Modern NLP through practical problems

Andrej Miščič, Luka Vranješ

O podjetju Valira AI

Valira AI je agencija, ki ponuja storitve na področju podatkovnih ved in umetne inteligence.

Projektno delo

Svetovanje

Izobraževanje

info@valira.ai

Kdo smo

Luka
Vranješ

Andrej
Miščič

Greta
Gašparac

Podatkovni znanstveniki, po izobrazbi magistri inženirji podatkovnih in računalniških ved.

Vizija: V področje vnesti dobre prakse in pomagati podjetjem odkriti dodano vrednost v njihovih podatkih.

Agenda

00 Theoretical introduction:

- Transformer architecture
- brief overview of model pretraining
- BERT, GPT -> ChatGPT

01 Introduction to HuggingFace ecosystem

02 BREAK

03 Practical part:

- Sentiment analysis with BERT
- Named Entity Recognition with BERT
- Text generation with GPT-2
- Abstractive summarization with BART
- Code completion and summarization with pretrained transformers

Motivation

time series forecasting

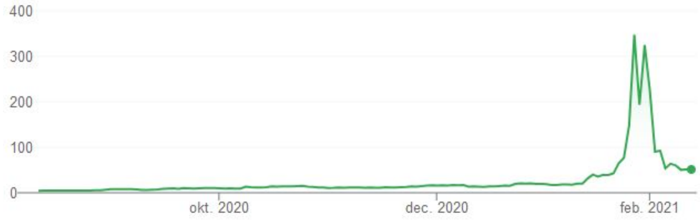


image caption generation

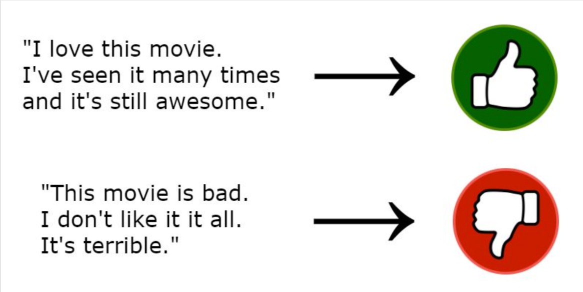
"man in black shirt is playing guitar."

question answering

What’s the capital of Slovenia?

Ljubljana

text classification



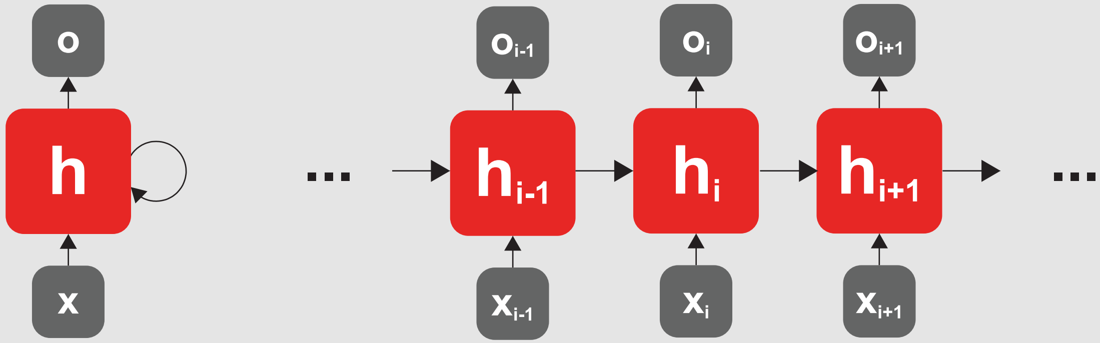
named-entity recognition

We are Andrej and Luka. We work for Valira AI.

RNNs

- extension of NNs for sequential data;

- information persists in hidden state h_i.



Problems with RNNs

- difficult to train to capture long-term dependencies [1]:

*I watched Spongebob on **RTL** when I was young, that's why I'm fluent in*

French.
German.
English.

Reasons:

- vanishing/exploding gradients:
  - for gradient we multiply the same term multiple times
  - use of saturated activation functions (*sigmoid, tanh*)

- the hidden state overwritten in every step:

$$h_i = \begin{cases} 0, & \text{if } i = 0 \\ \sigma(W_{in}x_i + W_{hid}h_{i-1}), & \text{otherwise} \end{cases}$$

[1] Bengio et al.: Learning long-term dependencies with gradient descent is difficult, 1994

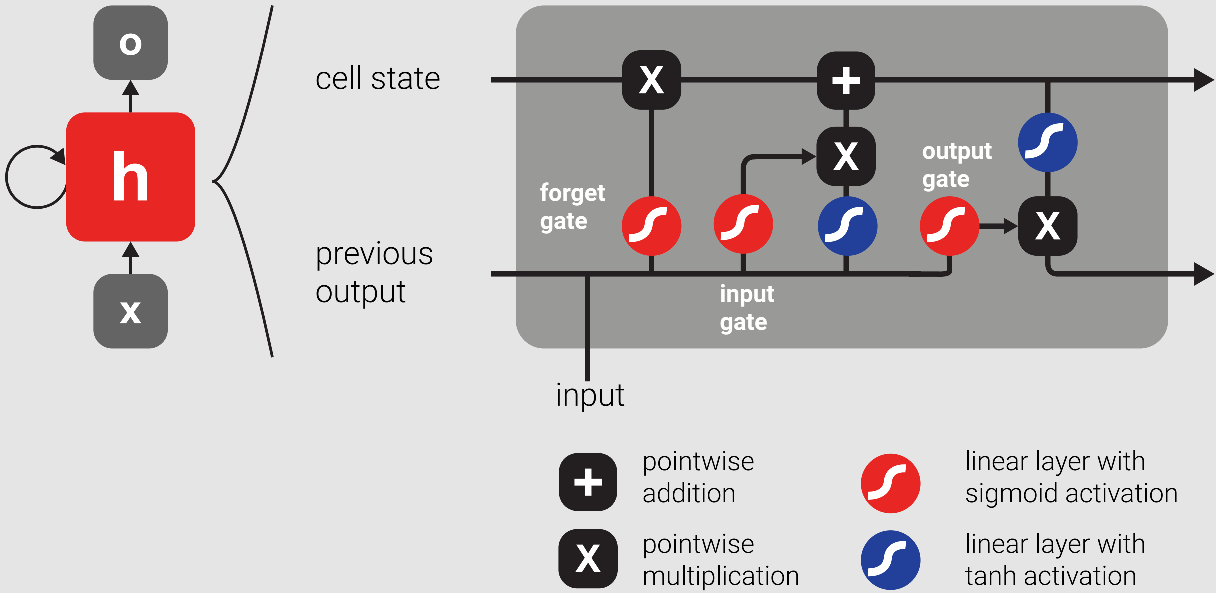
Solution: Gating

- LSTM [1], GRU [2]

- gates modulate the flow of information;

- cell state is not overwritten, old information is forgotten, new added.

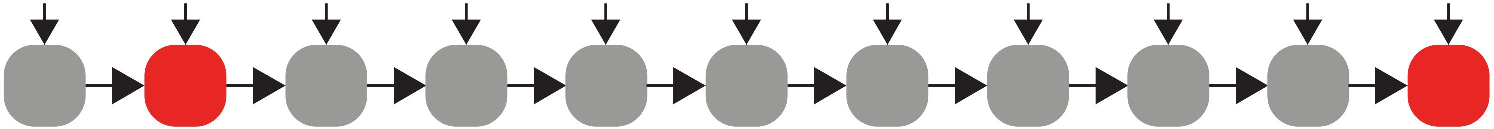
LSTM Unit



[1] Hochreiter, Schmidhuber: Long Short-Term Memory, 1997
[2] Cho et al.: On the properties of neural machine translation: Encoder-decoder approaches, 2014

Motivation for Transformers

- RNNs are inherently sequential which prevents parallelization;
- the problem of long-term dependencies:
    - gating somewhat mitigates this problem, however, the path length between any two dependant words is still O(n)



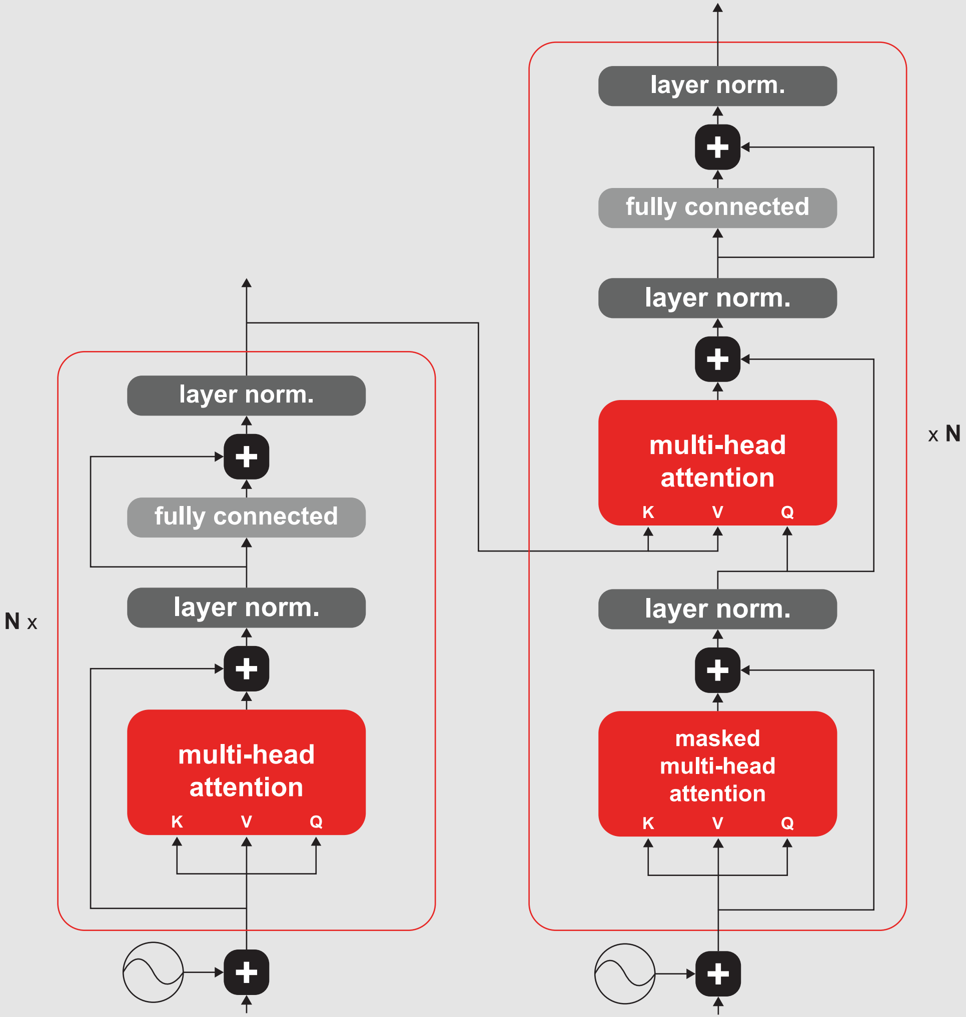
- Can we get rid of recurrence? What to replace it with?

Transformer[1]

- introduced for Neural Machine Translation;

- encoder-decoder architecture;

- uses **self-attention** in place of recurrence.



[1] Vaswani et al.: Attention Is All You Need, 2017

Self-attention

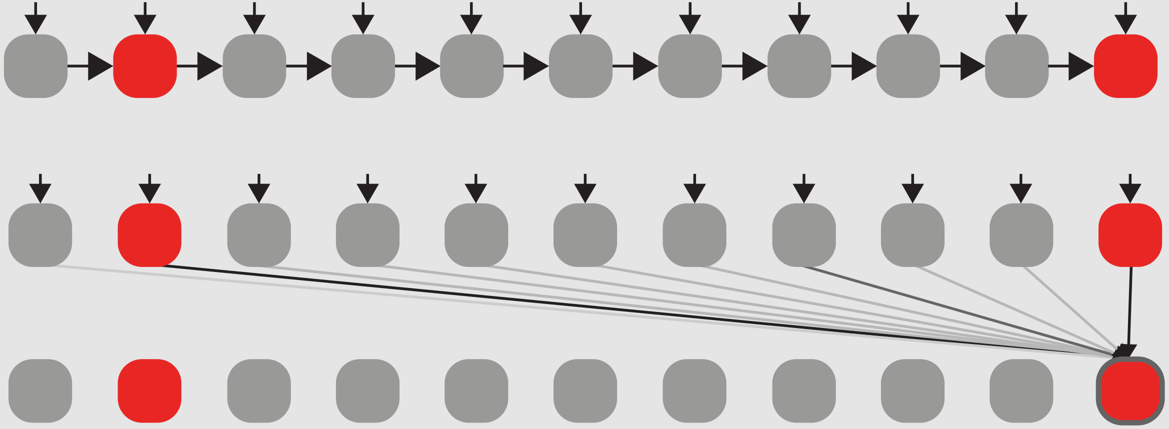
The animal didn't cross the street because it was too ?

The animal didn't cross the street because it was too wide.

The animal didn't cross the street because it was too tired.

- RNN: path length between two words is O(n); →

- in self-attention the path length is O(1). →



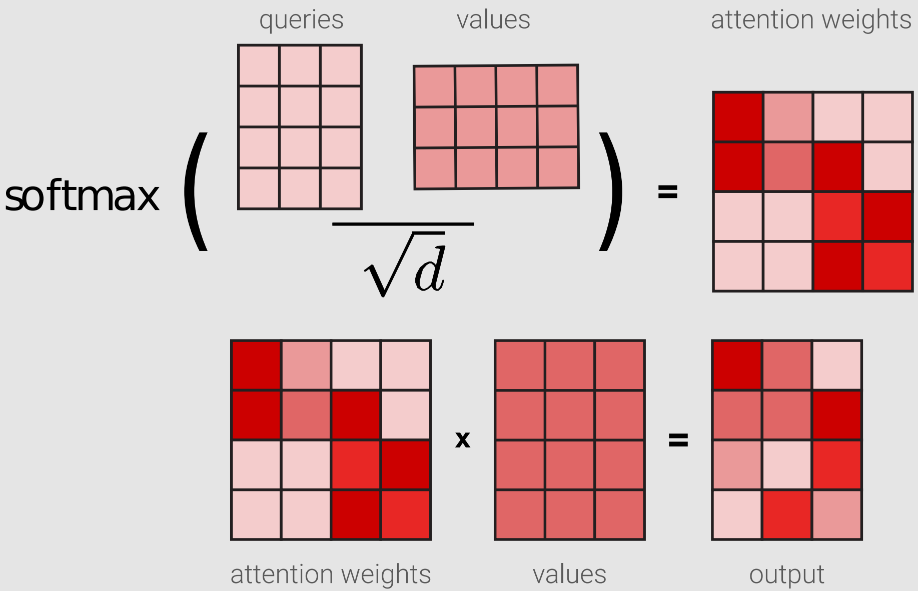
Example credit: Hang Dong.

Self-attention

summary of values **V** based on similarity between a particular query $\mathbf{Q_i}$ and keys **K**

Q, **K**, **V** - linear projections of token embeddings

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

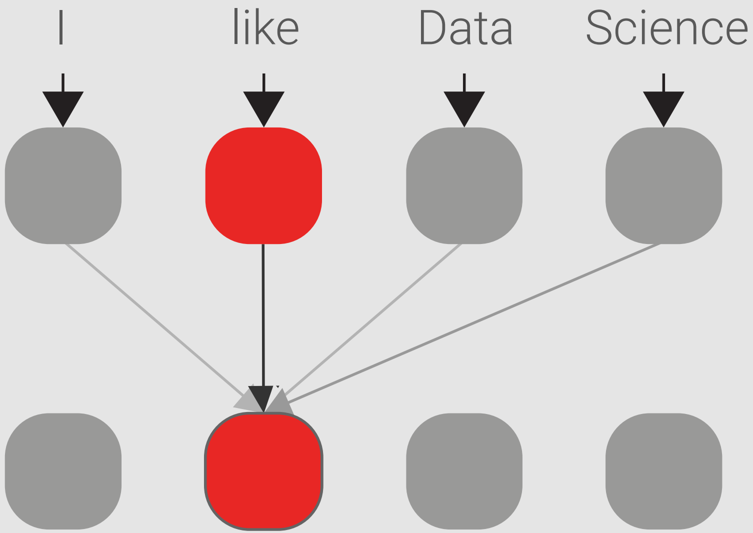


Problem

- single self-attention can be a bottleneck;

- cannot capture multiple interactions between words;

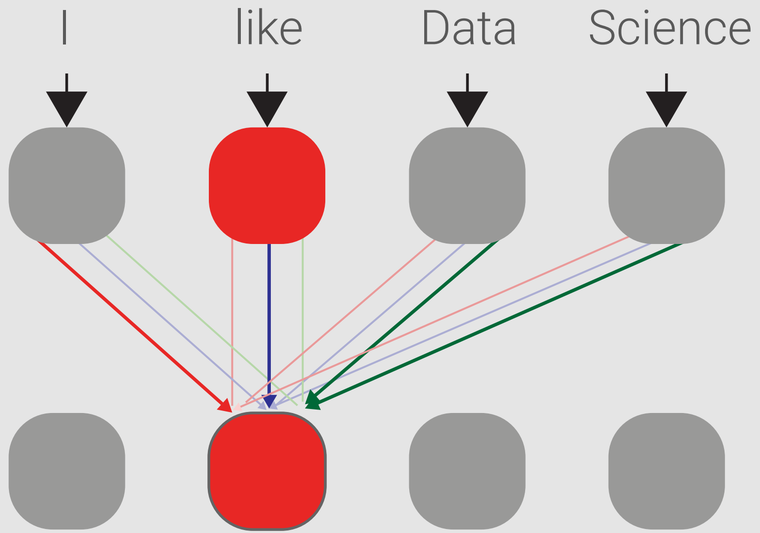
- in our example we want to know for word *like:*
    - who likes?
    - does what? (attend to itself)
    - likes what?



Solution

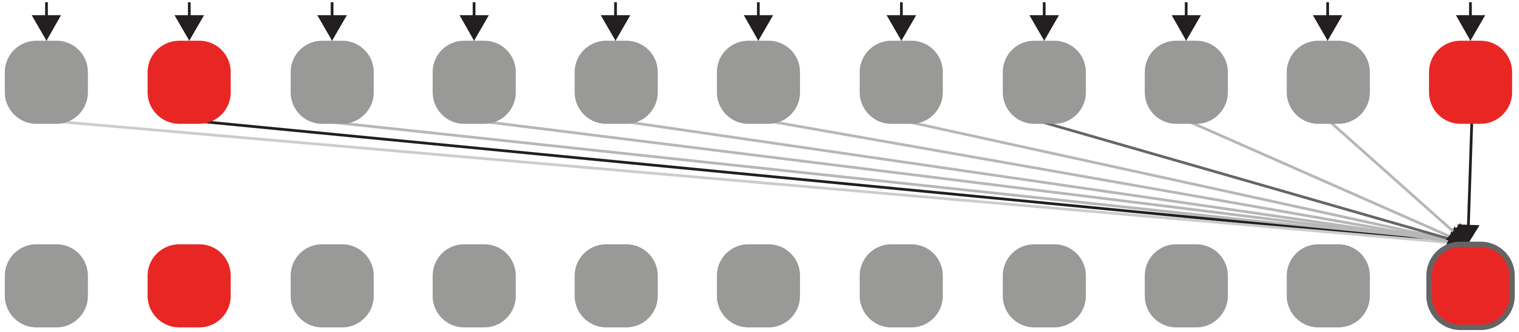
- multiple parallel copies of attention - Multi-Head attention;

- different attention heads can now pick up different interactions.



Problem

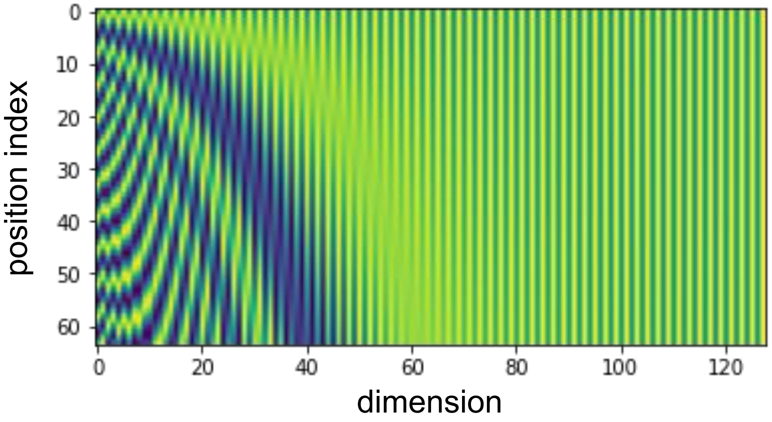
- by getting rid of recurrence we lose positional information which is important as our data is sequential;
- self-attention is permutation invariant, i.e. no matter the order of the inputs, the output will be the same.



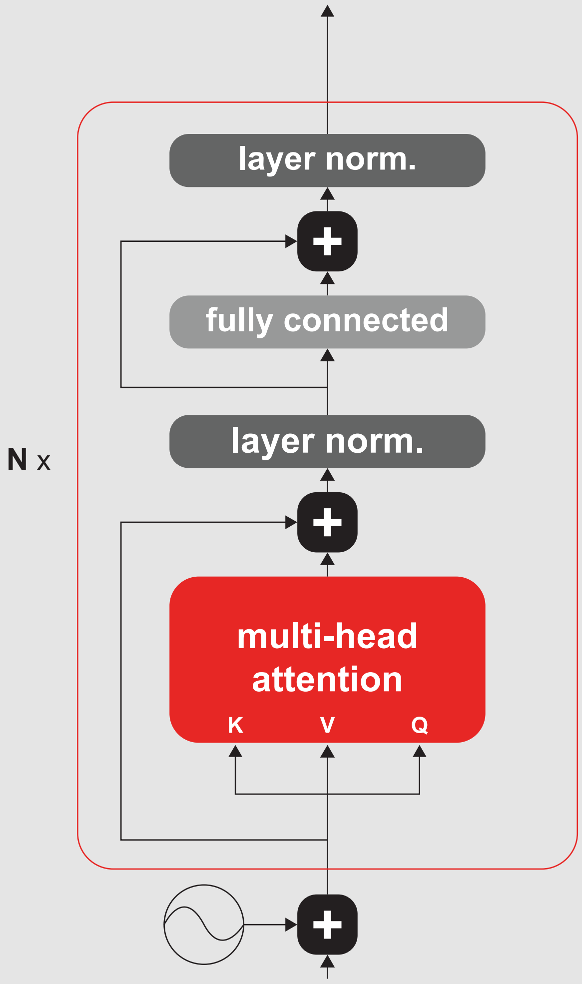
Solution: Positional encodings

- positional encodings have the same dimension as input embeddings and are added to them before the first self-attention layer;
- they can be either:
  - LEARNED: use an embedding layer to learn a pos. embedding for each position in the sequence;
  - FIXED: set before training, used in original paper.

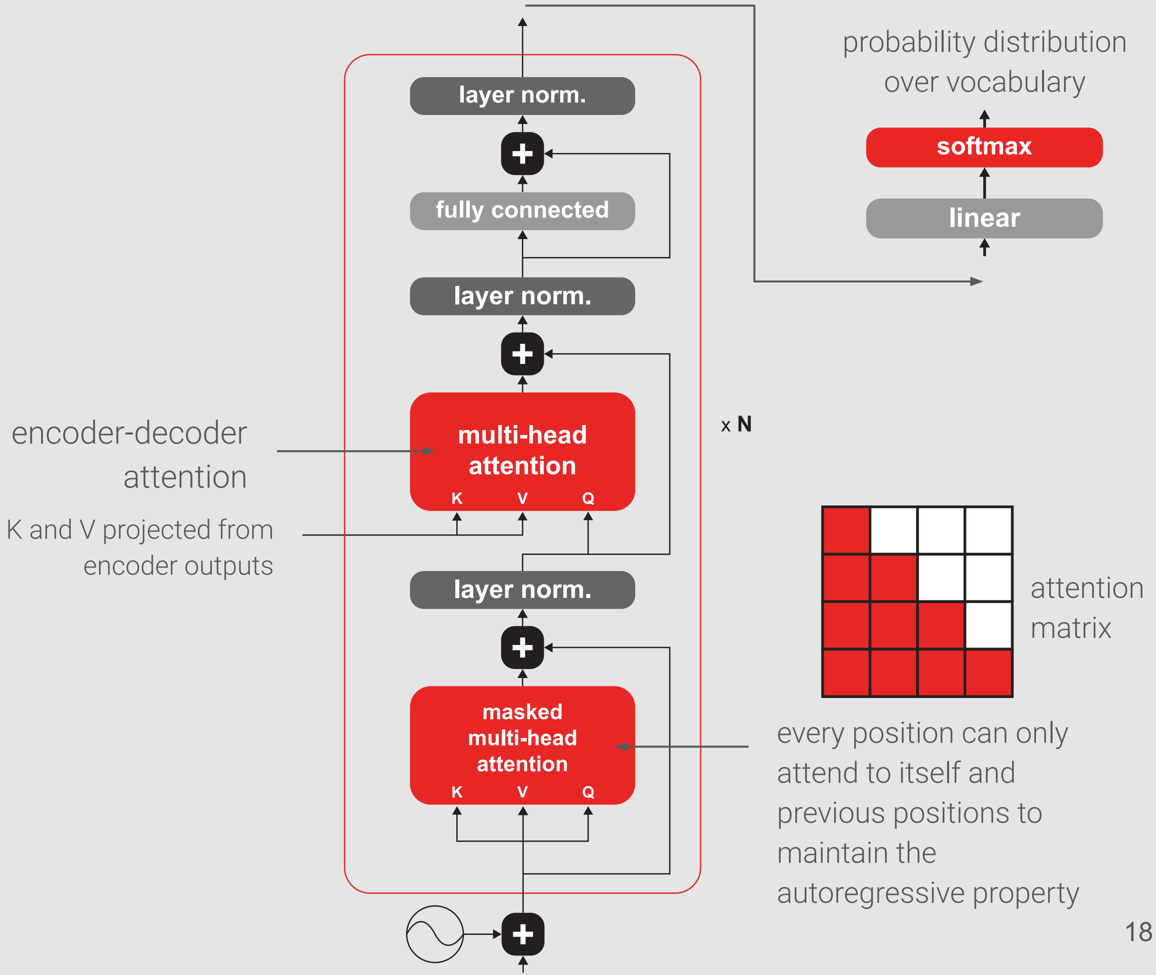
$$PE_{ij} = \begin{cases} \sin(i/10000^{\frac{j}{d_m}}) & \text{if j is even} \\ \cos(i/10000^{\frac{j-1}{d_m}}) & \text{if j is odd} \end{cases}$$



Transformer encoder



Transformer decoder



Pretraining

- deep learning requires lots of annotated data, which can be scarce;
- on the other hand, we have abundant unlabeled text data;

↓

- leverage this unlabeled data to pre-train word representations/models in a self-supervised manner and use them on downstream tasks.

Pretraining

- neural word embeddings, e.g. Word2Vec **[1]**, Glove **[2]**, (context-free);

  *We went to see a **play** at the local theater.*

  *Children went out to **play** in the park.*

- transfer learning (pretrain-then-finetune)
  - can we develop models that adapt to many NLP tasks with little to no modification?
  - BERT **[3]**, T5 **[4]**, BART **[5]**

[1] Mikolov et al.: Efficient Estimation of Word Representations in Vector Space, 2013.
[2] Pennington et al.: GloVe: Global Vectors for Word Representation, 2014.
[3] Devlin et al.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.
[4] Raffel et al.: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, 2019.
[5] Lewis et al.: BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension, 2019.

NLP problems examples

Natural Language Understanding

- text classification
- named-entity recognition
- reading comprehension
- etc.

encoder-only arch.

e.g. BERT, RoBERTa

Natural Language Generation

- machine translation
- abstractive summarization
- closed-book QA
- etc.

encoder-decoder arch.

e.g. T5, BART

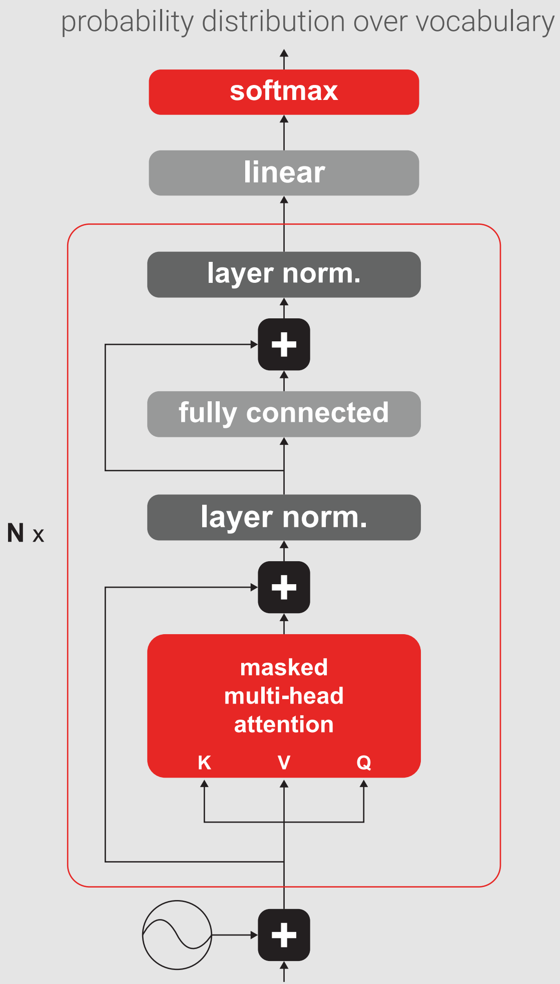
GPT [1]

- Generative Pre-trained Transformer;

- using only the decoder part of Transformer;

- pre-trained for language modelling, i.e. predicting next word given the context.

[1] Radford et al.: Improving Language Understanding by Generative Pre-Training, 2018.



Fine-tuning

$$L(X) = L_{LM}(X) + \lambda L_D(X)$$

language modelling loss

downstream task loss

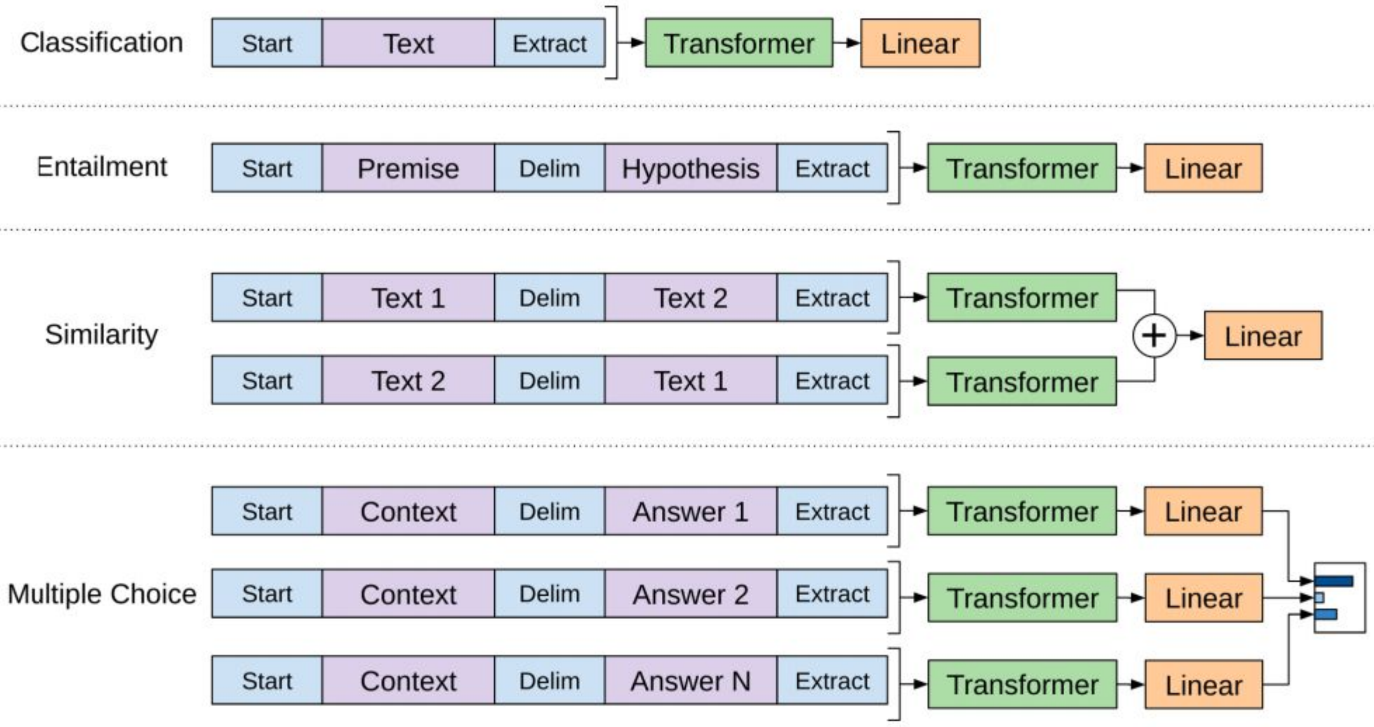


Figure source: [Radford et al.: Improving Language Understanding by Generative Pre-Training, 2018.]

GPT shortcomings

- language modelling is an unidirectional task, models predict the next word given the left context:

*‘What are those?’ he said while looking at my **[?]***

- better language understanding requires incorporating bidirectionality:

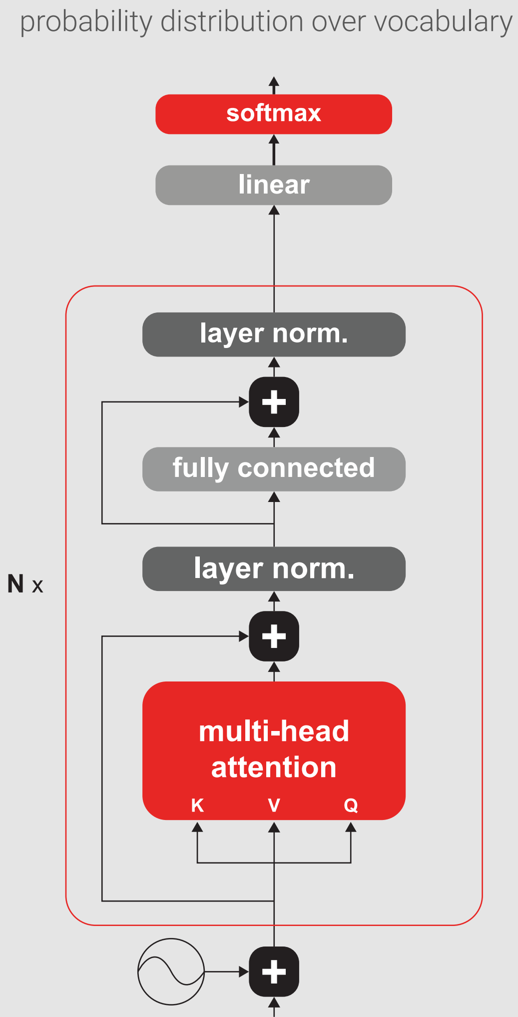
*‘What are **those**?’ he said while looking at my **crocs**.*

context

BERT [1]

- Bidirectional Encoder Representations from Transformers;

- using only the encoder part of Transformer.



[1] Devlin et al.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

Masked language modelling

- 15% of input words are masked, the model learns to predict the missing words

What *looking*

‘[MASK] are those?’ he said while [MASK] at my crocs.

Too much masking:
Model is not provided with enough context.

Too little masking:
Learning becomes very slow.

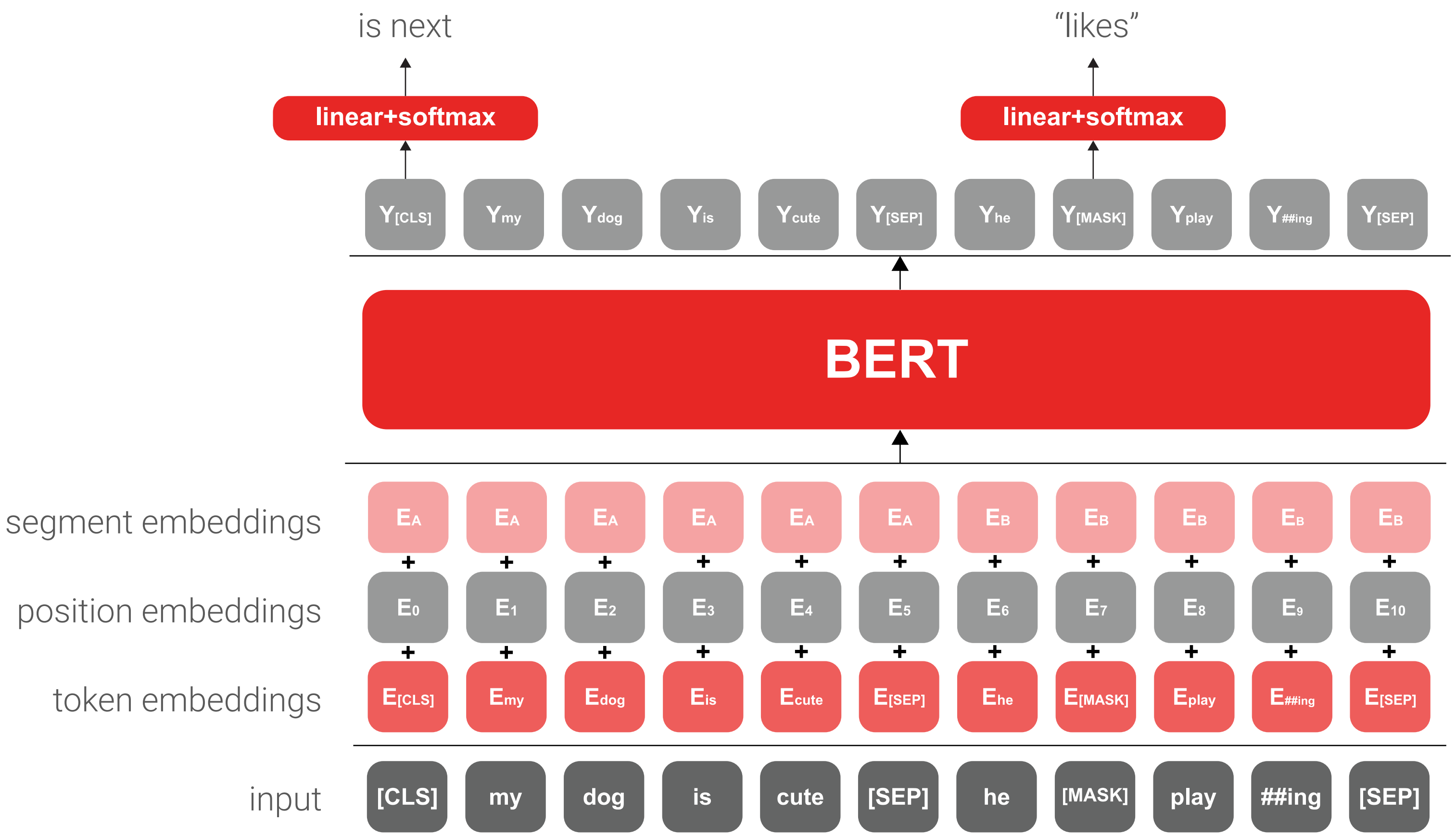
Next sentence prediction

- given a pair of sentences predict if they follow one another;
- aims to learn sentence relationships that are important is certain downstream tasks (e.g. question answering).

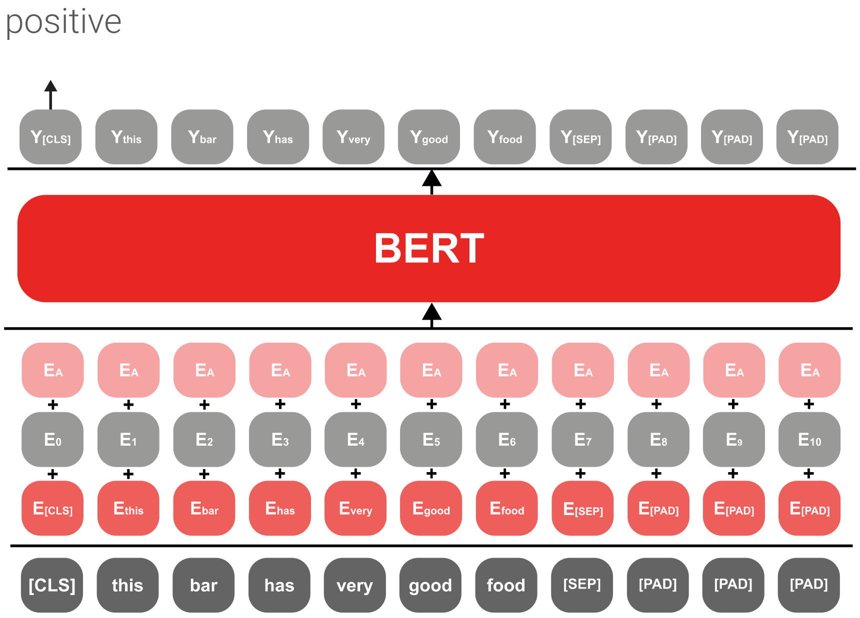
A: ‘What are those?’ he said while looking at my crocs.
B: My new shoes.
Ground truth: next

A: ‘What are those?’ he said while looking at my crocs.
B: The sky is blue.
Ground truth: not next

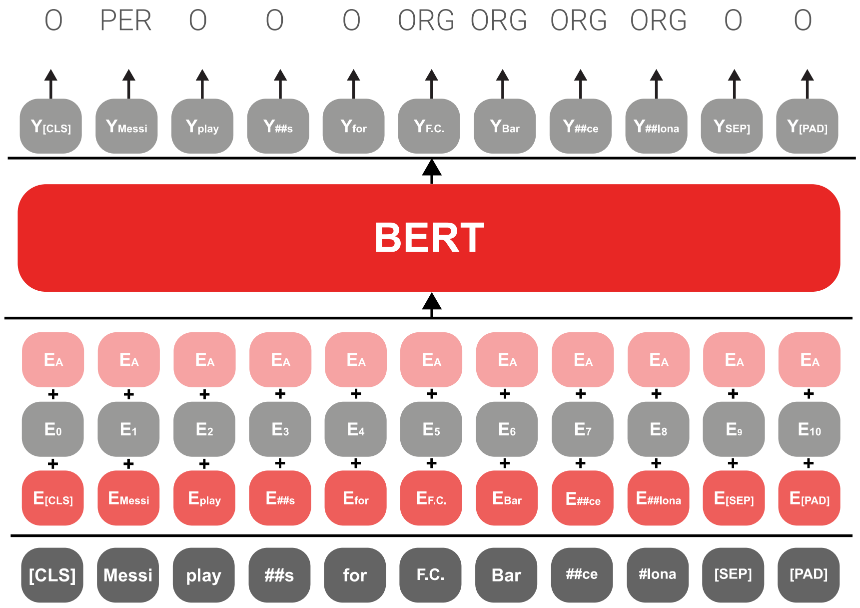
Pre-training



Fine-tuning



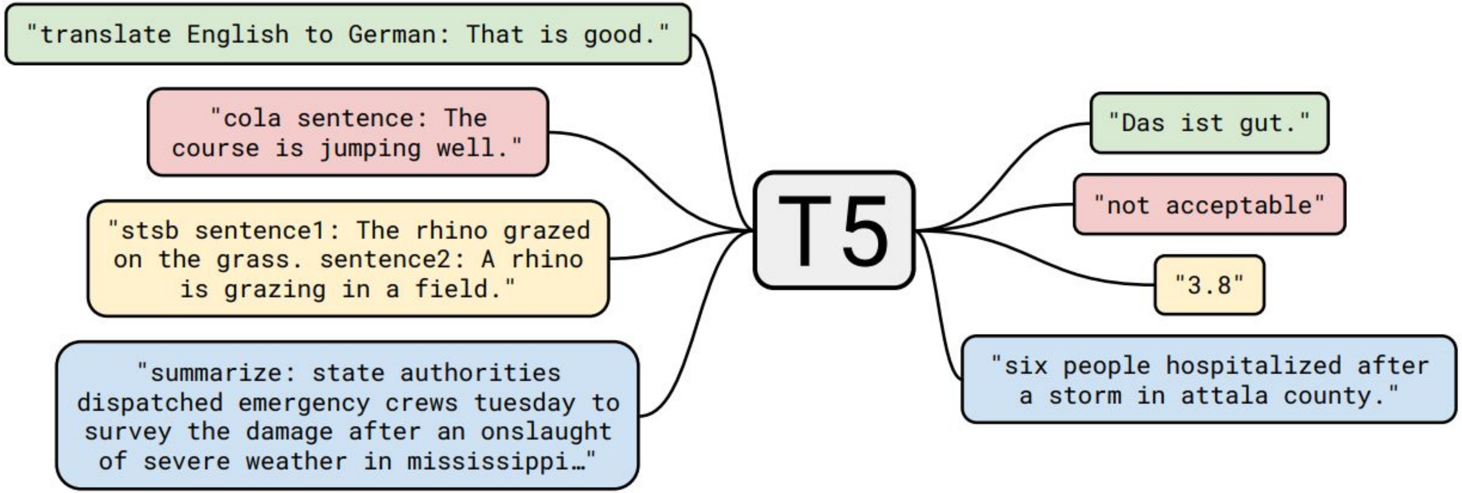
classification



named entity recognition

T5

- encoder-decoder model
- pretraining: span corruption - masking multiple consecutive tokens, the model generates them



Source: [Raffel et al.: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer.]

GPT evolution

GPT-1 [1] (2018)

~117M parameters

~5 GB data

GPT-2 [2] (2019)

~1.5B parameters

~40 GB data

- *zero-shot learning*

GPT-3 [3] (2020)

~175B parameters

~600 GB data

- *in-context learning*

GPT-3.5 (2022)

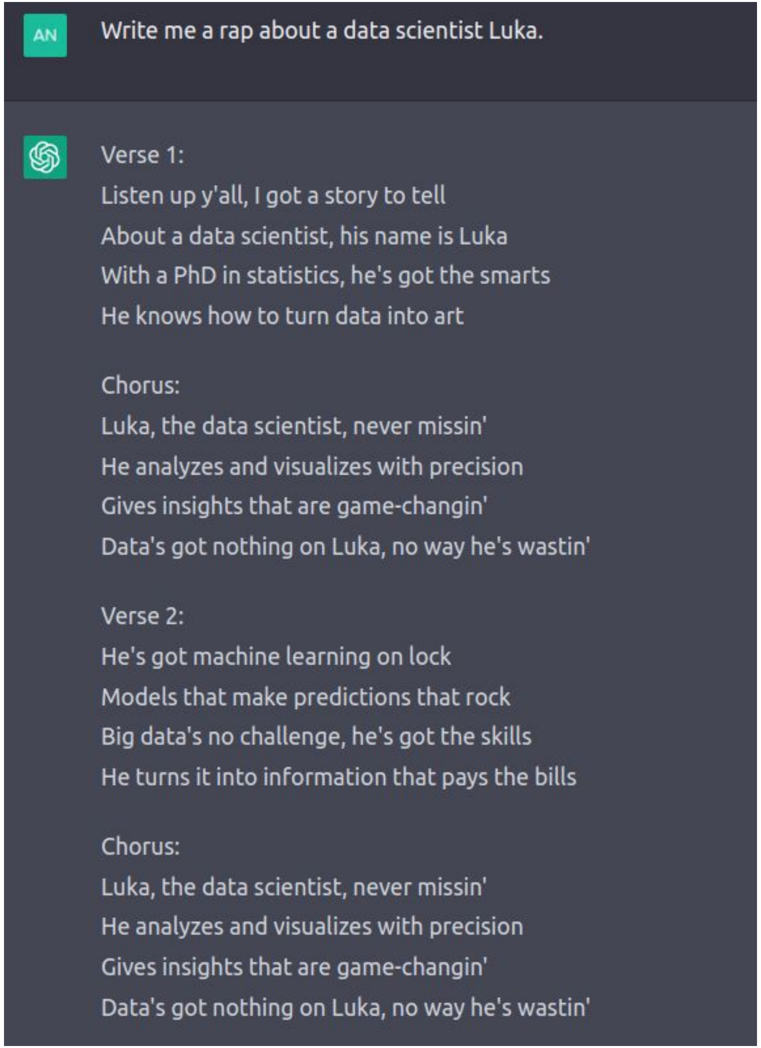
- trained on a mix of text and code

[1] Radford et al.: Improving Language Understanding by Generative Pre-Training, 2018.
[2] Radford et al.: Language Models are Unsupervised Multitask Learners, 2019.
[3] Brown et al.: Language Models are Few-Shot Learners, 2020.

ChatGPT

- chatbot built on top of GPT-3.5 family of models, release in late 2022

- fine-tuned using reinforcement learning from human feedback (RLHF)


AN Write me a rap about a data scientist Luka.

Verse 1:
Listen up y'all, I got a story to tell
About a data scientist, his name is Luka
With a PhD in statistics, he's got the smarts
He knows how to turn data into art

Chorus:
Luka, the data scientist, never missin'
He analyzes and visualizes with precision
Gives insights that are game-changin'
Data's got nothing on Luka, no way he's wastin'

Verse 2:
He's got machine learning on lock
Models that make predictions that rock
Big data's no challenge, he's got the skills
He turns it into information that pays the bills

Chorus:
Luka, the data scientist, never missin'
He analyzes and visualizes with precision
Gives insights that are game-changin'
Data's got nothing on Luka, no way he's wastin'

ChatGPT training

- supervised fine-tuning
(tune to dialog form)

- RLHF
(align to human preference)

- reward penalized based on initial model