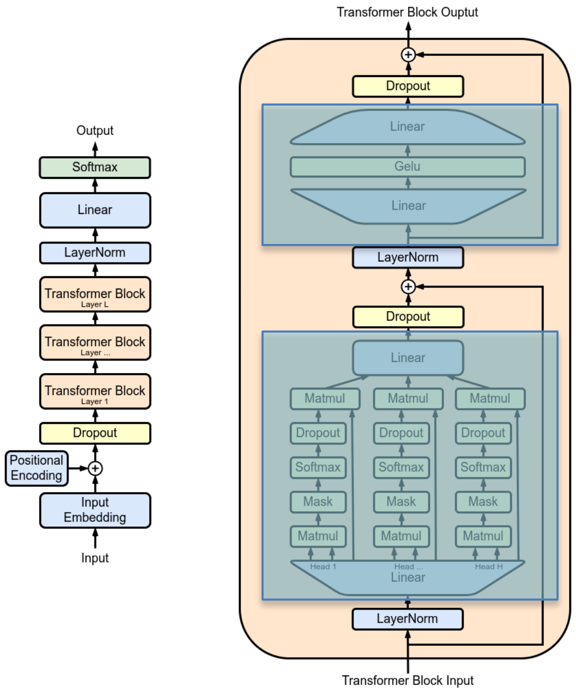
Attention ("Causality")

When predicting next word only information from pervious token must be used.

[791, 9282, 374, 2216, 6555, 3432, 13, 358, 2846, 7422, 922, 2133, 369, 264, 198]

The weather is really nice today. I'm thinking about going for a

Attention Block



(B, T, C)

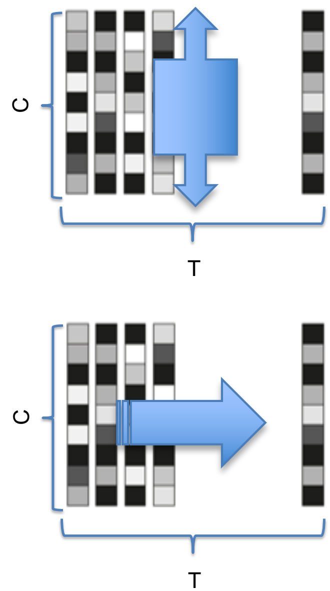
Position wise FcNN:
Adjusts representation

(B, T, C)

Attention Block:
Updates with information form (left) neighbors

(B, T, C)

Information Processing



Information is processed in two steps first

- Along T
- Along C

Total Memory $O(T) + O(C)$

Most operations along C-Direction

Image credit Wikipedia

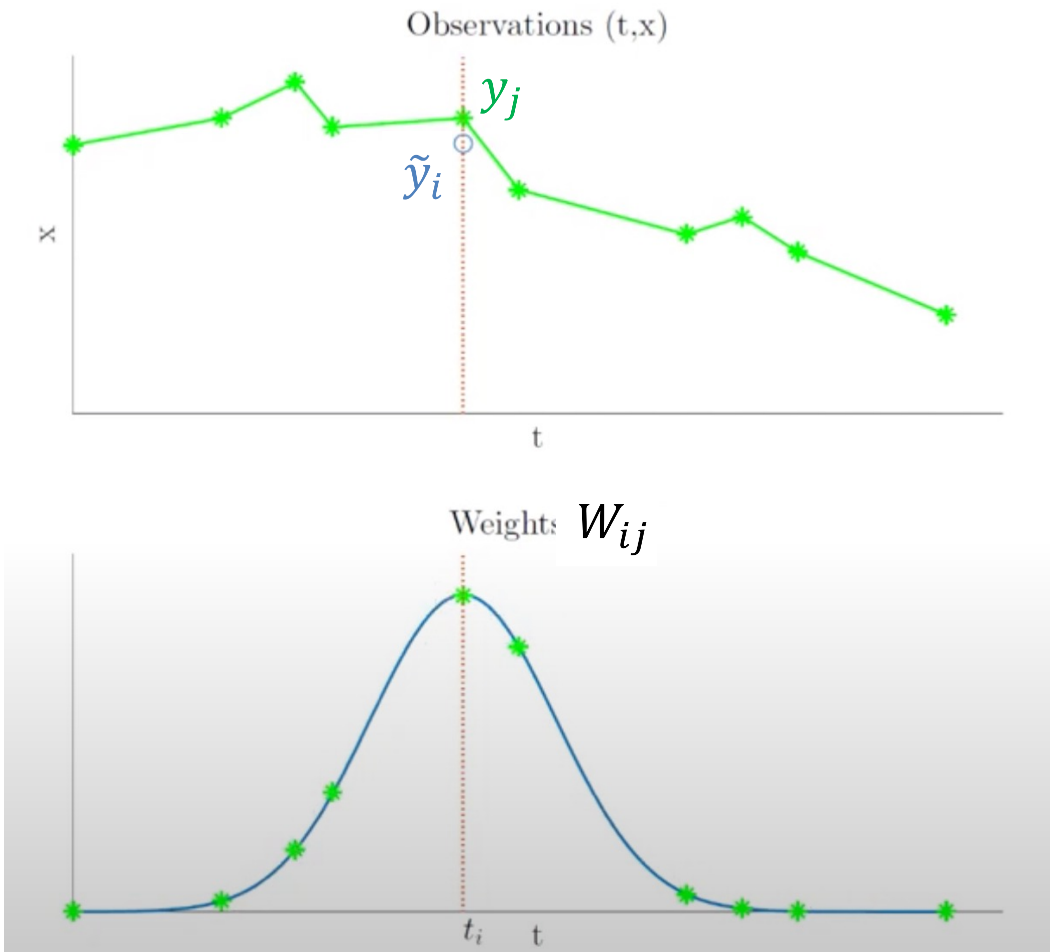
Intuition:: Weighted Average

Smoothing

$$\tilde{y}_i = \sum W_{ij} y_j$$

Weights sum up to 1

$\sum_j W_{ij} = 1$ (for all i)



Slide taken from Lennart Svensson

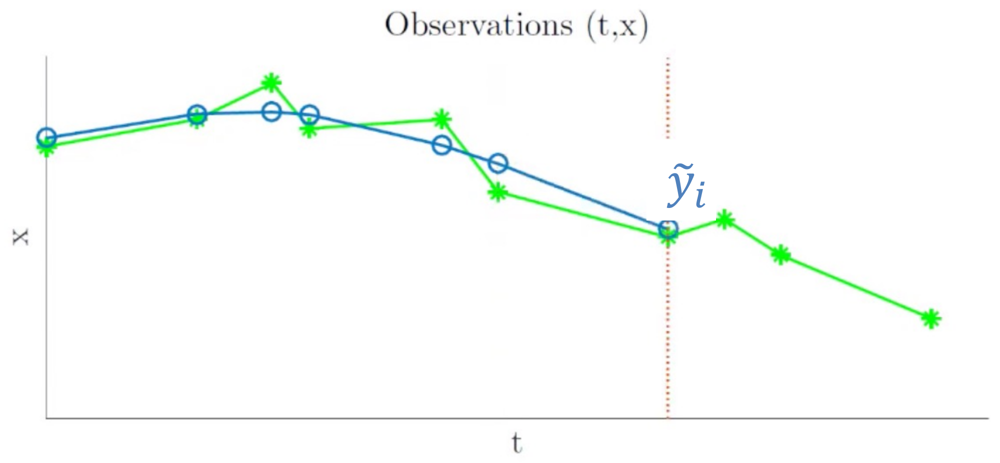
Intuition :: Weighted Average

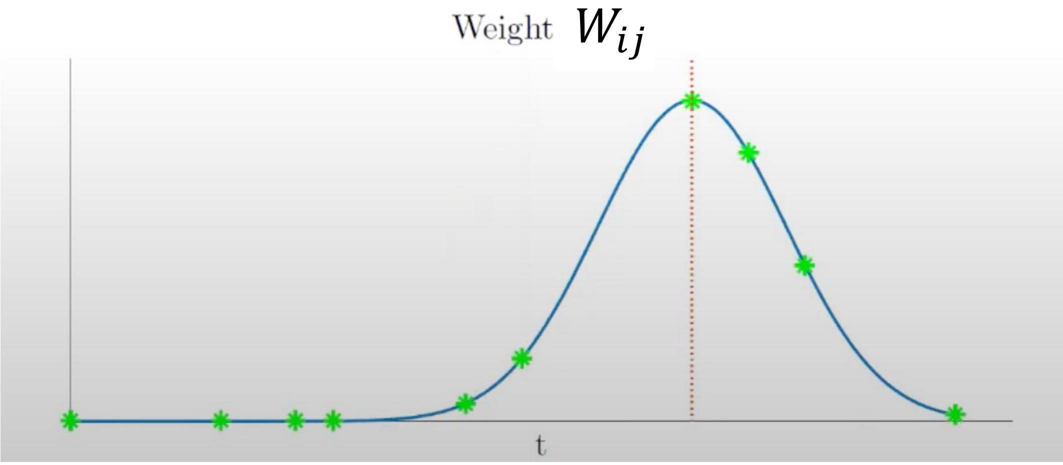
Smoothing

$$\tilde{y}_i = \sum W_{ji} y_i$$

Weights sum up to 1

$\sum_j W_{ji} = 1$ (for all i)





Including the Neighborhood gives better prediction / representation

Intuition:: Weighted Average of words

To what does "friend" refer to?

Emma hates games but she is a great friend

- Word vectors: x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9

- New representation of friend (which is less generic and describes friends in context of the other words j)
  - $\widetilde{x_9} = \sum_j W_{9j} x_j$
- Maybe better to use $v_j = W^V x_j$ instead of the value x_j to determine the weights (additional flexibility)
  - $\widetilde{x_9} = \sum_j W_{9j} v_j$

- Note that weights W_{9j} are specific to friend (no 9)
  - Which word(s) do you think should have height weights?
  - To which word should the network pay attention when analyzing friend?

An example: Emma hates games but she is a great friend

- Imaging the network at a stage where it has to figure out the relationships of objects
- This could be done in the following space
  - Dim1: Score that the word is a person name
  - Dim2: Score that the word is animal name
  - Dim3: Score that the word is a noun
  - Dim4: Score that the word is an adjective

- Examples of attention /weight of friend to Emma
  - Emma this is called a key
    - Might be a person or animal name, it's a noun and no adjective
    - $\mathrm{k}_1 = (1.2, 0.8, 1.0, 0.0)$
  - Friend, the word **itself** might be (no person / animal name)
    - $k_9 = (0,0,1.0,0)$
    - It **look at** (persons names, animals names, and adjectives).
    - $\mathrm{q}_9 = (1.0, 0.9, 0.5, 1)$

- The similarity between i and j is the dot-product between q_i and k_i
  - For q_9, k_1=$1.2 * 1 + 0.8 * 0.9 + 1.0 * .5 + 0 * 1$ =2.42

Queries
Words looking at

From which word

	Emma	hates	games	but	she	is	a	great	friend
Pers	1.1	0.0	0.0	0.0	1.1	0.0	0.2	0.5	1.0
Anim	0.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.9
Noun	0.9	1.0	1.0	0.0	0.0	0.0	1.0	1.0	0.5
Adj	0.0	0.0	0.0	0.0	-1.0	0.0	0.5	0.0	1.0

Keys
Words themself

To which word

	Emma	hates	games	but	she	is	a	great	friend
Pers	1.2	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
Anim	0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Noun	1.0	-0.5	1.0	-1.0	0.0	-0.5	-1.0	-1.0	1.0
Adj	0.0	0.0	-1.0	0.0	-1.0	0.0	0.0	1.2	0.0

$\widetilde{W}_{\text{from,to}}$

to

from

	Emma	hates	games	but	she	is	a	great	friend
Emma	2.78	-0.45	0.9	-0.9	1.1	-0.45	-0.9	-0.9	0.9
hates	1.00	-0.50	1.0	-1.0	0.0	-0.50	-1.0	-1.0	1.0
games	1.00	-0.50	1.0	-1.0	0.0	-0.50	-1.0	-1.0	1.0
but	0.00	0.00	0.0	0.0	0.0	0.00	0.0	0.0	0.0
she	1.32	0.00	1.0	0.0	2.1	0.00	0.0	-1.2	0.0
is	0.00	0.00	0.0	0.0	0.0	0.00	0.0	0.0	0.0
a	1.24	-0.50	0.5	-1.0	-0.3	-0.50	-1.0	-0.4	1.0
great	1.60	-0.50	1.0	-1.0	0.5	-0.50	-1.0	-1.0	1.0
friend	2.42	-0.25	-0.5	-0.5	0.0	-0.25	-0.5	0.7	0.5

Emma	friend
1.2	1.0
0.8	0.9
1.0	0.5
0.0	1.0

$$1.2 * 1 + 0.8 * 0.9 + 1.0 * .5 + 0 * 1 = 2.42$$

Implementation Detail

f From which word

Queries
Words looking at

i

	Emma	hates	games	but	she	is	a	great	friend
Pers	1.1	0.0	0.0	0.0	1.1	0.0	0.2	0.5	1.0
Anim	0.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.9
Noun	0.9	1.0	1.0	0.0	0.0	0.0	1.0	1.0	0.5
Adj	0.0	0.0	0.0	0.0	-1.0	0.0	0.5	0.0	1.0

t to which word

Keys
Words themself

i

	Emma	hates	games	but	she	is	a	great	friend
Pers	1.2	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
Anim	0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Noun	1.0	-0.5	1.0	-1.0	0.0	-0.5	-1.0	-1.0	1.0
Adj	0.0	0.0	-1.0	0.0	-1.0	0.0	0.0	1.2	0.0

$$\widetilde{W}_{f,t} = \sum_{i=1}^{9} Q_{i,f} \cdot K_{i,t}$$

```
np.einsum('if,it->ft', Q,K)
```

The Einstein summation goes over repeated indices. Easy to extend when having higher dimensions tensors such as (B,T,C)

```
np.einsum('bif,bit->bft', Q,K)
```

Normalization

- The weight matrix $W_{f,t}$ need to be normalized
- $W_{f,t} = \frac{e^{\widetilde{W}_{f,t}}}{\sum_t e^{\widetilde{W}_{f,t}}}$

$$W_{f,t} = \text{softmax}(\widetilde{W}_{f,t})$$

	Emma	hates	games	but	she	is	a	great	friend
Emma	0.61	0.02	0.09	0.02	0.11	0.02	0.02	0.02	0.09
hates	0.24	0.05	0.24	0.03	0.09	0.05	0.03	0.03	0.24
games	0.24	0.05	0.24	0.03	0.09	0.05	0.03	0.03	0.24
but	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
she	0.19	0.05	0.14	0.05	0.41	0.05	0.05	0.02	0.05
is	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
a	0.31	0.05	0.15	0.03	0.07	0.05	0.03	0.06	0.24
great	0.35	0.04	0.19	0.03	0.11	0.04	0.03	0.03	0.19
friend	0.58	0.04	0.03	0.03	0.05	0.04	0.03	0.10	0.09

2 Issues

1. Systematic dependency of “Peakyness” on dimension m of the query space

$$W_{f,t} = \text{softmax}\left(\frac{\widetilde{W}_{f,t}}{\sqrt{m}}\right)$$

2. Causality. E.g. Emma is not allowed to depend on hates
   (we want parallel training)

Complete Code for weight matrix

```
m = Q.shape[0]
wtilde = np.einsum('if,it->ft', Q,K)
T = wtilde.shape[0]
for i in range(0,T):
    for j in range(i+1, T):
        wtilde[i,j] = -np.inf
w = softmax(wtilde/np.sqrt(m))
pd.DataFrame(np.round(w, 2), columns=df_queries.index, index=df_keys.index)
```

✓ 0.0s

to

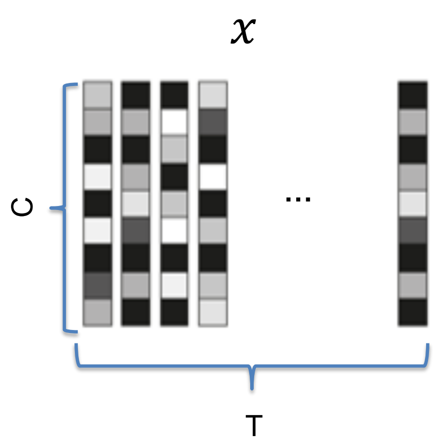
from

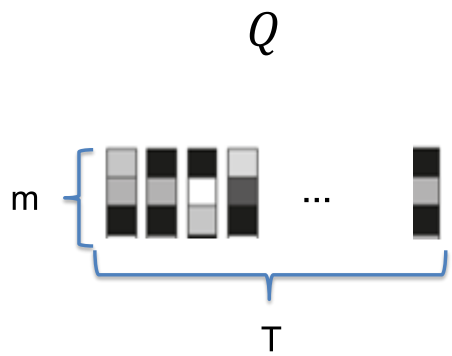
	Emma	hates	games	but	she	is	a	great	friend
Emma	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
hates	0.68	0.32	0.00	0.00	0.00	0.00	0.00	0.00	0.00
games	0.40	0.19	0.40	0.00	0.00	0.00	0.00	0.00	0.00
but	0.25	0.25	0.25	0.25	0.00	0.00	0.00	0.00	0.00
she	0.23	0.12	0.20	0.12	0.34	0.00	0.00	0.00	0.00
is	0.17	0.17	0.17	0.17	0.17	0.17	0.00	0.00	0.00
a	0.27	0.11	0.19	0.09	0.13	0.11	0.09	0.00	0.00
great	0.26	0.09	0.19	0.07	0.15	0.09	0.07	0.07	0.00
friend	0.30	0.08	0.07	0.07	0.09	0.08	0.07	0.13	0.12

The query, the keys, and the values

- Going into the query space, W^Q and W^K **are learned!**
  - We summarize our search **query** for $x_i = \text{"}Friend\text{"}$ with $q_i = W^Q x_i$
  - We summarize our search **key** with $k_j = W^K x_j$

$$Q = W^Q x$$

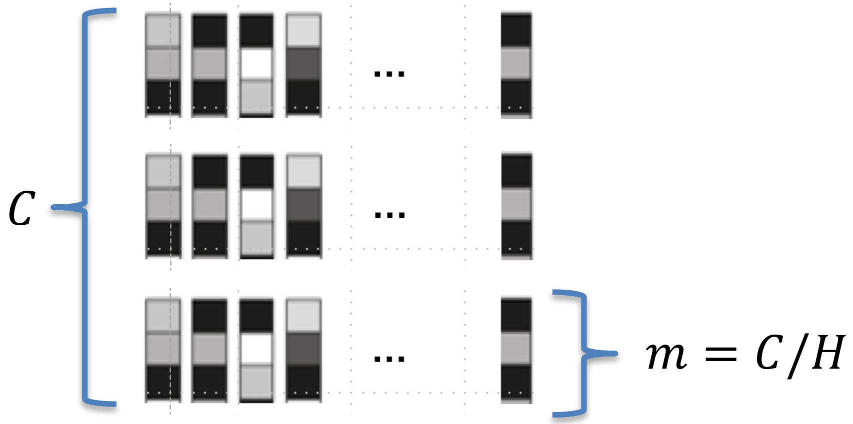


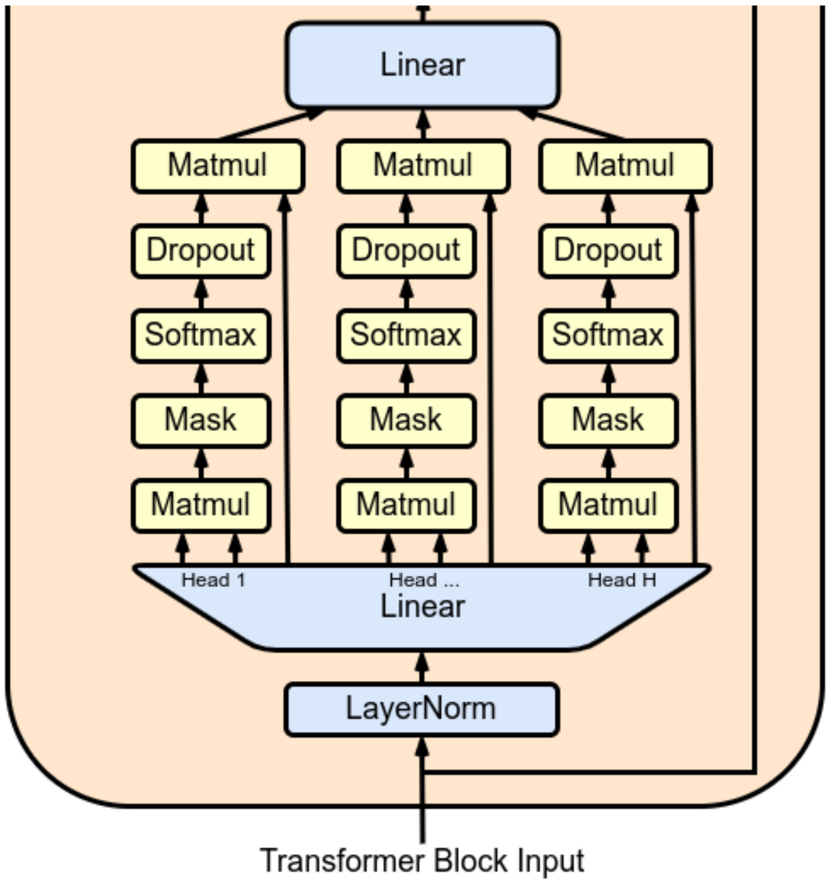


- For averaging (getting the new value) **also learned**
  - $\widetilde{x_9} = \sum_j W_{9j} v_j$
  - $v_j = W^V x_j$ we take transformation v_j instead of x_j

Multihead Attention

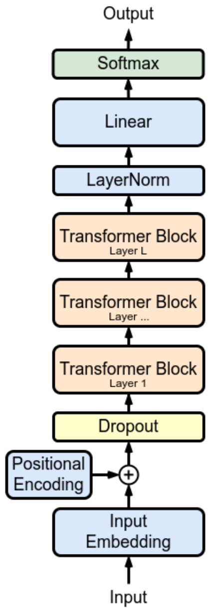
- Choose query space dimension $m = C/H$ where H are number of heads. Same with values.

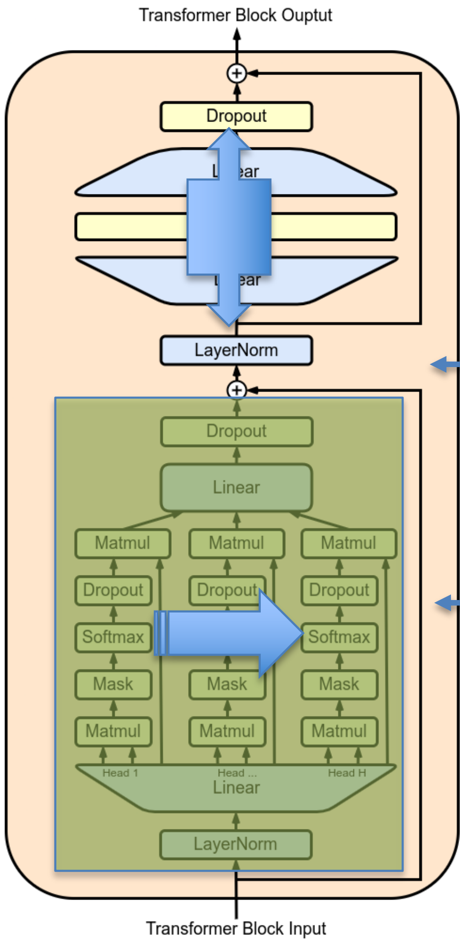




The complete Transformer Block

Remaining





FcNN

Normalization (Z-Trafo)

Skip Connections
Resnet like

Position is not

Who does "friend" refer to?

Emma hates games but she is a great friend

- Word vectors: x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9

There is no distance, coded yet.

The quantity x_1 is just the first token x_2 second.

GTP simply embeds uses the positions as token numbers [0,1,2,…,T] and embeds them.

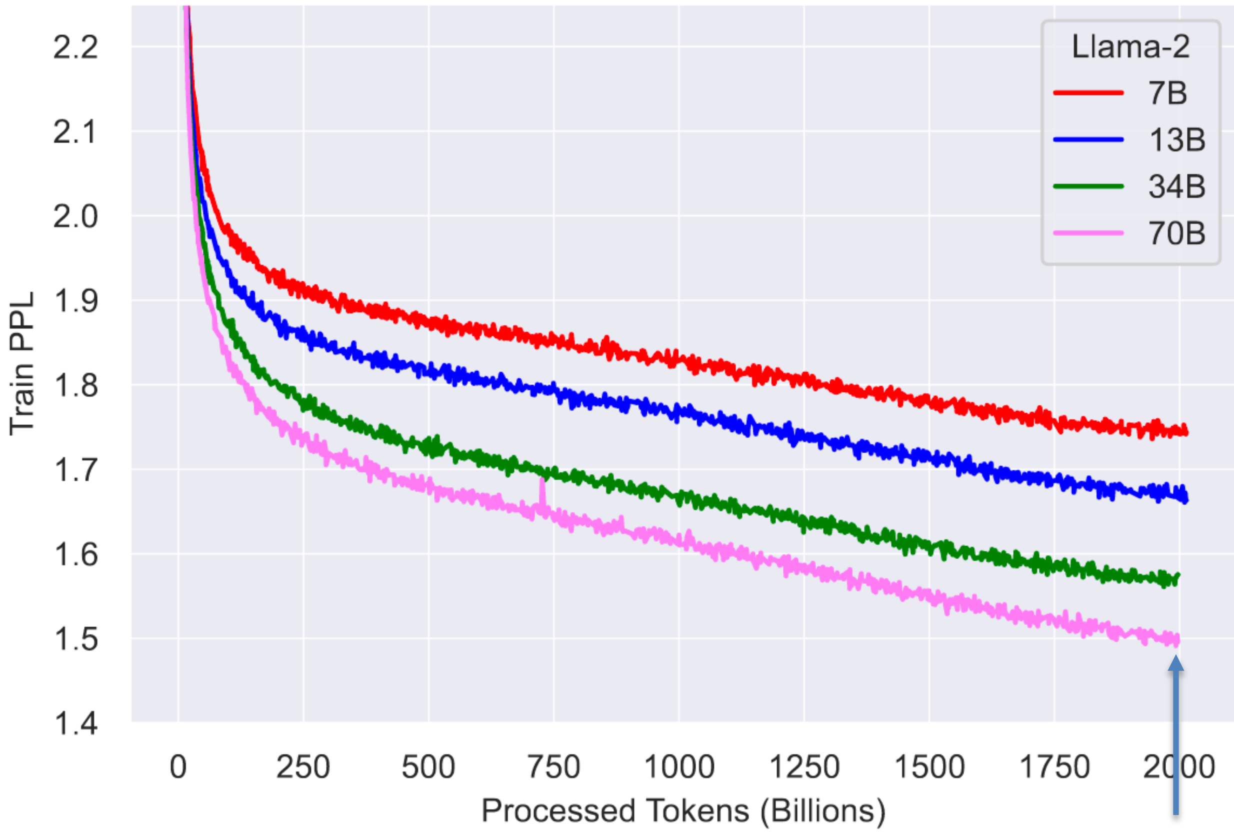
Demo (ATTO-GTP)

https://github.com/ioskn/mldl_htwg/blob/master/transformers/atto_GTP.py

Training LLama2

Number of parameters 7B-70B
Number of Tokens $N_{\text{token}} = 32k$

Context Length $T = 2k - 4k$
Number of of Layers 32-80
Context Size C=4096-8192



For 70B
1.7 Million GPU Hours
291.42 tCO2eq
(70 Household for a year)

http://arxiv.org/abs/2307.09288

Alignment Finetuning

Plain vanilla LLM

prompt **What is the capital of France?**

What is France's largest city?

What is France's population?

What is the currency of France?

Finetuned versions (Chat-GPT and Chat LLama)

prompt **What is the capital of France?**

Paris

Further aspects of fine tuning (Helpful, Honest, Harmless)

No magic in transformers.

…Then a miracle happens

Hard to make predictions, these days

Yann LeCun
Jan 2022

"I take an object, I put it on the table, and I push the table […] GPT-5000 is never gona learn what happens, this information is not present in any text.

ChatGPT-3.5 **(Dec 2022)**

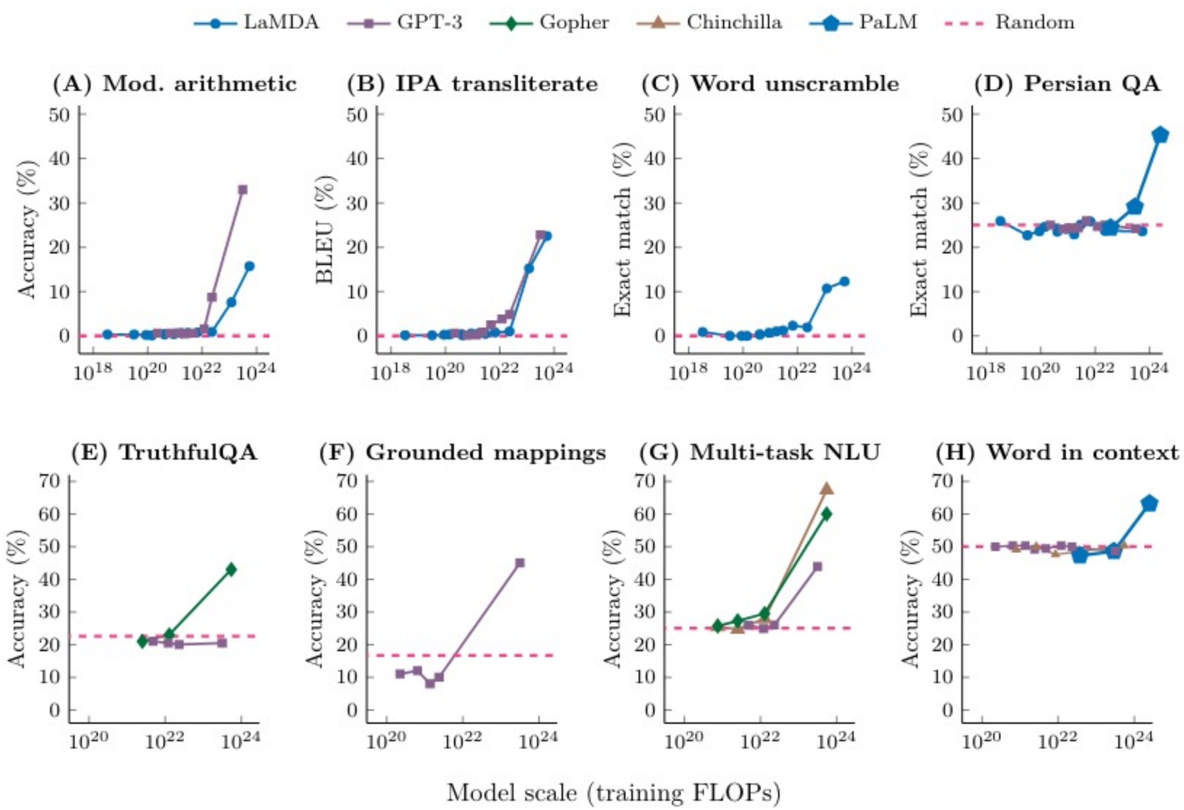
I put an object on the table and I push the table. What happens to the object? Give your best guess in max 3 sentences.

When you push the table, the object on it is likely to move along with the table, assuming the push isn't too violent or the object isn't overly fragile or unstable. The object's movement will be subject to the principles of inertia and friction; it will try to remain in its current state of motion until an external force acts upon it. However, if the push is too forceful, the object might slide or topple over.

https://twitter.com/liron/status/1659618568282185728

More is different

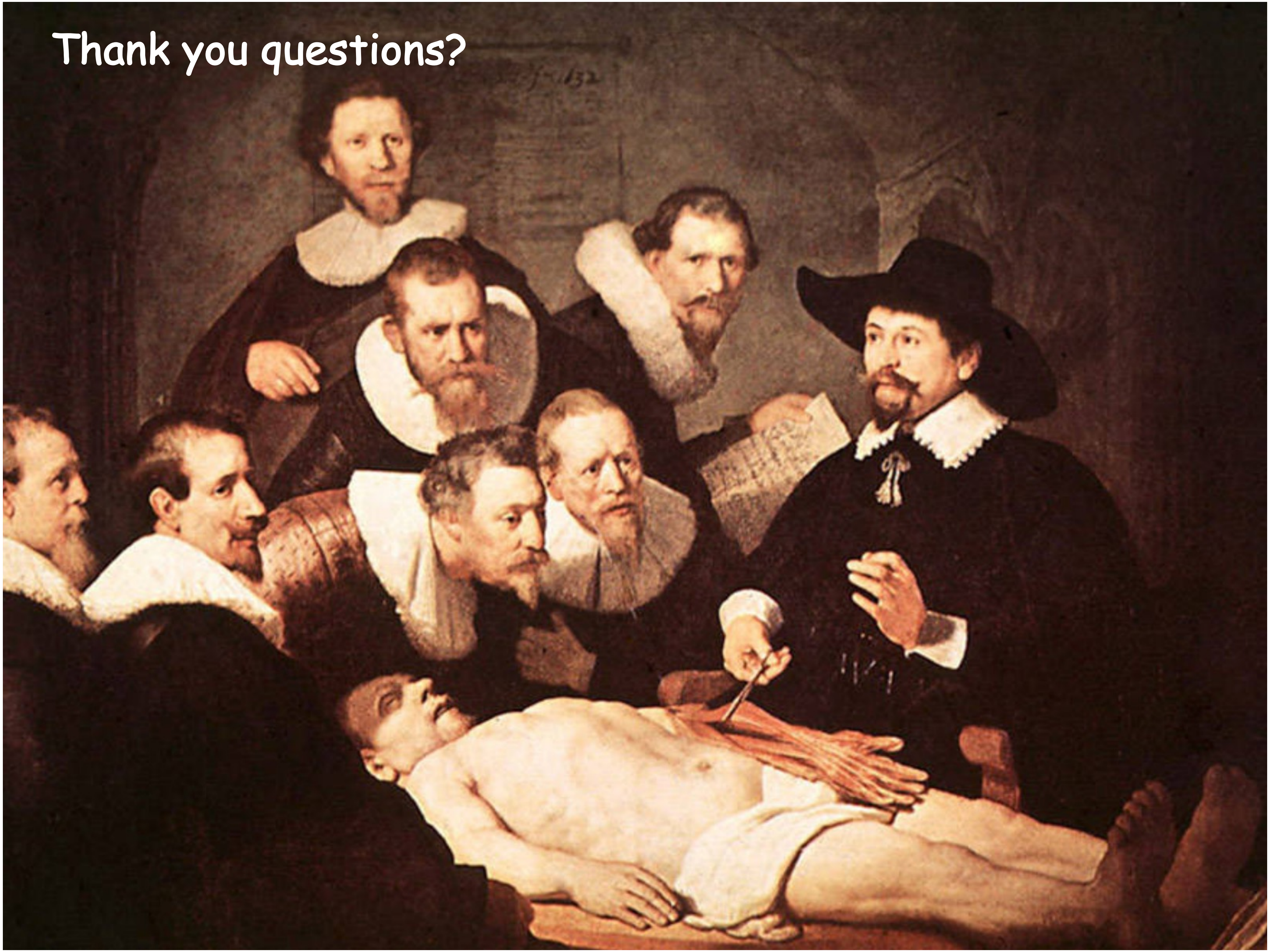
- LLM where trained to predict the next token in a sequence of tokens but…



Stochastical parrots 🦜?

- Just as dissecting a brain wouldn't necessarily reveal the nature of human consciousness, examining the mechanics of an LLM doesn't fully explain its emergent capabilities.

https://arxiv.org/pdf/2206.07682.pdf


Thank you questions?