Transformers and Large Language Models

Introduction to Large Language Models

Language models

- Remember the simple n-gram language model
  - Assigns probabilities to sequences of words
  - Generate text by sampling possible next words
  - Is trained on counts computed from lots of text
- Large language models are similar and different:
  - Assigns probabilities to sequences of words
  - Generate text by sampling possible next words
  - **Are trained by learning to guess the next word**

Neural Large Language Models (LLMs)

- Self-supervised learners
  - Take a text, remove a word
  - Use your neural model to guess what the word was
  - If the model is wrong, use stochastic gradient descent to make the model guess better next time
- Advantages (?):
  - All we need is a **lot of** text (GPT3: 500 billion tokens)
  - (And a **lot of** compute)

LLMs are built out of transformers

Transformer: a specific kind of network architecture, like a fancier feedforward network, but based on attention

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

A very approximate timeline

1990 Static Word Embeddings

2003 Neural Language Model

2008 Multi-Task Learning

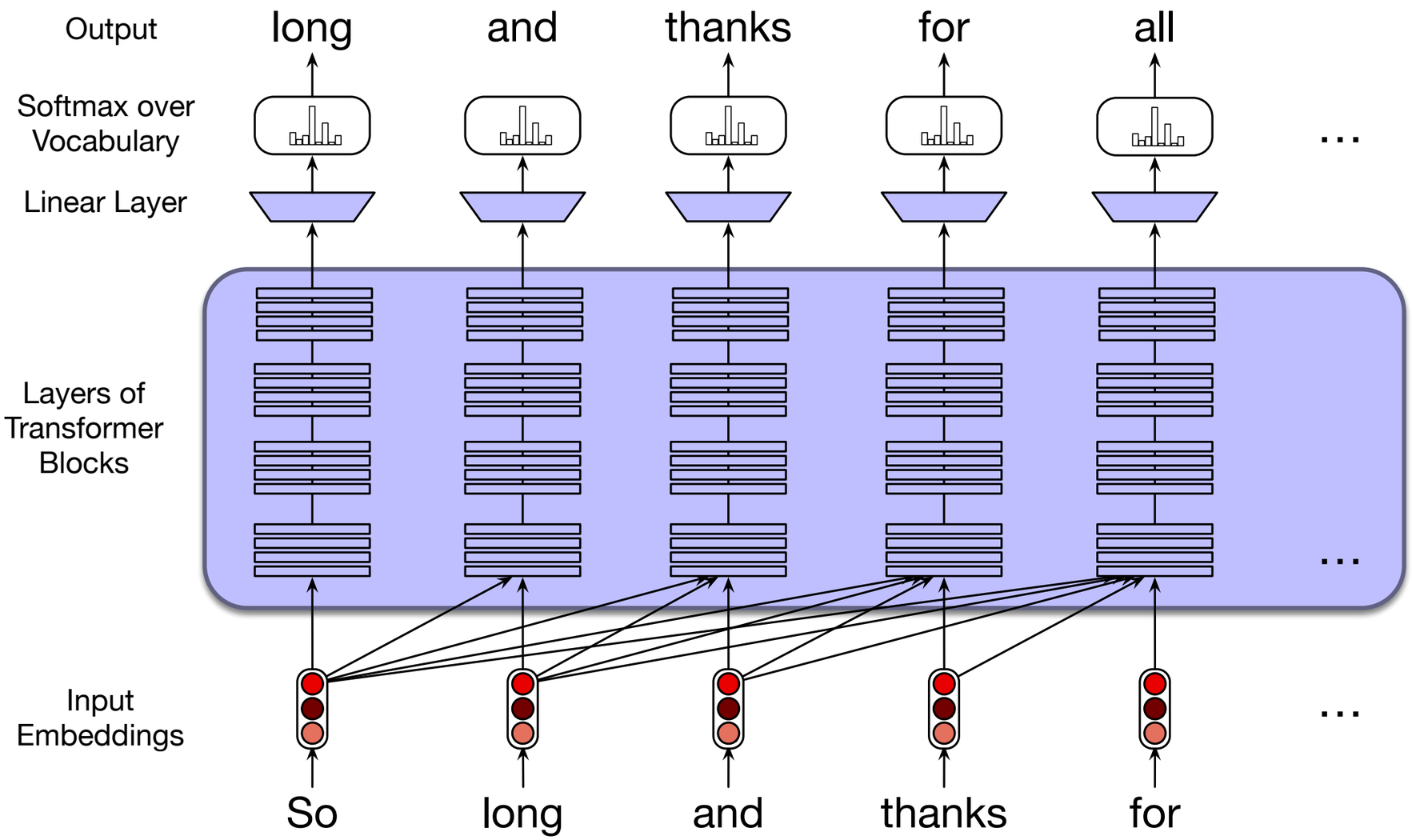
2015 Attention

2017 Transformer

2018 Contextual Word Embeddings and Pretraining

2019 Prompting

A picture of a transformer language model



Transformers and Large Language Models

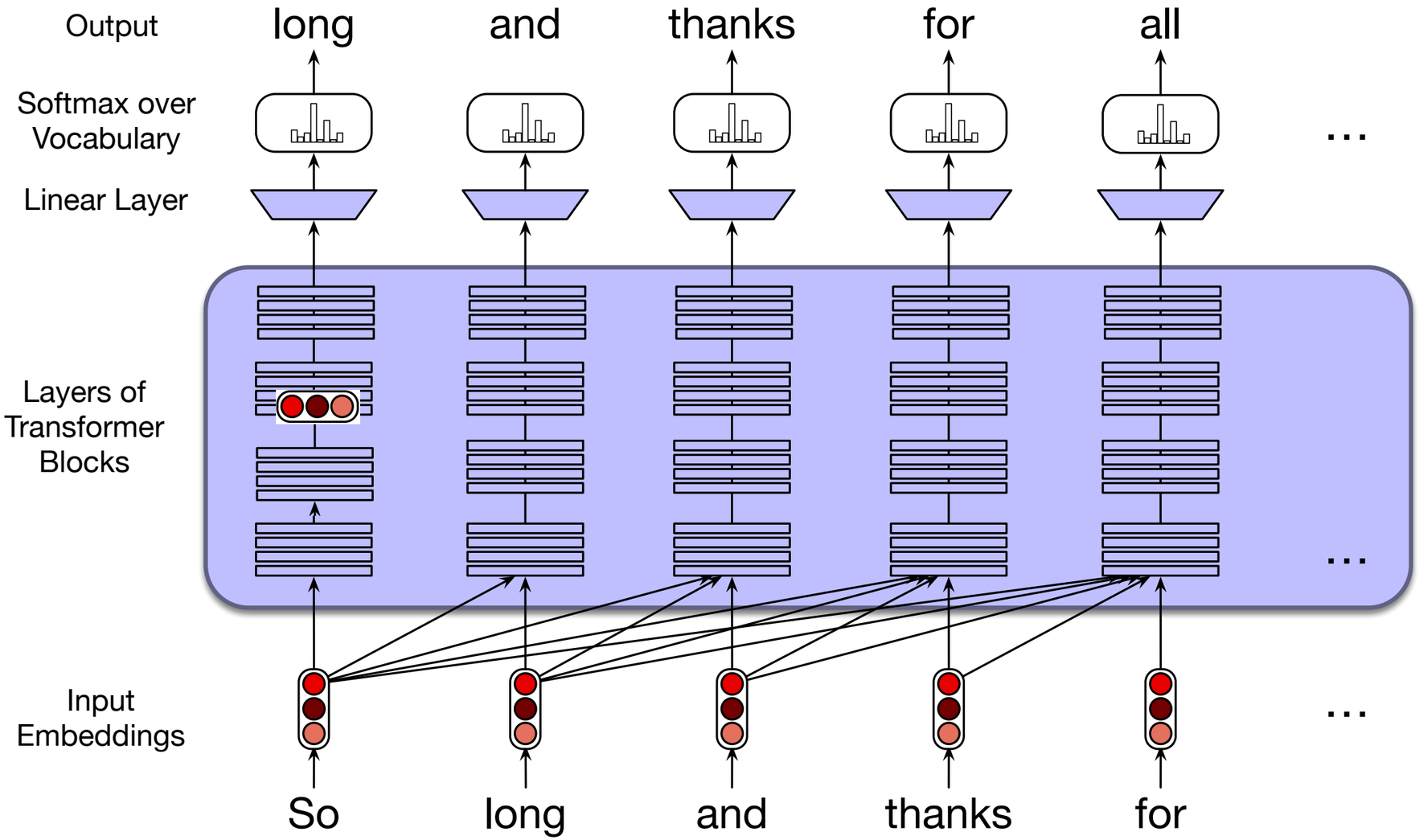
Introduction to Large Language Models

Transformers and Large Language Models

Attention

Instead of starting with the big picture

Let's consider the embeddings for an individual word from a particular layer



Problem with static embeddings (word2vec)

They are static! The embedding for a word doesn't reflect how its meaning changes in context.

What is the meaning represented in the static embedding for "it"?

Contextual Embeddings

- Intuition: a representation of meaning of a word should be different in different contexts!
- **Contextual Embedding**: each word has a different vector that expresses different meanings depending on the surrounding words
- How to compute contextual embeddings?
  - **Attention**

Contextual Embeddings

`The chicken didn't cross the street because it`

What should be the properties of "it"?

`The chicken didn't cross the street because it was too` **`tired`**
`The chicken didn't cross the street because it was too` **`trafficy`**

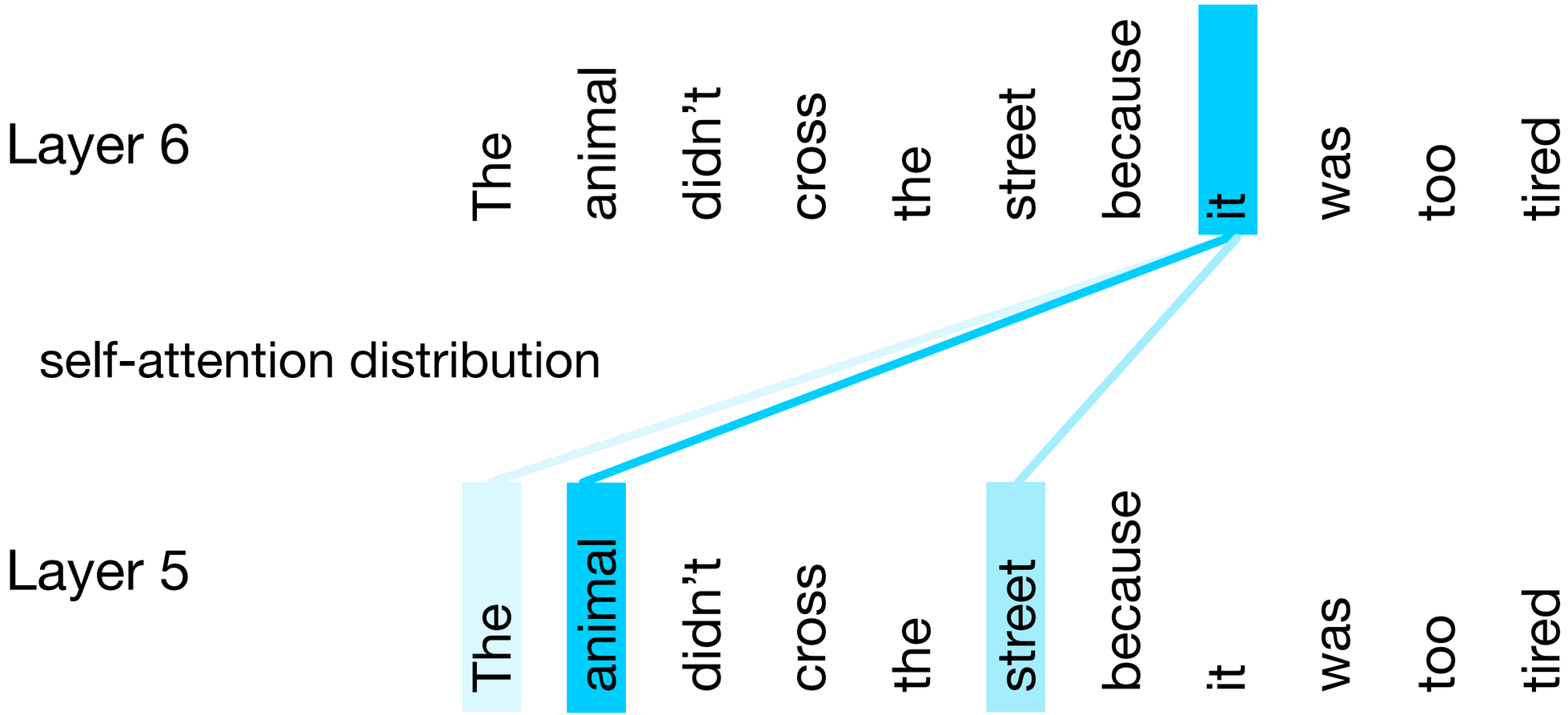
At this point in the sentence, it's probably referring to either the chicken or the street

Intuition of attention

Build up the contextual embedding from a word by selectively integrating information from all the neighboring words

We say that a word "attends to" some neighboring words more than others

Intuition of attention:

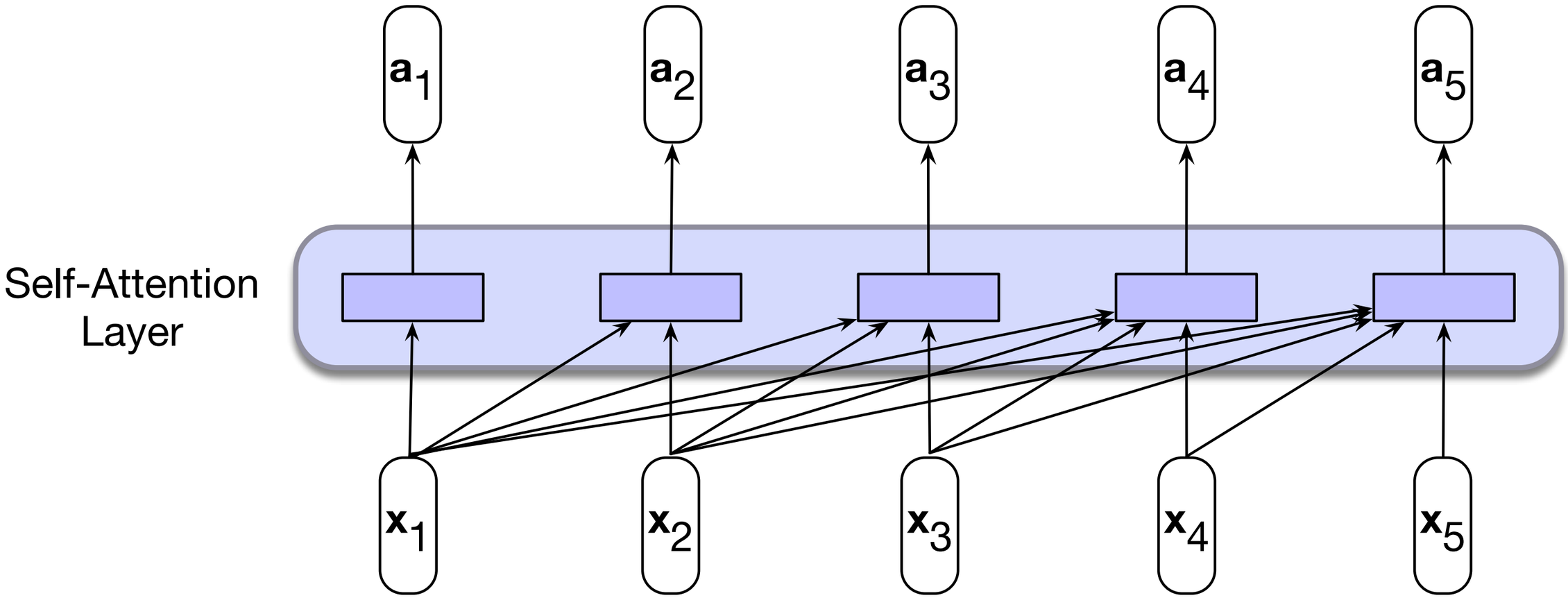


Attention definition

A mechanism for helping compute the embedding for a token by selectively attending to and integrating information from surrounding tokens (at the previous layer).

More formally: a method for doing a weighted sum of vectors.

Attention is left-to-right



Simplified version of attention: a sum of prior words weighted by their similarity with the current word

Given a sequence of token embeddings:

$$\mathbf{x}_1 \quad \mathbf{x}_2 \quad \mathbf{x}_3 \quad \mathbf{x}_4 \quad \mathbf{x}_5 \quad \mathbf{x}_i$$

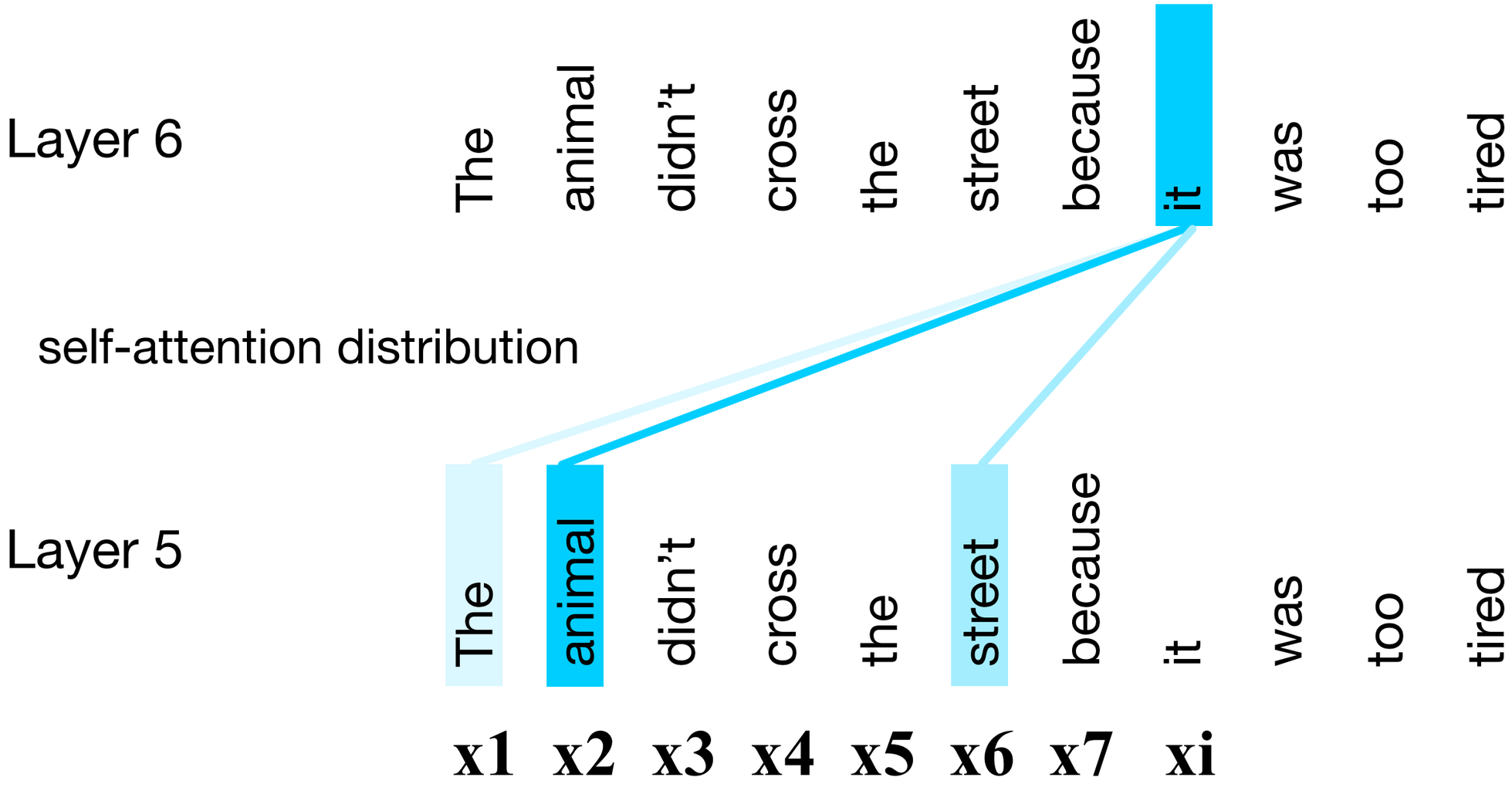
Produce: $\mathbf{a}_i$ = a weighted sum of $\mathbf{x}_1$ through $\mathbf{x}_5$
Weighted by their similarity to $\mathbf{x}_i$

$$\text{score}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$$

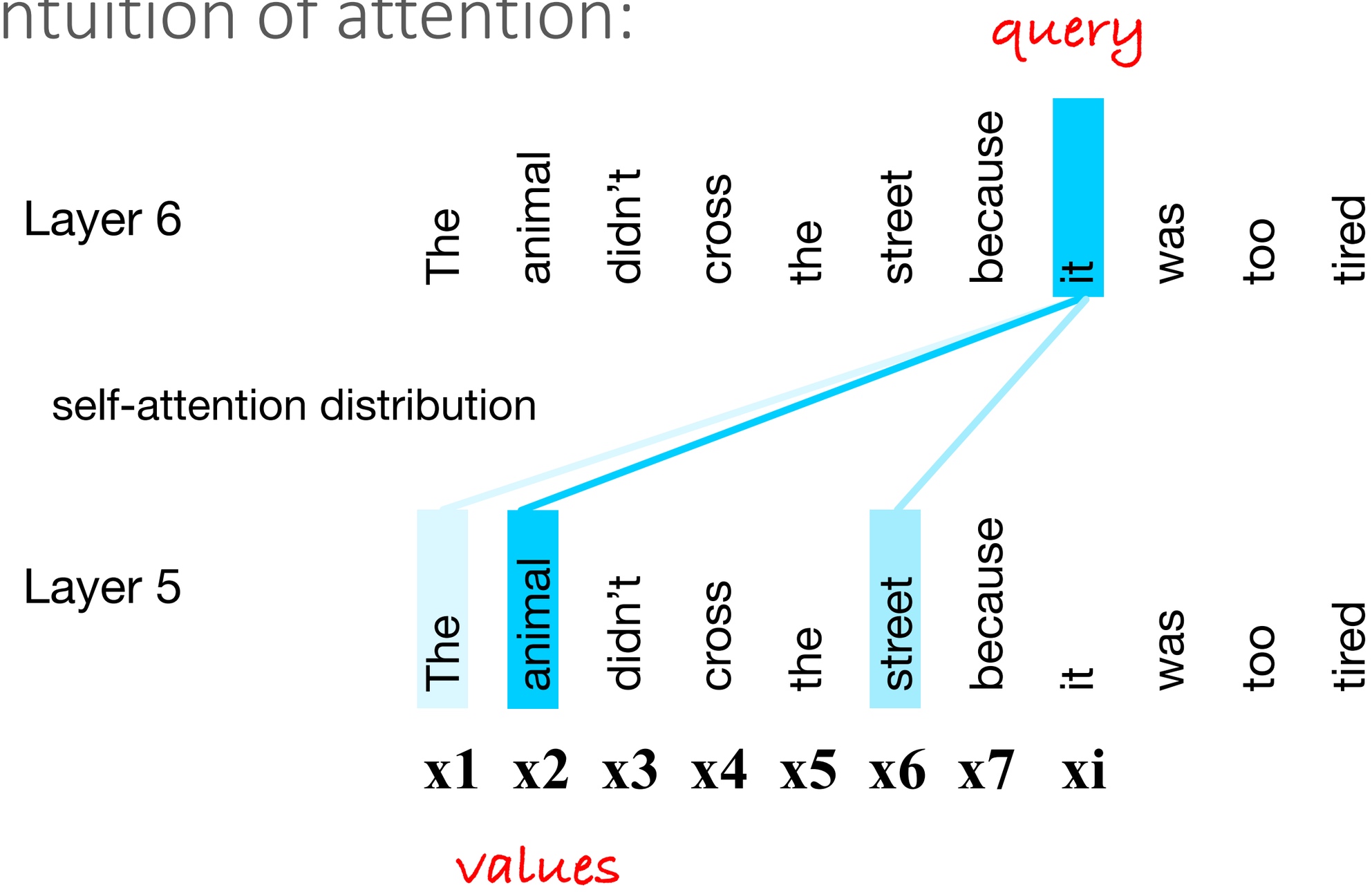
$$\alpha_{ij} = \text{softmax}(\text{score}(\mathbf{x}_i, \mathbf{x}_j)) \quad \forall j \leq i$$

$$\mathbf{a}_i = \sum_{j \leq i} \alpha_{ij} \mathbf{x}_j$$

Intuition of attention:



Intuition of attention:

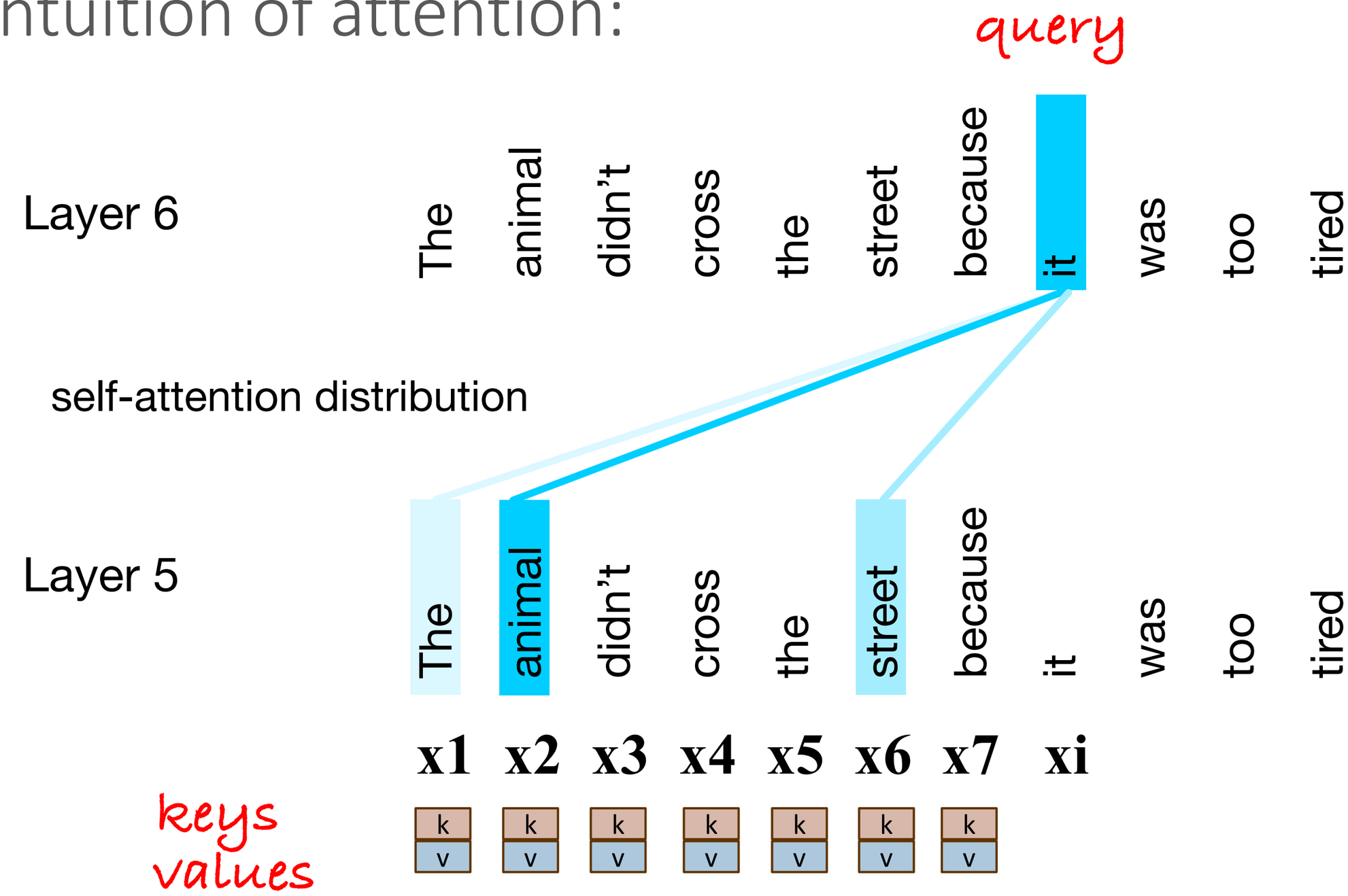


Attention

Actually it's slightly more complicated, but I won't get into that

High-level idea: instead of just having a query and a set of values, each embedding actually also has a key

Intuition of attention:



Summary

Attention is a method for enriching the representation of a token by incorporating contextual information

The result: the embedding for each word will be different in different contexts!

Contextual embeddings: a representation of word meaning in its context.

Transformers and Large Language Models

Attention

Transformers and Large Language Models

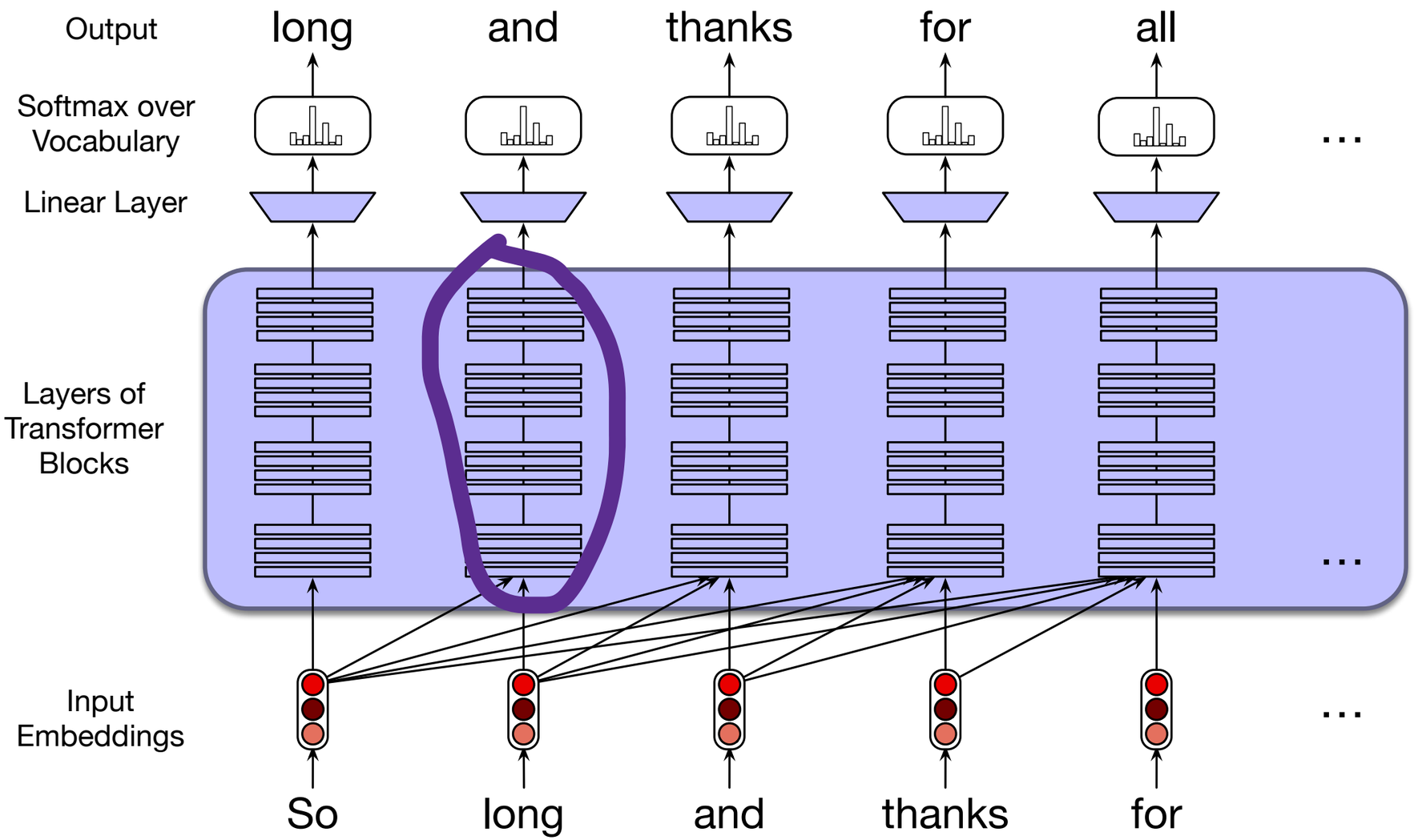
The rest of the transformer applied to language modeling

The transformer

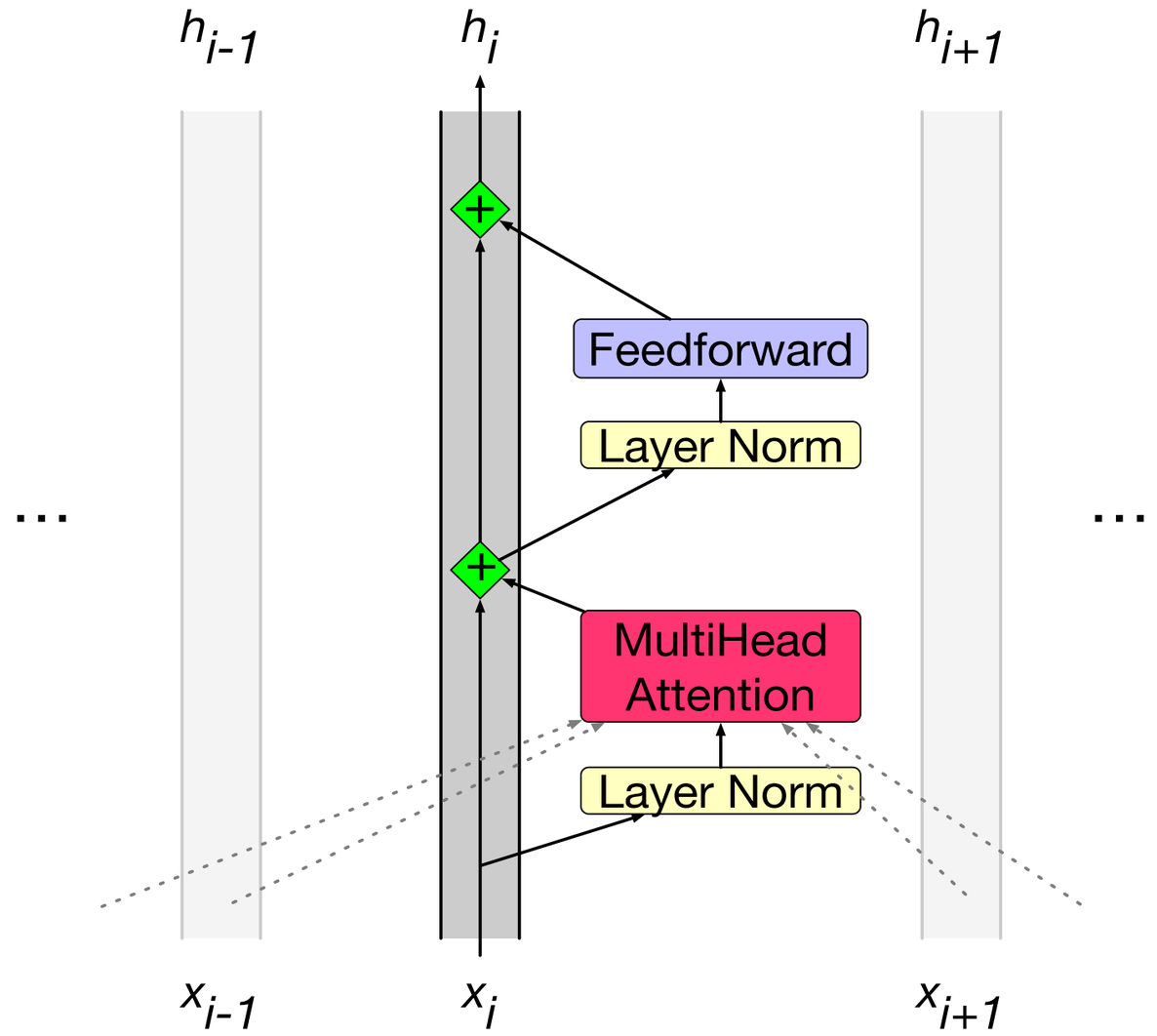
Attention is just part of computing embeddings in a transformer.

Let's see more of the mechanism

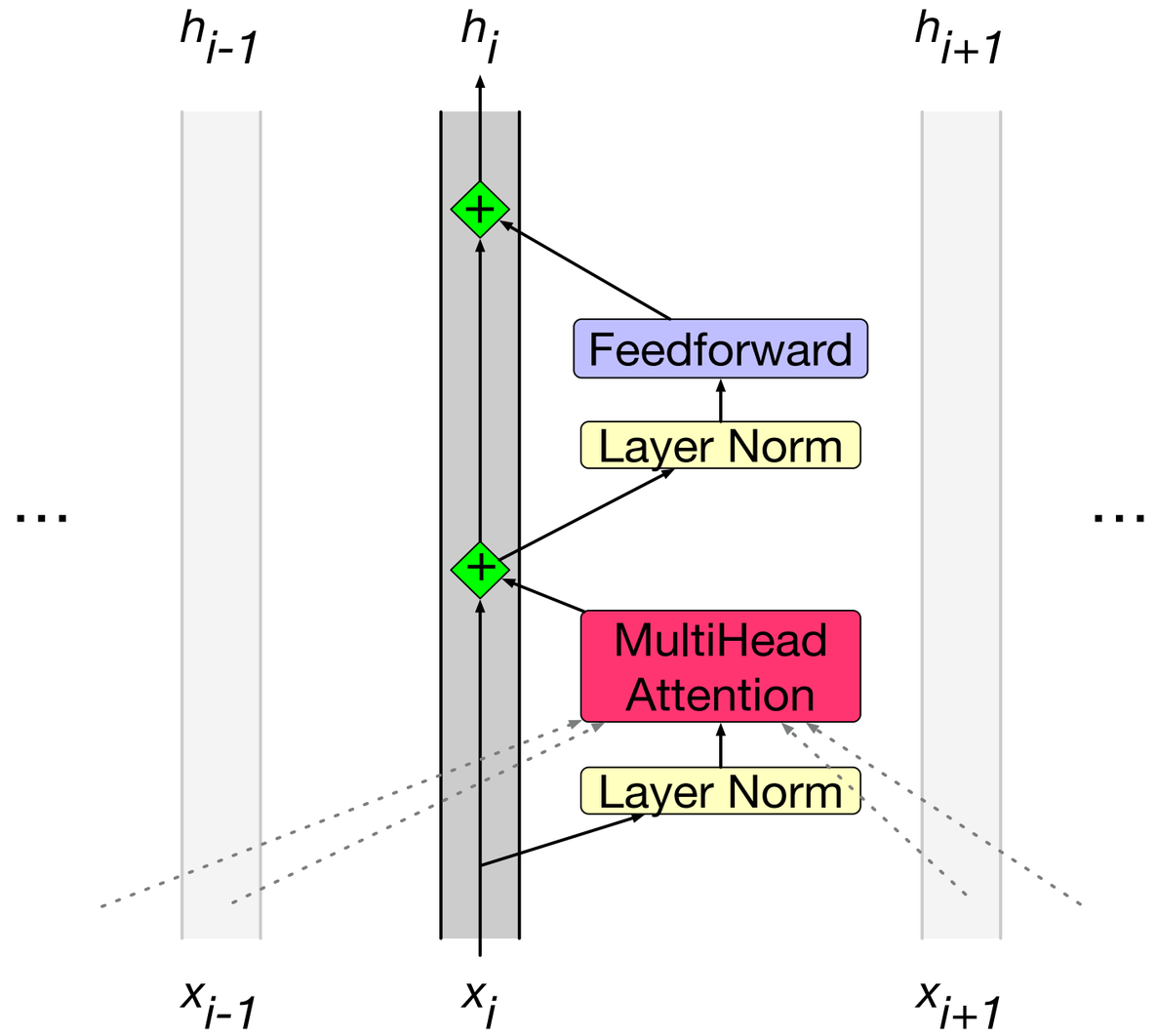
Reminder: transformer language model



The residual stream: each token gets passed up and modified

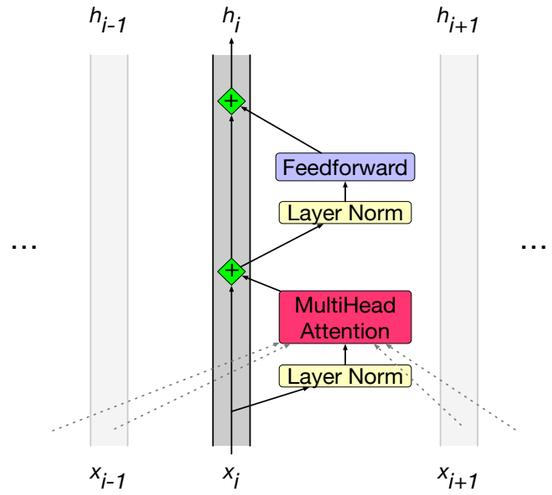


We'll need nonlinearities, so a feedforward layer

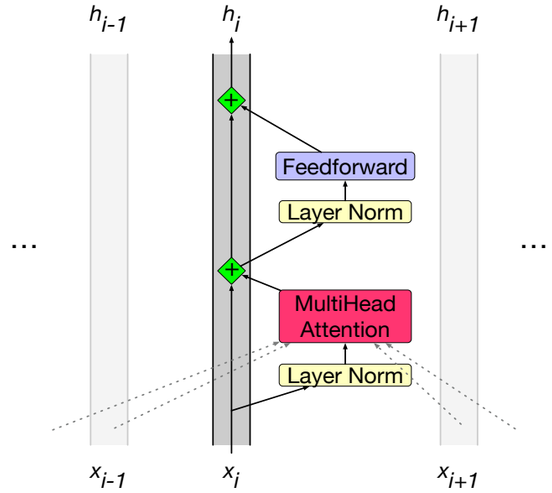


A transformer is a stack of these blocks

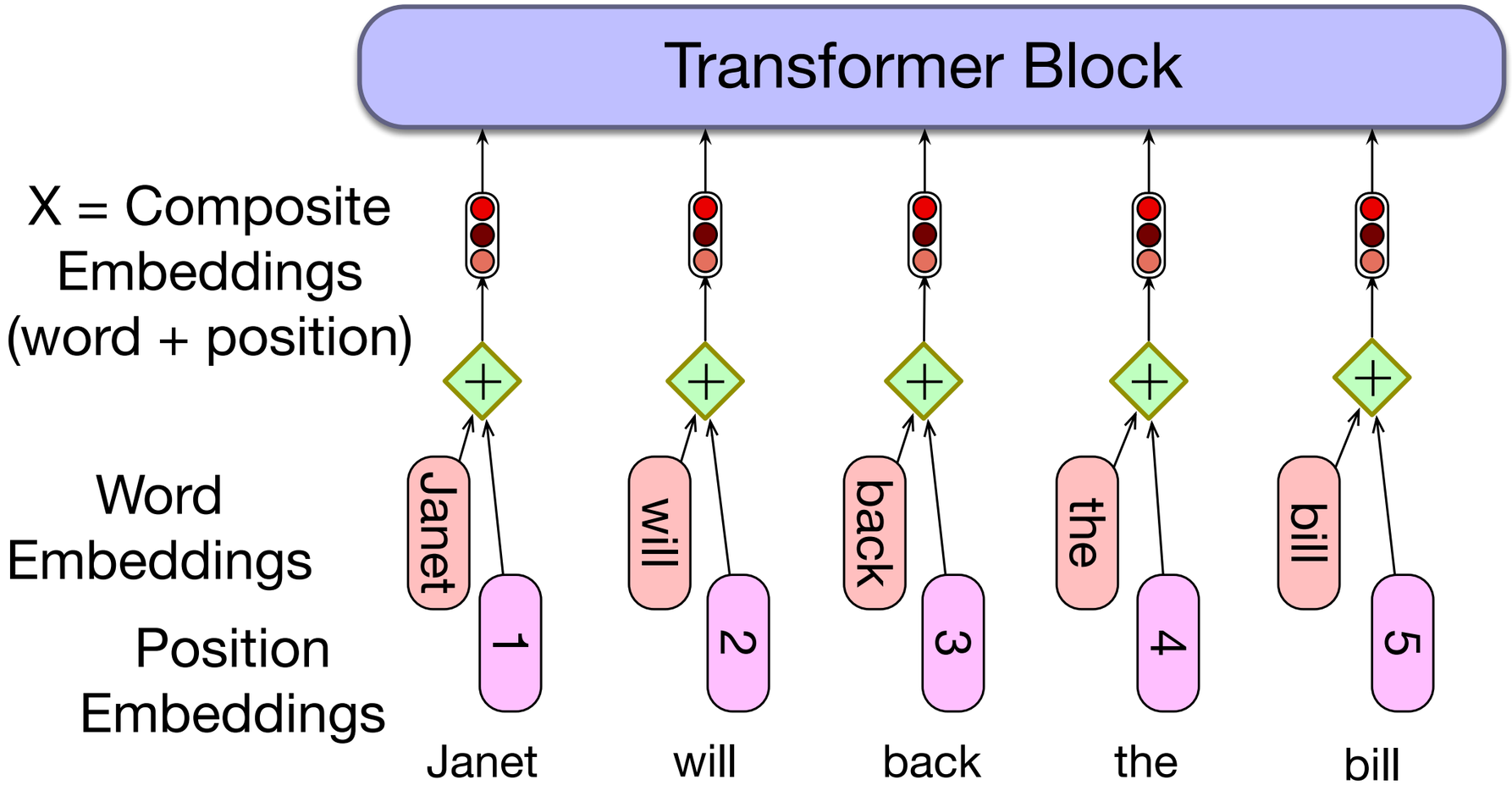
Block 2



Block 1



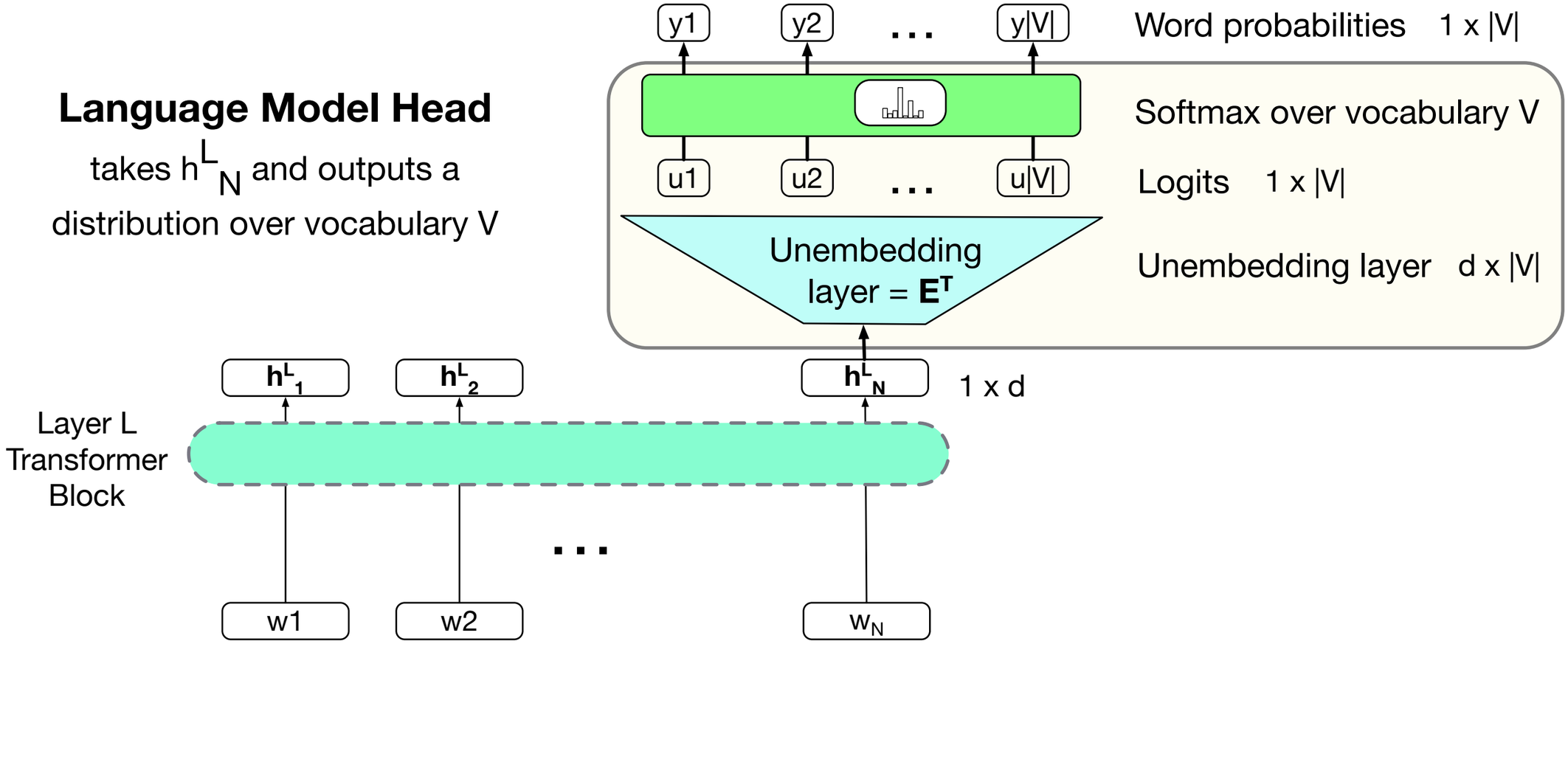
Inputs


Transformer Block
X = Composite Embeddings (word + position)
Word Embeddings
Position Embeddings
Janet
will
back
the
bill
1
2
3
4
5
Janet
will
back
the
bill

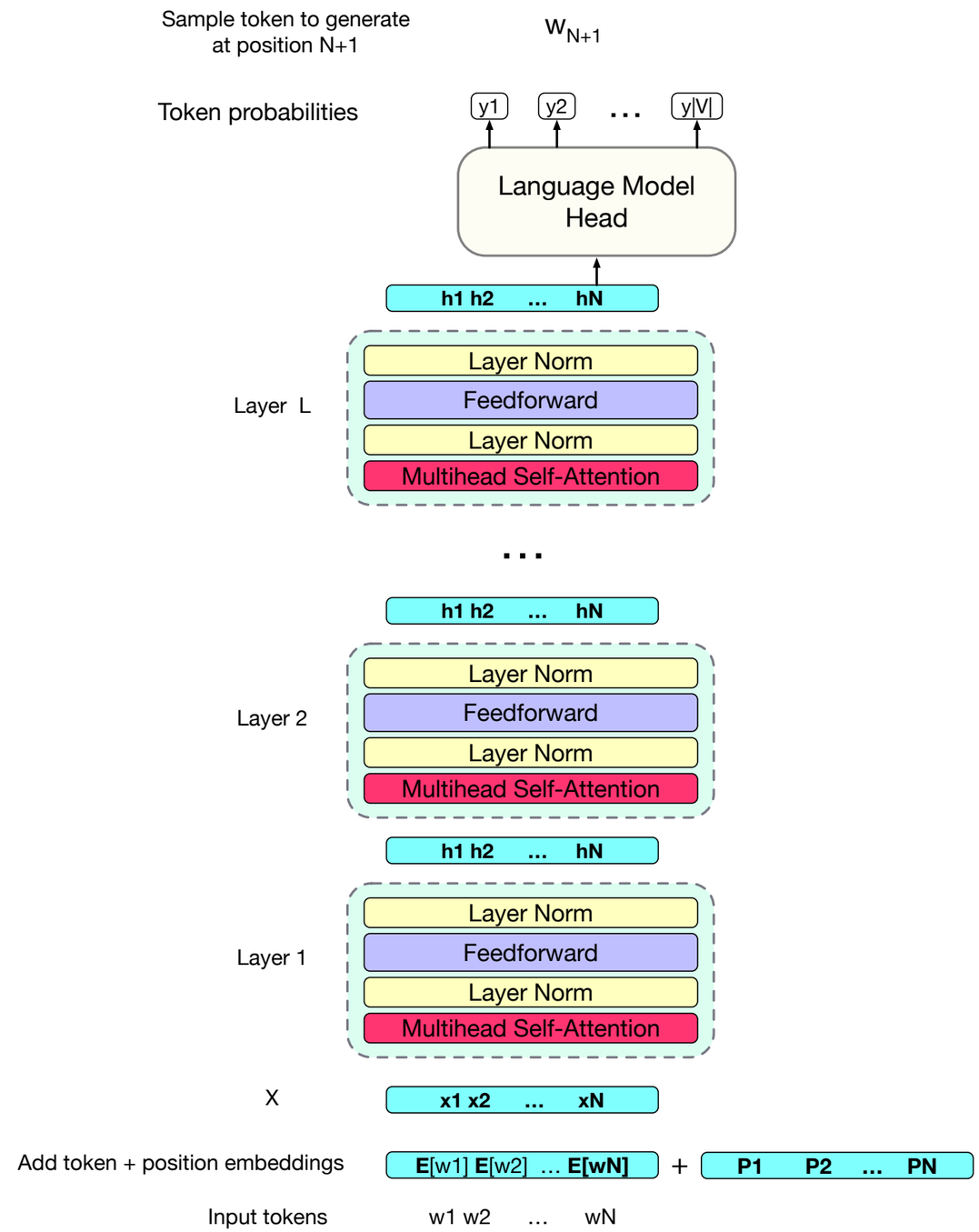
Language modeling head

Language Model Head

takes h^L_N and outputs a distribution over vocabulary V



The final transformer model



Transformers and Large Language Models

The rest of the transformer applied to language modeling

Large Language Models

Pretraining (and how to train transformers for language modeling)

Pretraining

The big idea that underlies all the amazing performance of language models

First **pretrain** a transformer model on enormous amounts of text

Then **apply** it to new tasks.

Intuition of language model training

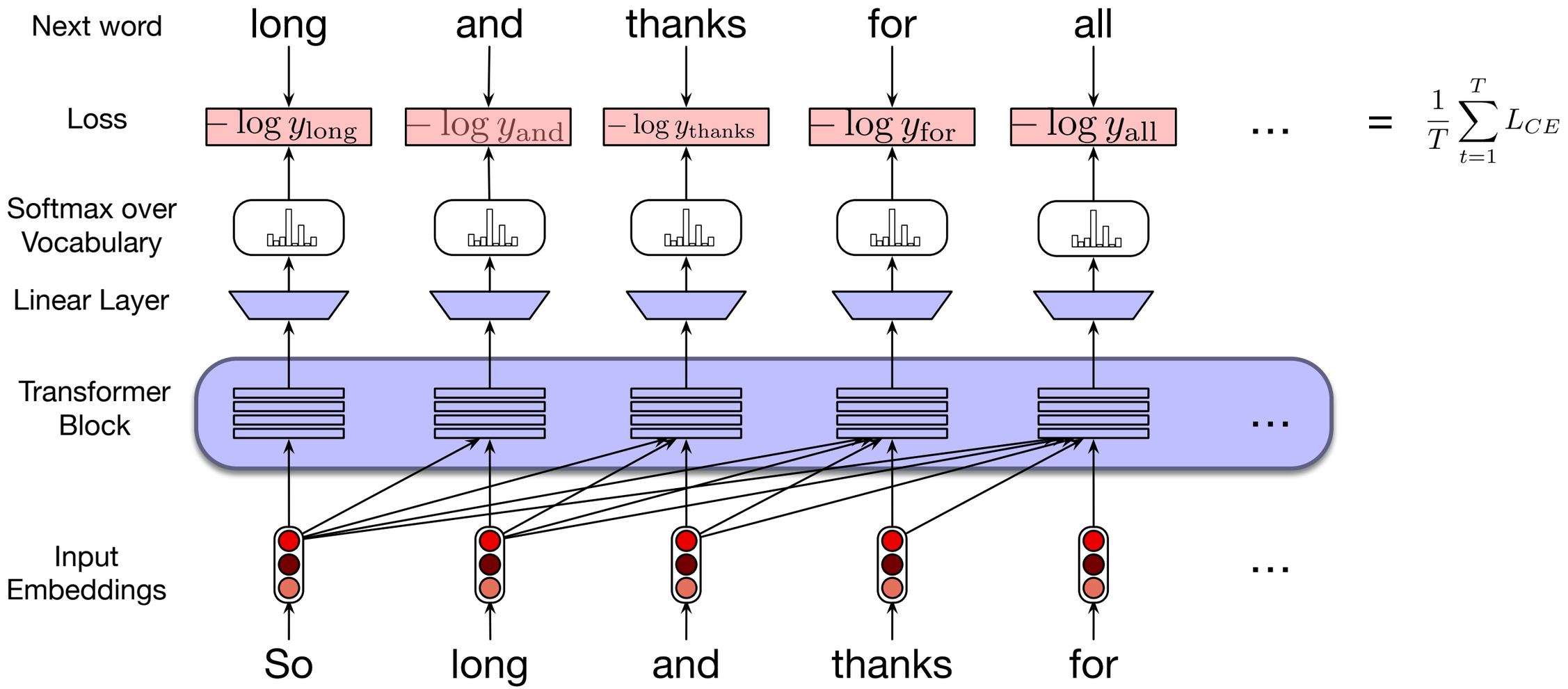
We just train them to predict the next word!

1. Take a corpus of text
2. At each time step *t*
   i. ask the model to predict the next word
   ii. train the model using gradient descent to minimize the error in this prediction

Intuition of language model training: loss

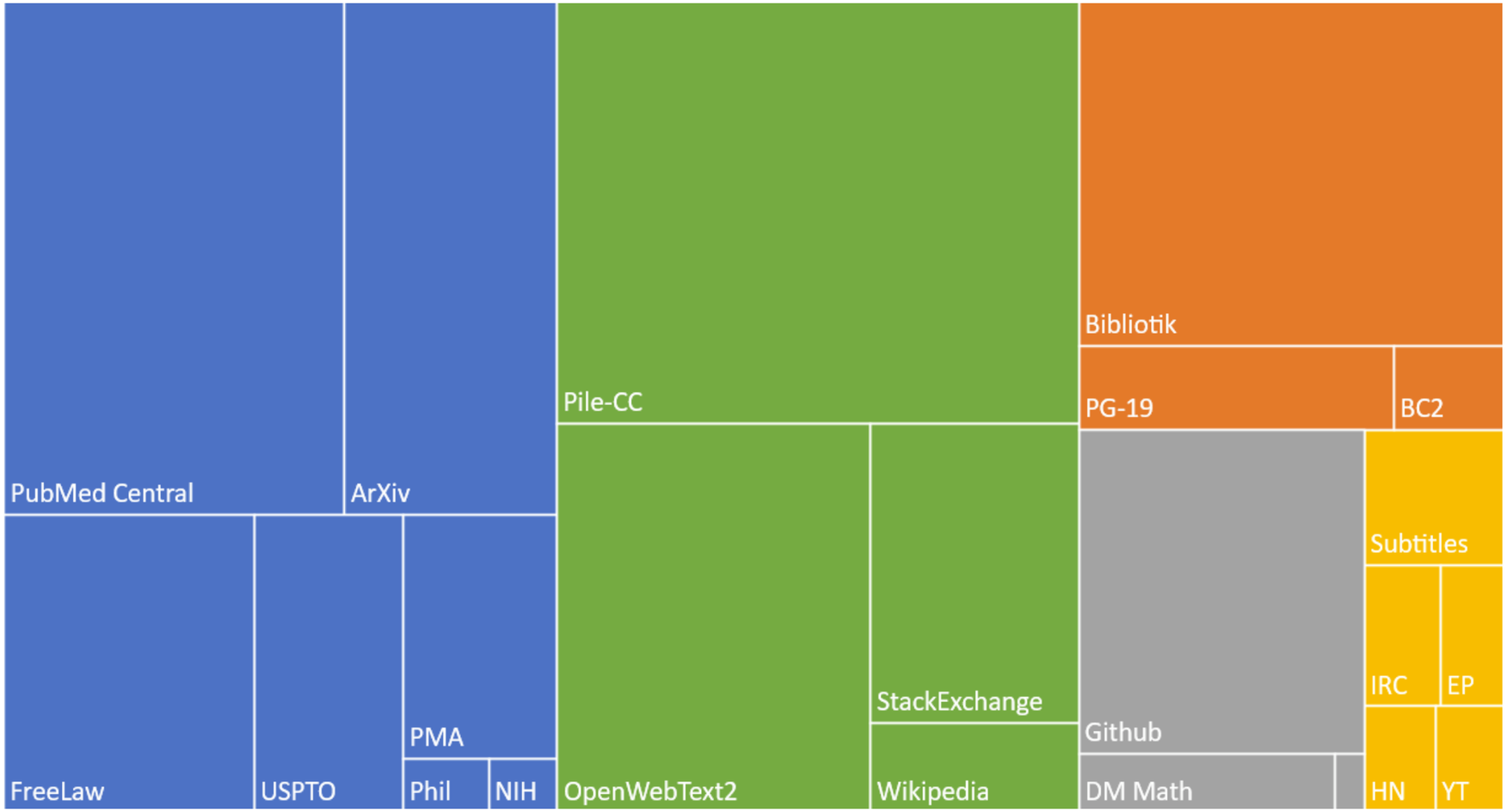
- Same loss function: **cross-entropy loss**
  - We want the model to assign a high probability to true word *w*
  - = want loss to be high if the model assigns too low a probability to w
- CE Loss: The negative log probability that the model assigns to the true next word w
  - If the model assigns too low a probability to w
  - We move the model weights in the direction that assigns a higher probability to w

Training a transformer language model



Pretraining Data: mostly from the web (Common Crawl)

The Pile:



What does a model learn from pretraining?

- There are canines everywhere! One dog in the front room, and two dogs
- It wasn't just big it was enormous
- The author of "A Room of One's Own" is Virginia Woolf
- The doctor told me that he
- The square root of 4 is 2

Big idea

Text contains enormous amounts of knowledge

Pretraining on lots of text with all that knowledge is what gives language models their ability to do so much

Large Language Models

Pretraining (and how to train transformers for language modeling)

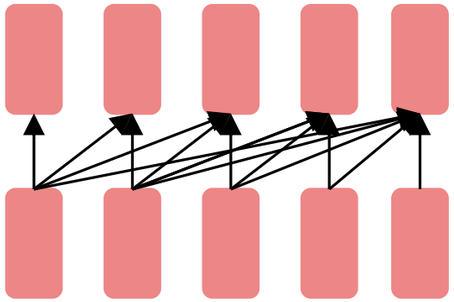
Large Language Models

Large Language Models: Applying pretrained models to new tasks

Big idea

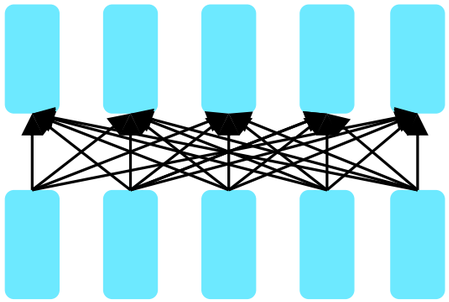
Many tasks can be turned into tasks of predicting words!

Three architectures for large language models



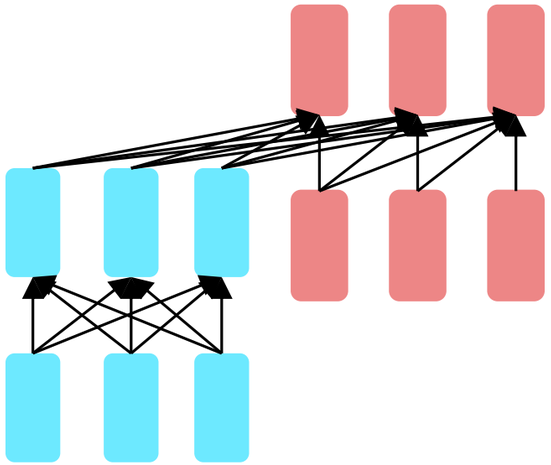
Decoders

GPT, Claude, Llama 2 Mixtral



Encoders

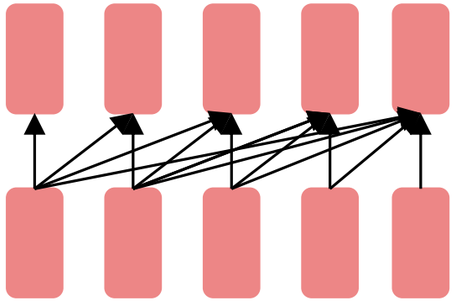
BERT family, HuBERT



Encoder-decoders

Flan-T5, Whisper

Decoders

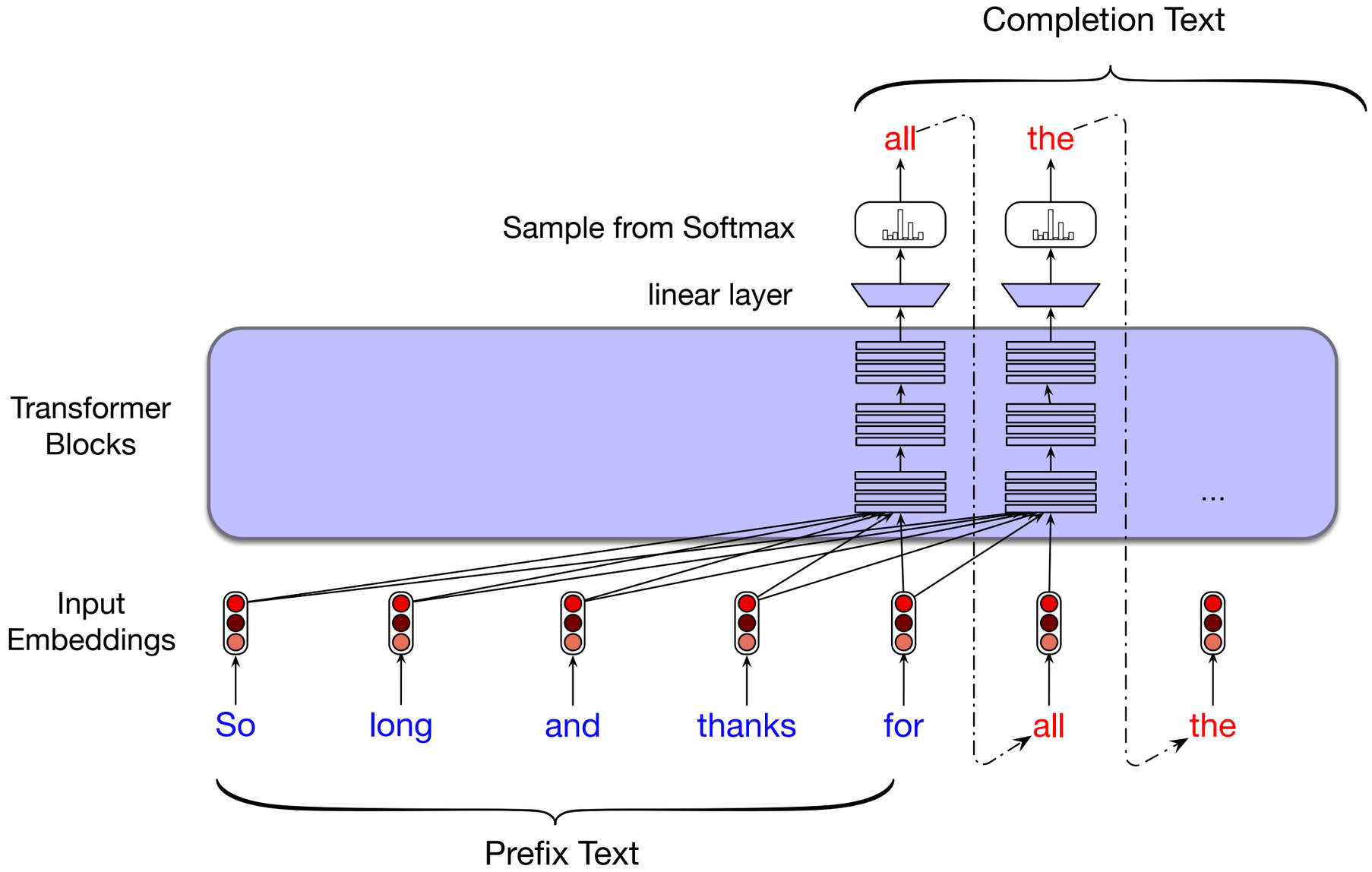


Also called:

- Causal LLMs
- Autoregressive LLMs
- Left-to-right LLMs

- Predict words left to right

Conditional Generation: Generating text conditioned on previous text!



Framing lots of tasks as conditional generation

Sentiment analysis: “I like Jackie Chan”

1. We give the language model this string:
   `The sentiment of the sentence "I like Jackie Chan" is:`
2. And see what word it thinks comes next:

$$P(positive|The\ sentiment\ of\ the\ sentence\ “I\ like\ Jackie\ Chan”\ is:)$$

$$P(negative|The\ sentiment\ of\ the\ sentence\ “I\ like\ Jackie\ Chan”\ is:)$$

Framing lots of tasks as conditional generation

QA: “Who wrote The Origin of Species”

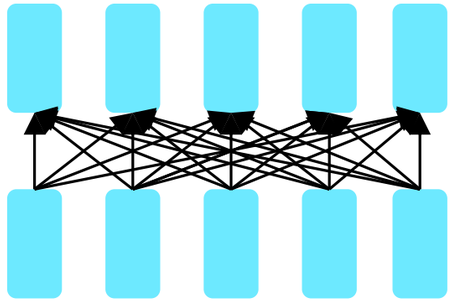
1. We give the language model this string:

```
Q: Who wrote the book ‘‘The Origin of Species"?  A:
```

2. And see what word it thinks comes next:

$$P(w|Q: \textit{Who wrote the book “The Origin of Species”? A:})$$

Encoders

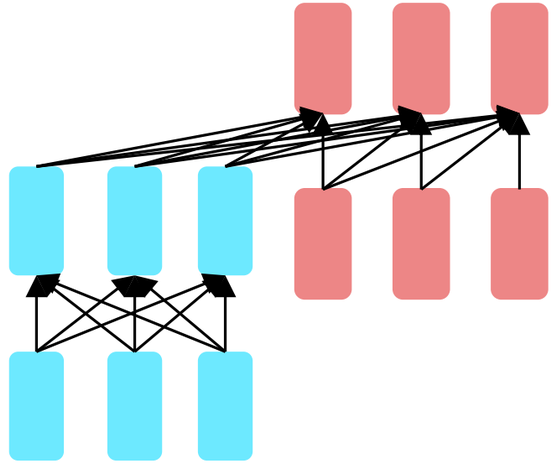


Many varieties!

- Popular: Masked Language Models (MLMs)
- BERT family

- Trained by predicting words from surrounding words on both sides
- Are usually **fine-tuned** (trained on supervised data) for classification tasks.

Encoder-Decoders



- Trained to map from one sequence to another
- Very popular for:
  - machine translation (map from one language to another)
  - speech recognition (map from acoustics to words)

Many more things I didn't talk about

Instruction Fine-tuning

Preference Alignment

Prompt Engineering

Where to learn these? CS224N!

Large Language Models

Large Language Models: Applying pretrained models to new tasks

Large Language Models

Harms of Large Language Models

Hallucination

Chatbots May 'Hallucinate' More Often Than Many Realize

What Can You Do When A.I. Lies About You?

People have little protection or recourse when the technology creates and spreads falsehoods about them.

Air Canada loses court case after its chatbot hallucinated fake policies to a customer

The airline argued that the chatbot itself was liable. The court disagreed.

Current research direction to address hallucination

Retrieval-Augmented Generation (RAG)

Use information retrieval to **retrieve** some passages from a high-quality source

Then use a language model to **generate** an answer from those passages

Copyright

Authors Sue OpenAI Claiming Mass Copyright Infringement of Hundreds of Thousands of Novels

The Times Sues OpenAI and Microsoft Over A.I. Use of Copyrighted Work

Millions of articles from The New York Times were used to train chatbots that now compete with it, the lawsuit said.

Privacy

Toxicity and Abuse

The New AI-Powered Bing Is Threatening Users.

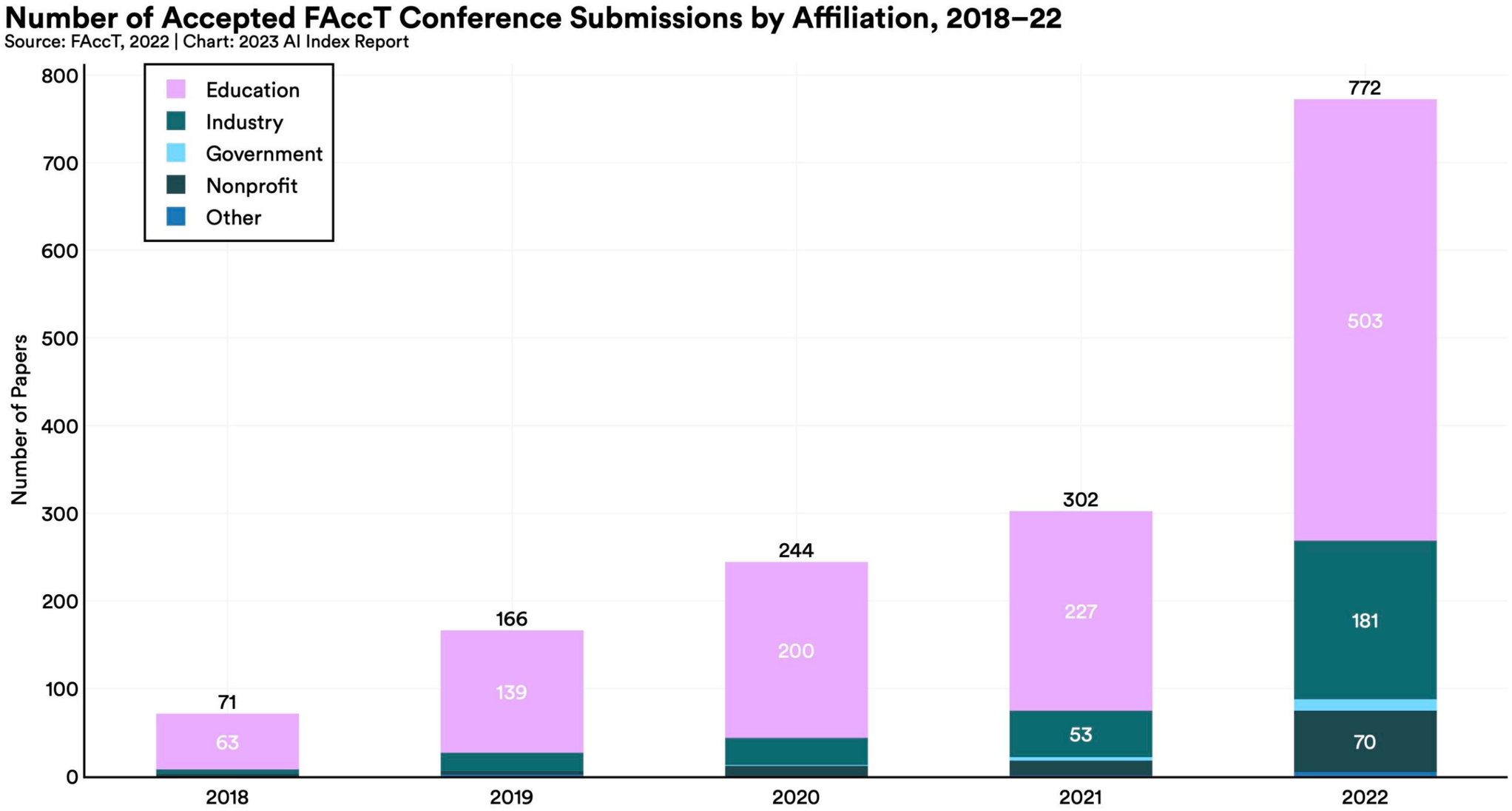
Cleaning Up ChatGPT Takes Heavy Toll on Human Workers

Contractors in Kenya say they were traumatized by effort to screen out descriptions of violence and sexual abuse during run-up to OpenAI's hit chatbot

Misinformation

Chatbots are generating false and misleading information about U.S. elections

Vast growth in interest in Ethics of LLMs and AI



HAI: AI Index 2023

Large Language Models

Harms of Large Language Models

Large Language Models

Our last class together!

Learning goals

Write regular expressions for text tasks
Apply the edit distance algorithm
Build a supervised classifier
Build a search engine
Work with neural word embeddings
Train a neural network
Build a recommendation engine
Build a chatbot
Prompt a large language model

What's next? Spring 2024 NLP adjacent courses

CS224N: Natural Language Processing with Deep Learning (Chris Manning)

Algorithmic internals: transformers, GPT, parsing, machine translation and other applications. More of the gory details! More math, more machine learning

CS224C: NLP for Computational Social Science (Diyi Yang)

…machine learning and theories from social science to study human behaviors and important societal questions at scale. NLP, social networks, causal inference, application to social topics like hate speech, misinformation, and social movements.

CS 224S: Spoken Language Processing: (Andrew Maas)

Introduction to spoken language technology with an emphasis on dialogue and conversational systems.

CS 336: Language Modeling from Scratch (Tatsu Hashimoto and Percy Liang)

every aspect of language model creation, including data collection and cleansing for pre-training, transformer model construction, model training, and evaluation before deployment. Application required.

CS 246: Mining Massive Data Sets (Jure Leskovec)

Next year courses!

CS 224V: Conversational Virtual Assistants with Deep Learning (Monica Lam)

Topics include: (1) growing LLMs' knowledge, (2) stopping LLMs from hallucination (3) experimentation and evaluation of conversational assistants based on LLMs, (5) controlling LLMs to achieve tasks, (6) persuasive LLMs, (7) multilingual assistants, and (8) combining voice and graphical interfaces.

CS329X Human Centered NLP) (Diyi Yang)

human-centered design thinking in NLP, human-in-the-loop algorithms, fairness, and accessibility.

CS329R Race and NLP (Dan Jurafsky and Jennifer Eberhardt)

Integrate methods from natural language processing with social psychological perspectives on race to build practical systems that address significant societal issues

Fun courses outside of CS

Spring:

Linguist 173: Invented Languages

Linguist 134A: The Structure of Discourse

Linguist 156: Language, Gender, and Sexuality

COMM 154: The Politics of Algorithms

Next year:

Linguistics 150: Language and Society

Linguistics 130a: Introduction to semantics & pragmatics