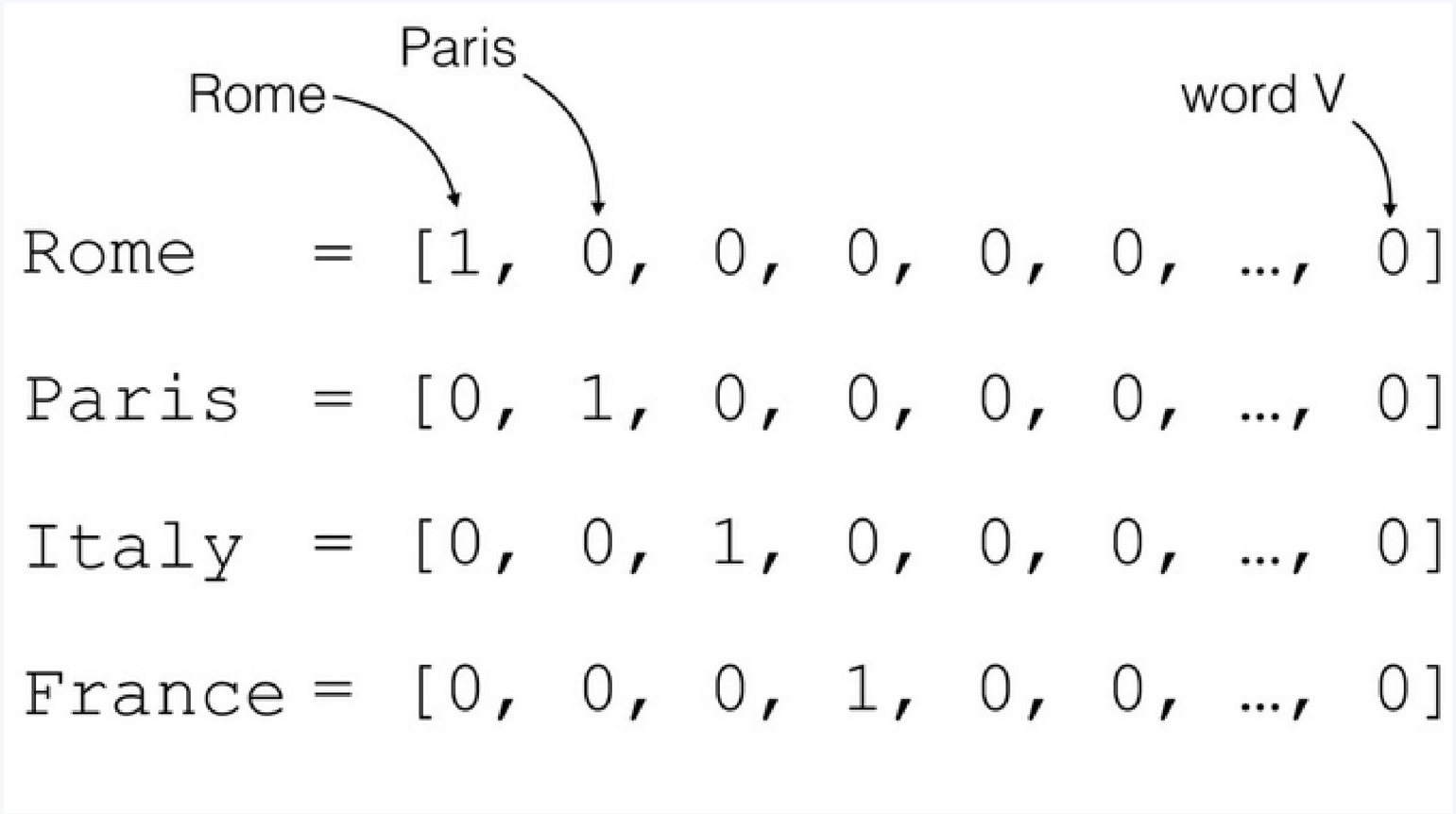
Large Language Models
Part 2

How does it work?
Explaining **Transformers**

What are **Tokens/Embeddings?**


Rome
Paris
word V
Rome = [1, 0, 0, 0, 0, 0, …, 0]
Paris = [0, 1, 0, 0, 0, 0, …, 0]
Italy = [0, 0, 1, 0, 0, 0, …, 0]
France = [0, 0, 0, 1, 0, 0, …, 0]

apple

app

le

token 1

token 2

In GPT3, The vocabulary depends only on the frequency of tokens.

Problem:

We went to use unicode.

There are thousands of unicode characters.

The vocabulary would be very big.

Solution:

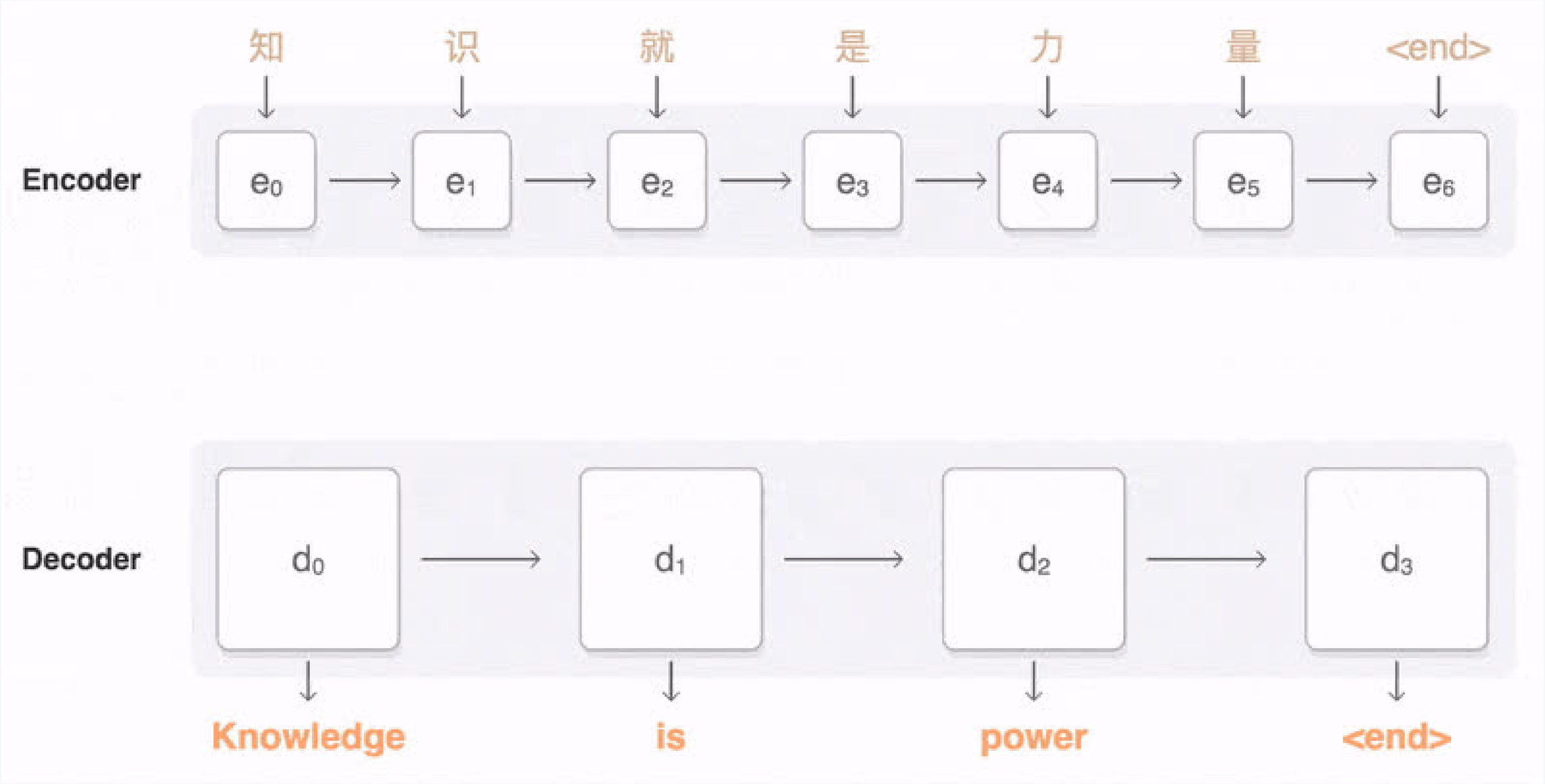
Instead of having each base Unicode character as the initial vocabulary, the tokenization are performed not on the character level, but on the byte level.

This is called **byte-pair encoding**.

Can deep learning do NLP?

RNN & LSTM

RNN & LSTM


知
识
就
是
力
量
<end>
Encoder
e0
e1
e2
e3
e4
e5
e6
Decoder
d0
d1
d2
d3
Knowledge
is
power
<end>

2 problems:

- RNNs are slow
- RNNs can forget

Transformer

Type of Machine Learning model designed for **processing sequences**
(like sequence of words)

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 ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

1 Introduction

Recurrent neural networks, long short-term memory [13] and gated recurrent [7] neural networks in particular, have been firmly established as state of the art approaches in sequence modeling and

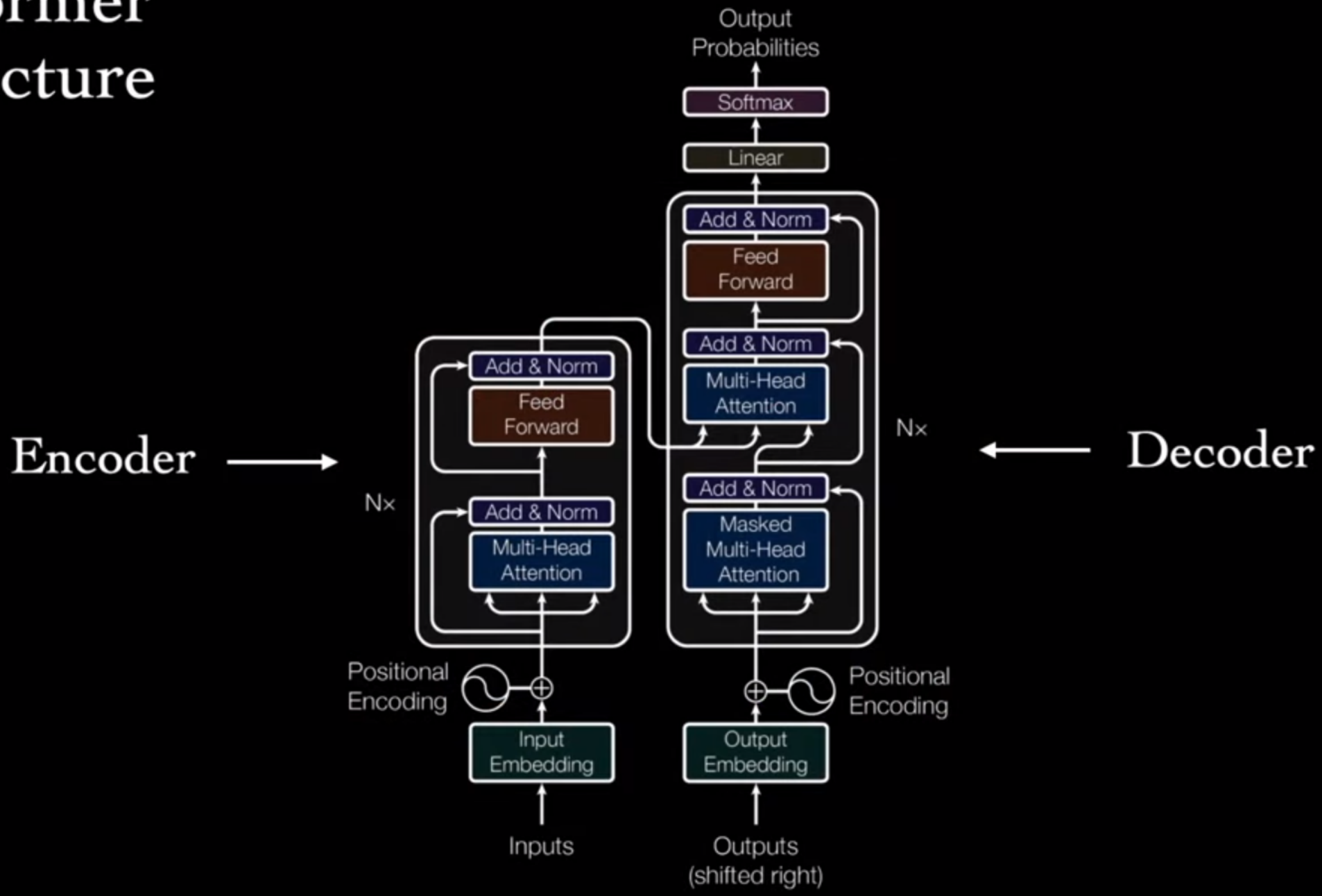
[*]Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

[†]Work performed while at Google Brain.

[‡]Work performed while at Google Research.

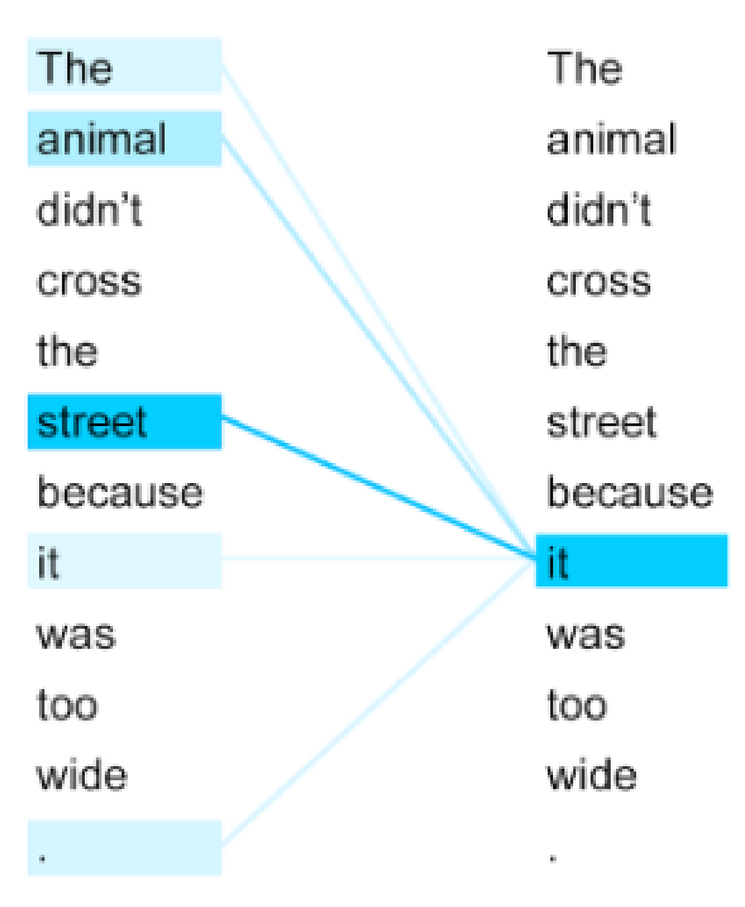
31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

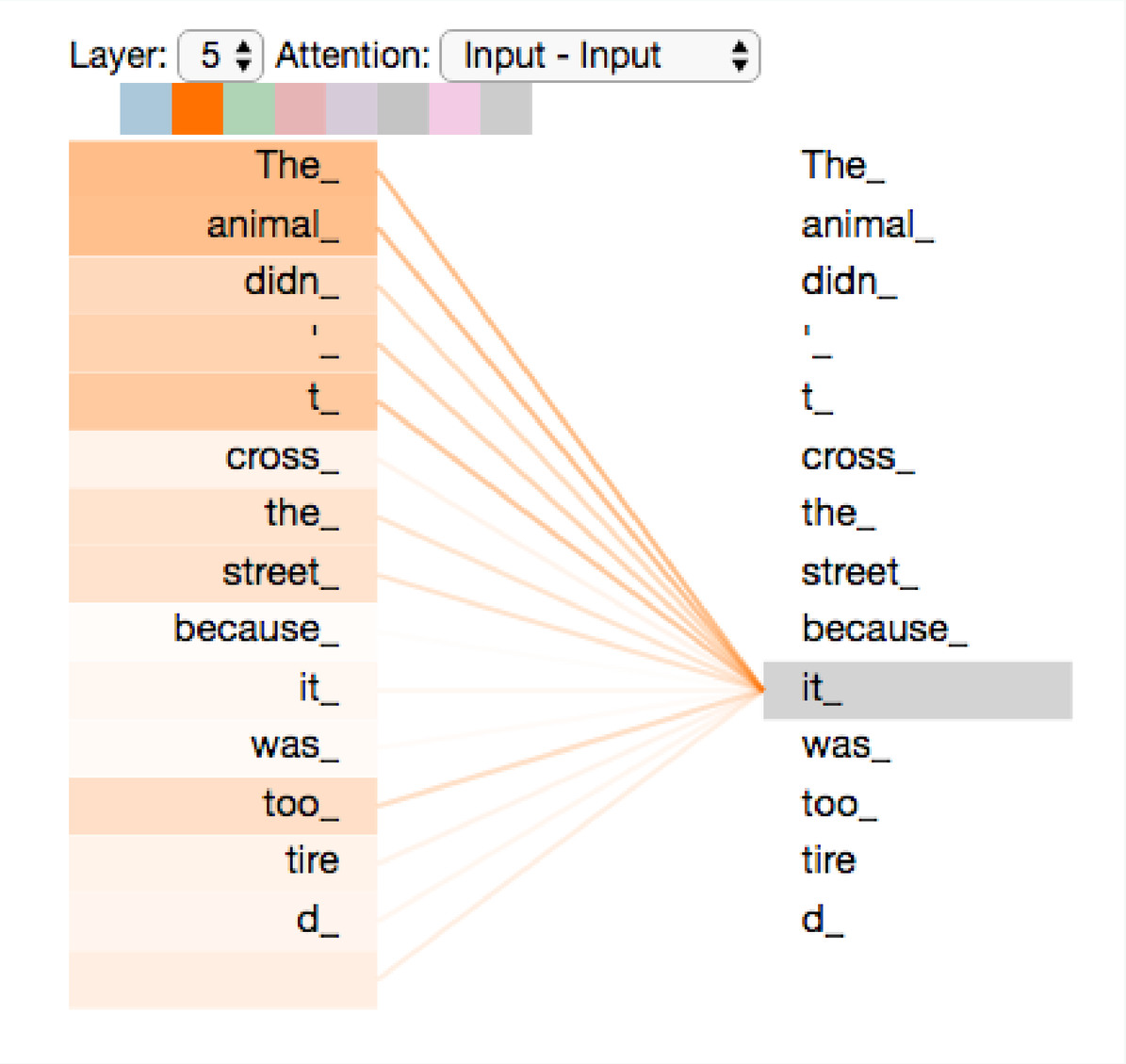
Transformer Architecture



Transformers process the whole sequence at once!

Transformers rely on self-attention!





Reasoning abilities of **Large Language Models**

Does GPT-3 understand the world?

Like all deep learning systems, GPT-3 looks for patterns in data.

It has been trained on a huge corpus of text that it’s mined for statistical regularities.

These regularities are stored as billions of weighted connections between the different nodes in GPT-3’s neural network.

You: "Neglect air resistance, and also neglect gravity. Pretend gravity doesn't exist. Suppose I shoot a bullet straight into the sky. Where does the bullet go?"

Hawking: "It keeps going up, and never comes back."

You: "Right. Let's think about a different version of the same problem. Neglect only air resistance, but include gravity in your reasoning. Suppose I shoot a bullet straight into the sky. Where does the bullet go?"

Hawking: "It falls to the ground."

You: "Why?"

Hawking: "Because... gravity... duh.

Can it reason?

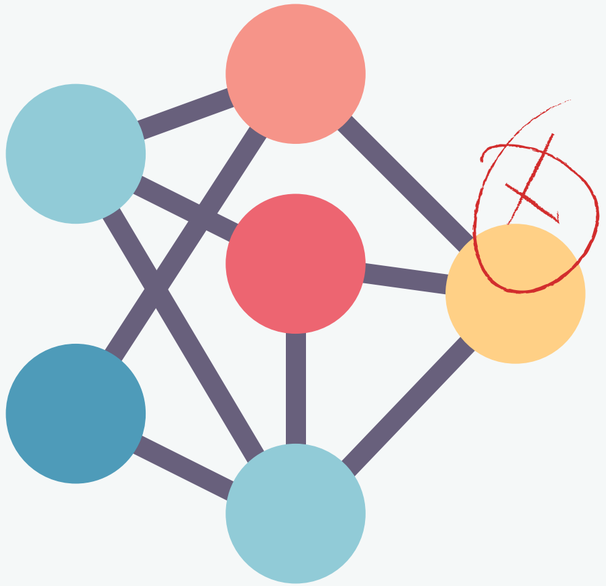
Fine-tuning

Through Prompt Engineering

Simply write a prompt.

Through Fine-tuning

Train GPT on small-scale dataset.



Summary

Why use GPT-3 instead of your
custom NLP models?

GPT-3 requires less data or no data at all
to solve complex NLP tasks!

How can I use Large Language Models?

- An addon that allows you to select a text and simplify it.
- Automatic extraction of hashtags for posts on your website.
- Simple search engine for your website that understands the queries.
- A discord bot for your server!

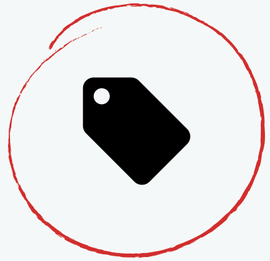
Hundreds of products use OpenAI API, explore them here:
https://gpt3demo.com/map

Business use cases - fast development of **typical NLP** and with **less data**!

- **Sentiment Analysis** (analyze social media posts, product reviews, or online surveys)
- **Text Extraction** (pull out relevant information from text, like keywords, specs, product names, entities, etc.)
- **Chatbots** (use question answering endpoint to answer customer questions)
- **Topic Classification** (analyze thousands of customer surveys, tickets and responses)
  - What percentage of customers talk about "*Pricing*"?
  - Automate the process of tagging incoming support tickets and automatically route them to the right person
- **Search Engines and Recommendation Systems** (internal tools for searching over knowledge database)

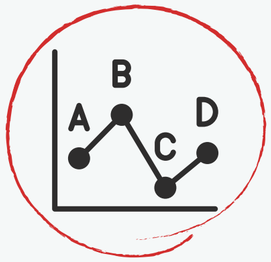
OpenAI API Endpoints

Semantic Search



Text Classification

Question Answering



Embeddings

Semantic Search

Query

This can be any text!

Demo Time

How is the **similarity score** computed?

The search query and each document are combined into prompts sent to the model.

Possibility

Please answer with 'yes' if the following text is similar to the query.

Text:
"""
(content of the document)
"""

Query:
"""
(content of the query)
"""

Answer:

(perhaps similarity score is just the probability of 'yes' answer)

Semantic search is too expensive for many documents.

It's best to only use it as an addition to the search engine, for example:

1. Find 10 best results with custom ML model
2. Use OpenAI semantic search to rerank the results

Semantic Search

Advantages

- very powerful semantic relatedness
- query can be composed of many words or sentences
- has knowledge about the world, can inferr things that aren't present in text

Disadvantages

- limited length of each document
- **(document + query < 2048 tokens)**
- no explainability
- can be very costful for large amount of documents
- can't constrain the knowledge of the GPT-3
- not clear what value of similarity score should we use as a threshold
- latency can be significant when many documents are analysed
- poor performance with text-2-text in my experience (good with word/phraze-2-text)

Text Classification

Why not simple prompt?

Prompt

```
This is a tweet sentiment classifier


Tweet: "I loved the new Batman movie!"
Sentiment: Positive
###
Tweet: "I hate it when my phone battery dies."
Sentiment: Negative
###
Tweet: "My day has been 👍"
Sentiment: Positive
###
Tweet: "This is the link to the article"
Sentiment: Neutral
###
Tweet: "This new music video blew my mind"
Sentiment:
```

Sample response

```
Positive
```

```python
openai.Classification.create(
  search_model="ada",
  model="curie",
  examples=[
    ["A happy moment", "Positive"],
    ["I am sad.", "Negative"],
    ["I am feeling awesome", "Positive"]
  ],
  query="It is a raining day :(",
  labels=["Positive", "Negative", "Neutral"],
)
```

```
Please classify a piece of text into the following categories: Positive, Negative, Neutral.

Text: I am sad.
Label: Negative
---
Text: A happy moment
Label: Positive
---
Text: I am feeling awesome
Label: Positive
---
Text: It is a raining day :(
Label:
```

examples

labels

query

Question Answering

“Facts are always based on assumptions, and so in a way, you’re getting GPT-3’s version of the facts which is really always going to be an opinion.”

Using simple prompt

It can often give us opinionated or simply wrong answer.

Davinci

What is DeFi?
DeFi is a term that stands for decentralized finance. It is a new type of financial system that is built on top of the blockchain.

Curie

What is DeFi?
DeFi is an acronym for decentralized finance. It's a term used to describe a movement that's looking to solve today's broken financial system and rebirth it from the ground up.

Question Answering Endpoint

First performs semantic search to find most relevant document.

Then combines provided query, examples and context into a prompt:

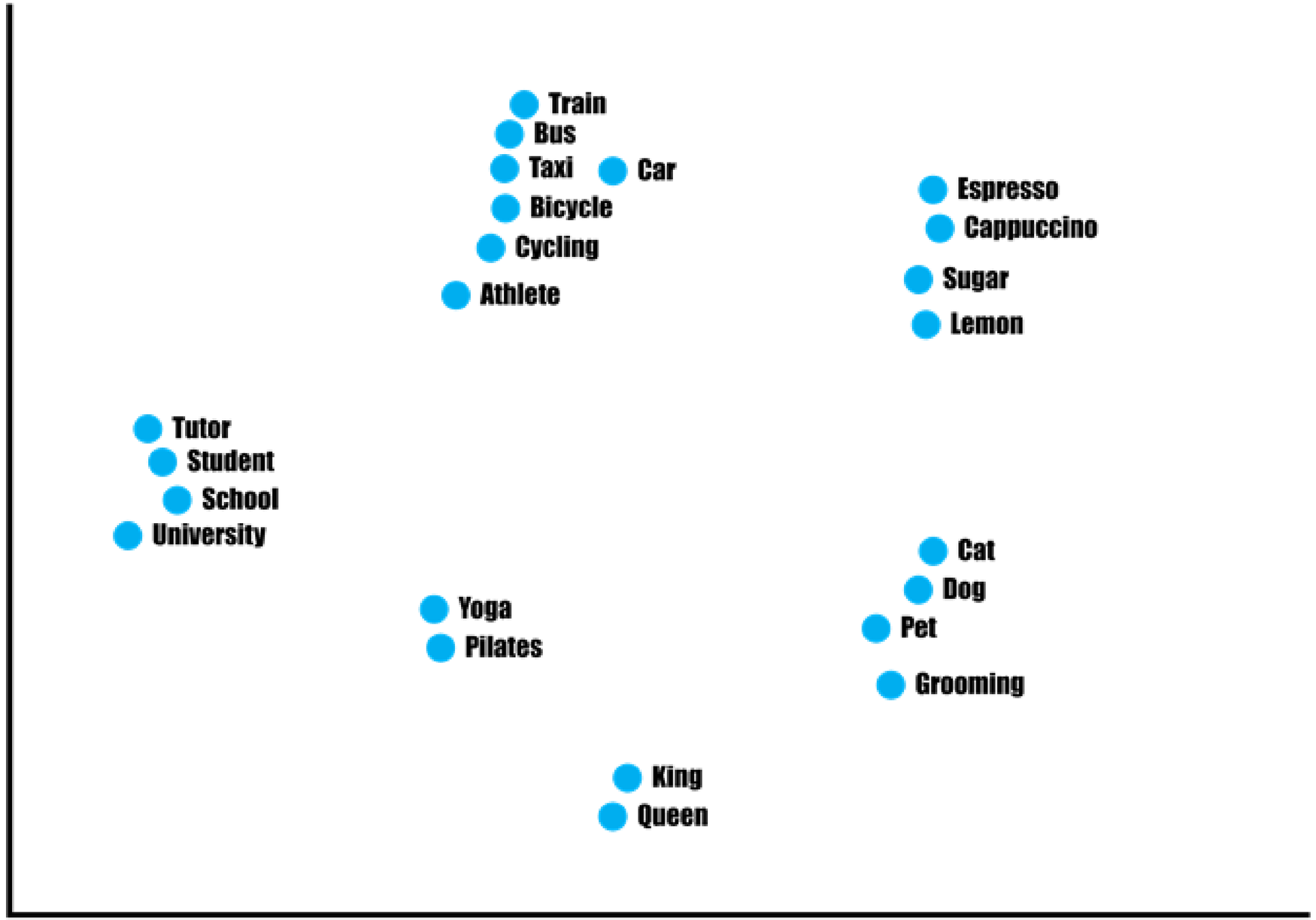
```
Please answer the question according to the above context.

===
Context: In 2017, U.S. life expectancy was 78.6 years.
===
Q: What is human life expectancy in the United States?
A: 78 years.
```

"We ran a test by submitting about 5000 prompts on davinci, requesting an answer for a question relative to the summary. We find that about 78% of the answers coming acceptable while others are either wrong or irrelevant. We use semantics search to filter out the irrelevant answers and still has about 8% of answers that are relevant but incorrect."

- some person on OpenAI forum

Embeddings

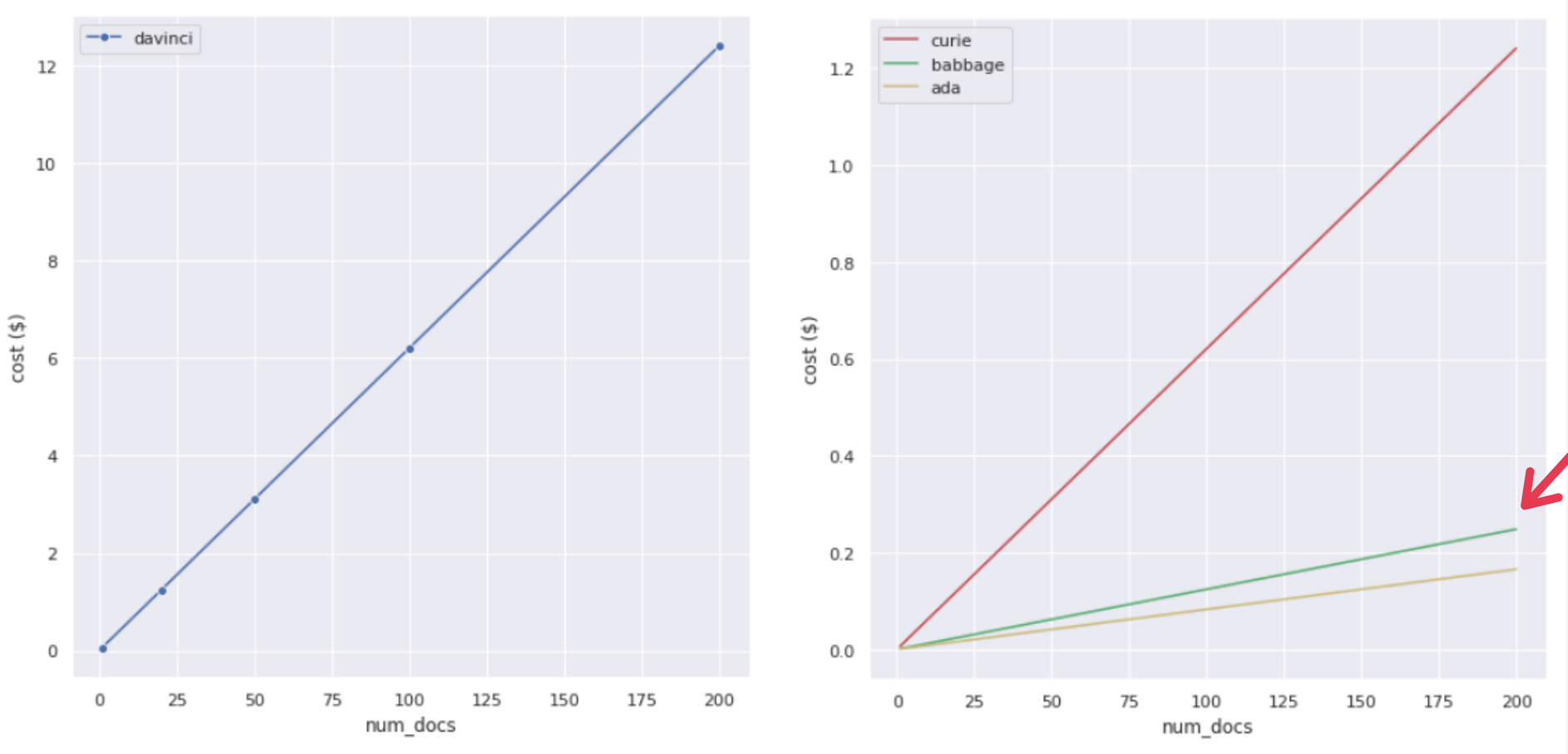

Train
Bus
Taxi
Car
Bicycle
Cycling
Athlete
Espresso
Cappuccino
Sugar
Lemon
Tutor
Student
School
University
Cat
Dog
Pet
Grooming
Yoga
Pilates
King
Queen

API Costs

MODEL		PRICE PER 1K TOKENS
Davinci	Most powerful	$0.0600
Curie		$0.0060
Babbage		$0.0012
Ada	Fastest	$0.0008

Comparison for semantic search endpoint

- *~750 words per each document*
- *query: 20 tokens*
- *extra 14 tokens per prompt used by the endpoint*



Babbage usually the best for semantic search

Text classification and question answering endpoints are based on semantic search!

Each document from our dataset is additional forward propagation through GPT-3!

Limitations of GPT-3

- Lacks long-term memory
- Limited input size: max 2048 tokens
- Slow inference time (model is large and predicts output word by word)
- Suffers from bias (it can be offensive towards races, religions, etc.)
- Lack of interpretability
- You can't constrain the knowledge of GPT-3

Code Generation

CODEX & GITHUB COPILOT

What is **Codex**?

Basically GPT-3, but the model is smaller and trained on code.

https://beta.openai.com/codex-waitlist

- Same architecture as GPT
- Currently biggest version has 12 billions parameters (GPT-3 has 175 billions)
- First trained on internet text (just like GPT)
- After being trained on text, it was additionaly trained on github code
- Has more rich vocabulary which encodes whiespaces better
- Currently supports 12 programming languages

Prompt

```
# Python 3
def remove_common_prefix(x, prefix, ws_prefix):
    x["completion"] = x["completion"].str[len(prefix) :]
    if ws_prefix:
        # keep the single whitespace as prefix
        x["completion"] = " " + x["completion"]
return x

# Explanation of what the code does

#
```

Sample response

The code above is a function that takes a dataframe and a prefix as input and returns a dataframe with the prefix removed from the completion column.

Prompt

Create a SQL request to find all users who live in California and have over 1000 credits:

SELECT

Sample response

```
* FROM users WHERE state = 'CA' AND credits > 1000
```

Natural language to OpenAI API
Create code to call to the OpenAI API usin...

Natural language to Stripe API
Create code to call the Stripe API using nat...

SQL translate
Translate natural language to SQL queries.

Python to natural language
Explain a piece of Python code in human un...

Calculate Time Complexity
Find the time complexity of a function.

Translate programming languages
Translate from one programming language ...

Explain code
Explain a complicated piece of code.

Python bug fixer
Find and fix bugs in source code.

JavaScript helper chatbot
Message-style bot that answers JavaScript ...

JavaScript to Python
Convert simple JavaScript expressions into ...

Write a Python docstring
An example of how to create a docstring for ...

JavaScript one line function
Turn a JavaScript function into a one liner.

Demo Time

https://beta.openai.com/codex-javascript-sandbox

GitHub Copilot

https://copilot.github.com

When are Codex & Copilot useful?

- Basic boilerplate code
- Writing tests
- Using standard libraries
- Writing simple things which there are lots of examples on the internet (e.g. there's lots of examples of 2D games in javascript on github, so it's not suprising the model is good at it)
- It's good at boring parts of the code! (like creating all SQL statements for a particular datamodel or getters and setters in java)
- Terminal commands in natural language

Open Source Alternatives

GPT-J

GPT-J

Can be run on high-end consumer GPU.
Requires around 50GB RAM and 20GB GPU memory.
You can download it from github or through HuggingFace library.

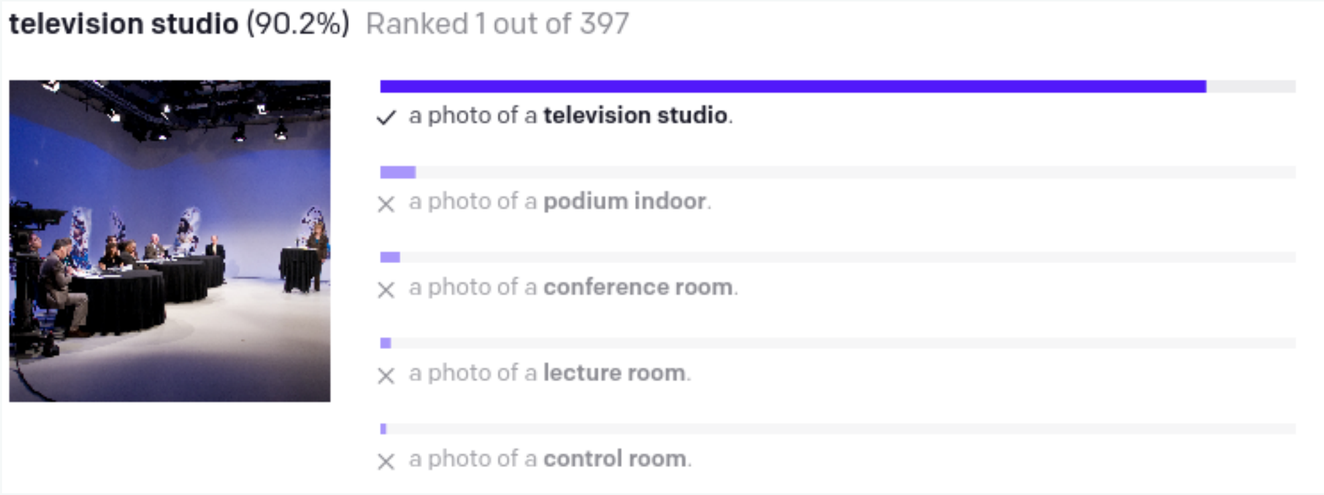
Future

GPT-4

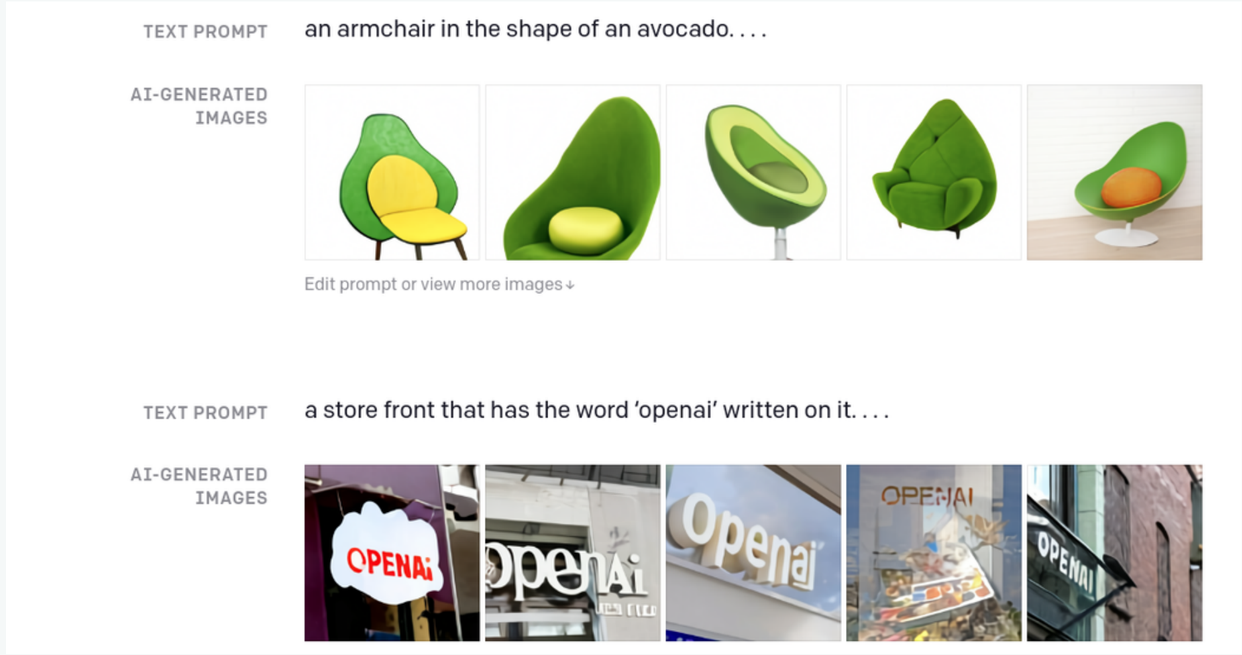
- *Will be a text model (as opposed to multi-modal)*
- *It will not be much bigger than GPT-3, but it will use way more compute*
- *GPT-4 will likely be able to work with longer context*

Image2text & Text2image

CLIP


television studio (90.2%) Ranked 1 out of 397
a photo of a television studio.
a photo of a podium indoor.
a photo of a conference room.
a photo of a lecture room.
a photo of a control room.

DALL-E


TEXT PROMPT
an armchair in the shape of an avocado. . . .
AI-GENERATED IMAGES
Edit prompt or view more images
TEXT PROMPT
a store front that has the word 'openai' written on it. . . .
AI-GENERATED IMAGES
OPENAI
OPENAI
OPENAI
OPENAI

Software 1.0 - writing code that does things for us

Software 2.0 - training models that do things for us

Software 3.0 - querying large models to do things for us