Transformers

Rina BUOY

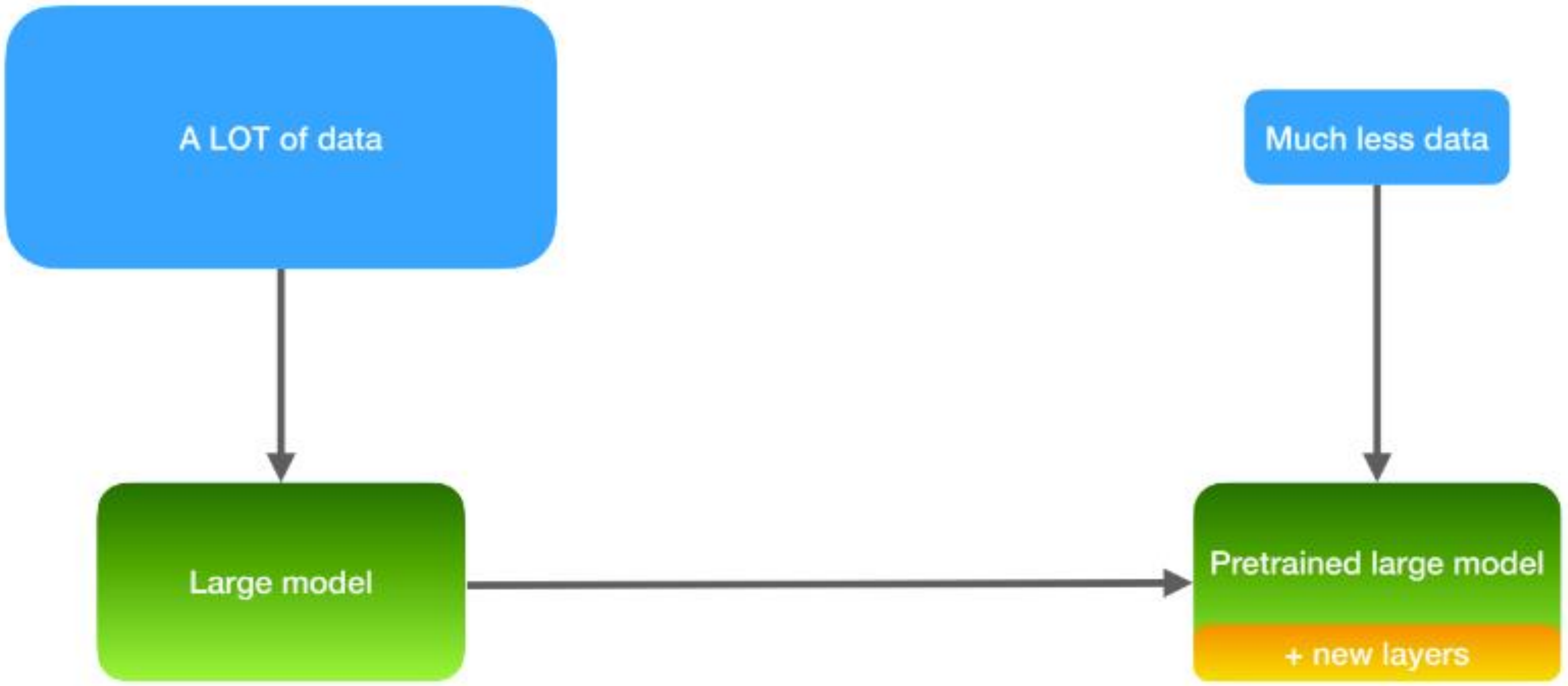
Source

AMERICAN UNIVERSITY
OF PHNOM PENH

STUDY LOCALLY. LIVE GLOBALLY.

Fine-tuning

Transfer Learning



A LOT of data

Large model

Much less data

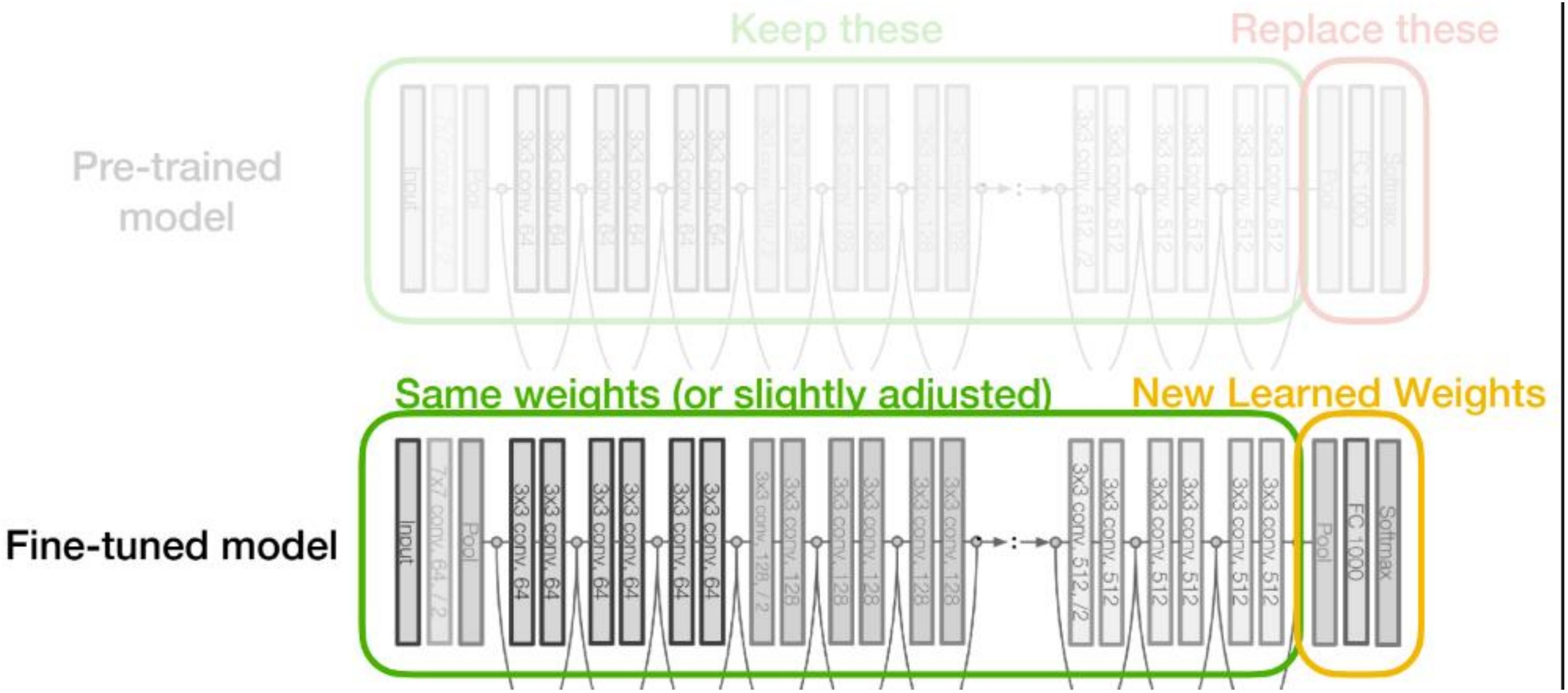
Pretrained large model

+ new layers

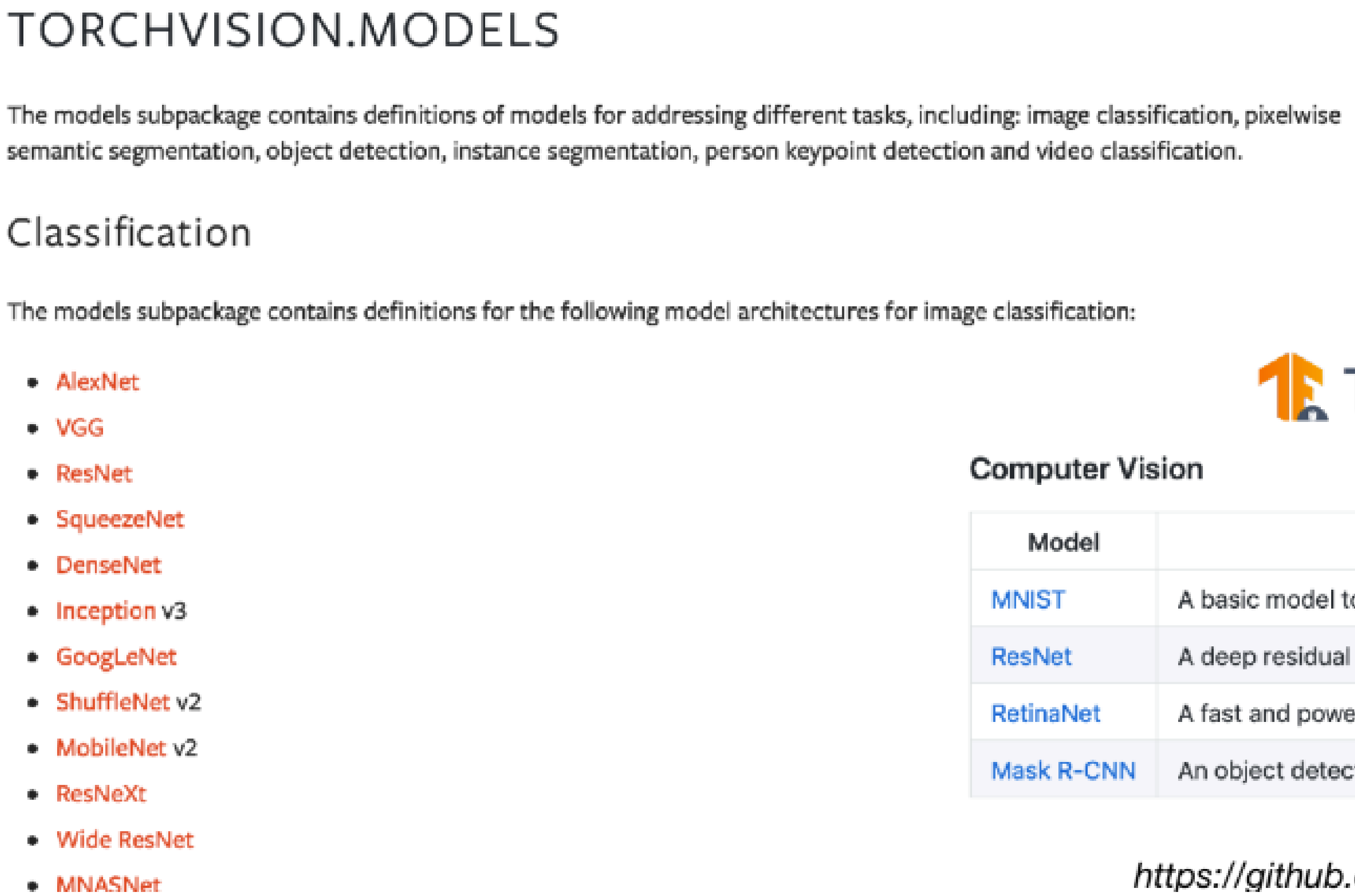
Traditional Machine Learning:
slow training on a lot of data

Transfer learning:
fast training on a little data

Transfer Learning



Model Zoos



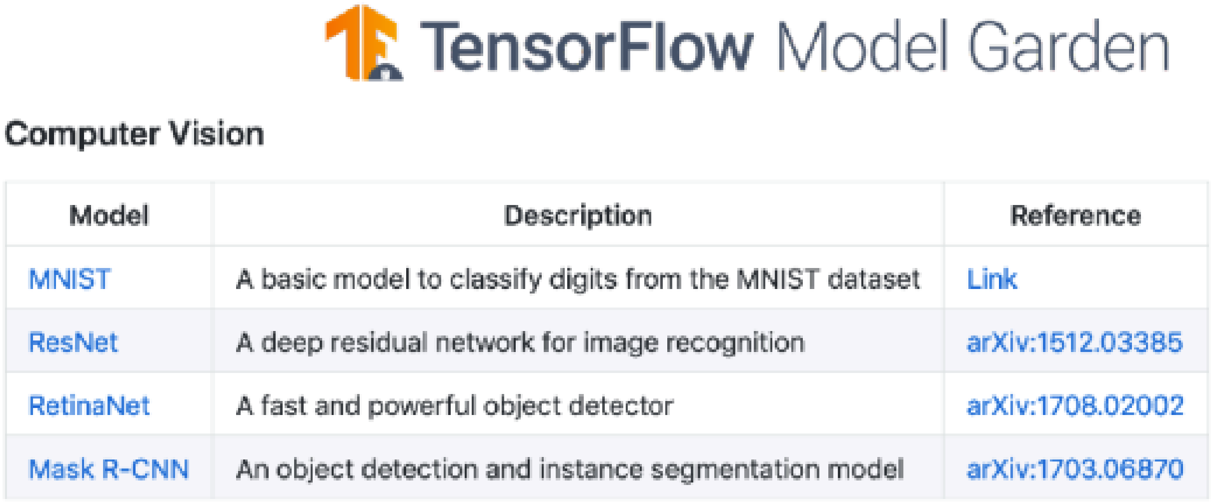
TORCHVISION.MODELS

The models subpackage contains definitions of models for addressing different tasks, including: image classification, pixelwise semantic segmentation, object detection, instance segmentation, person keypoint detection and video classification.

Classification

The models subpackage contains definitions for the following model architectures for image classification:

- AlexNet
- VGG
- ResNet
- SqueezeNet
- DenseNet
- Inception v3
- GoogLeNet
- ShuffleNet v2
- MobileNet v2
- ResNeXt
- Wide ResNet
- MNASNet



TensorFlow Model Garden

Computer Vision

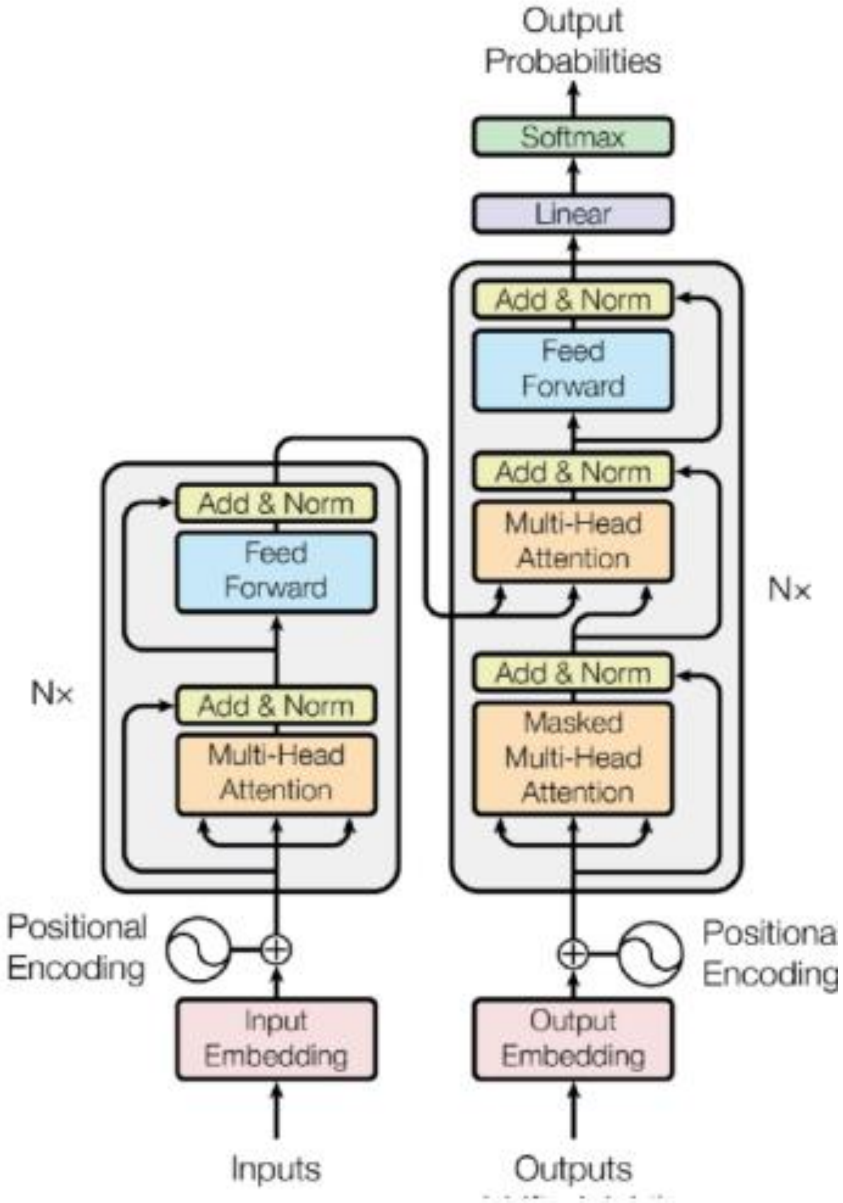
Model	Description	Reference
MNIST	A basic model to classify digits from the MNIST dataset	Link
ResNet	A deep residual network for image recognition	arXiv:1512.03385
RetinaNet	A fast and powerful object detector	arXiv:1708.02002
Mask R-CNN	An object detection and instance segmentation model	arXiv:1703.06870

https://github.com/tensorflow/models/tree/master/official

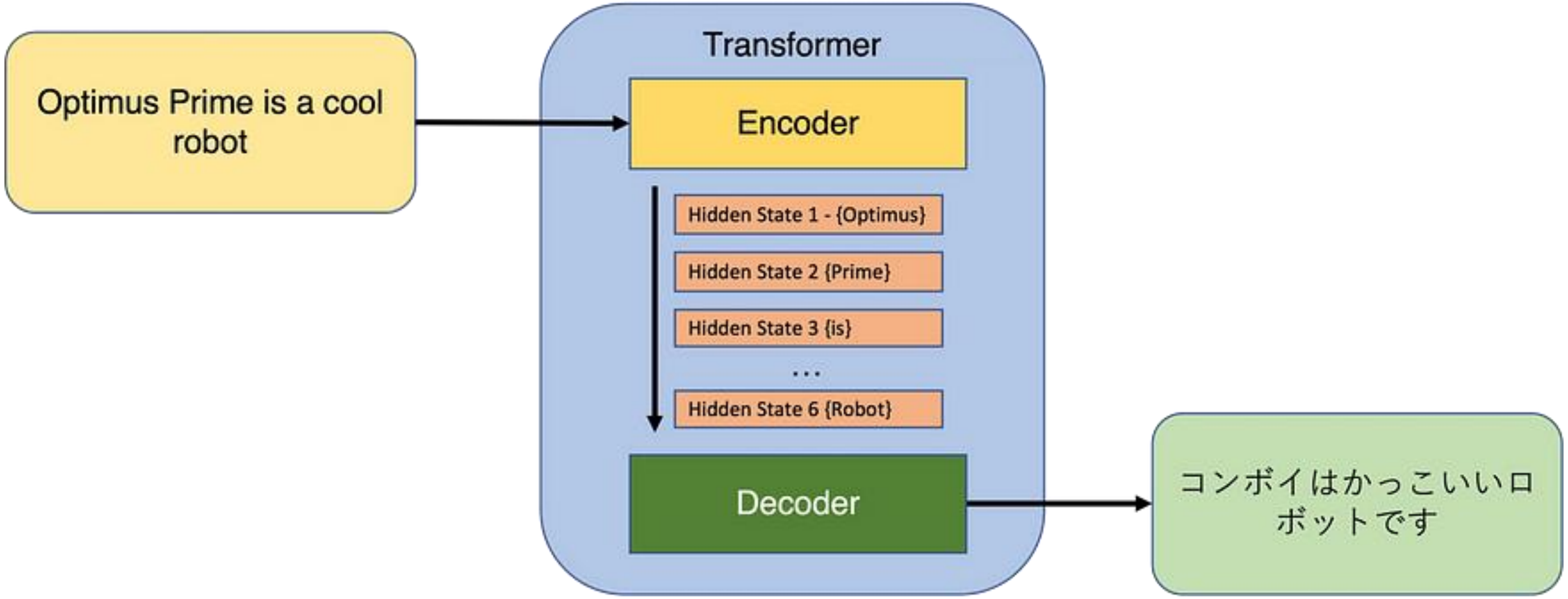
Transformers

Transformers

- Ground-breaking architecture that set SOTA on first translation and later all other NLP and CV tasks.
- Attention is all you need (2017)



Encoder-Decoder Transformers

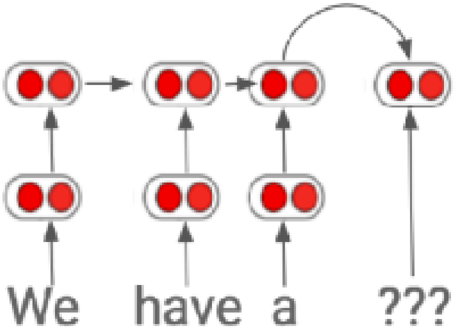


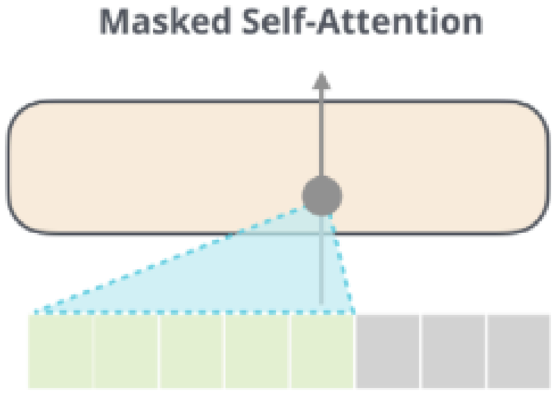
https://blog.gopenai.com/the-transformer-architecture-a-comprehensive-exploration-with-examples-b8c55b0e72e0

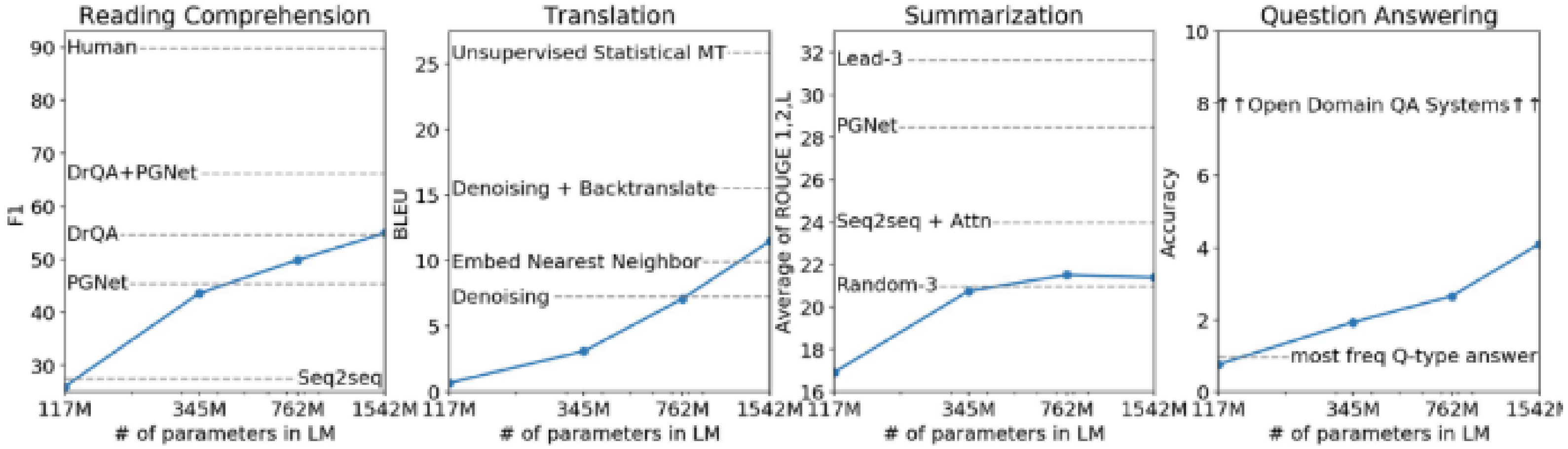
Large Language Models

GPT / GPT-2 (2019)

- Generative Pre-trained Transformer
- Decoder-only (uses masked self-attention)
- Trained on 8M web pages, largest model is 1.5B



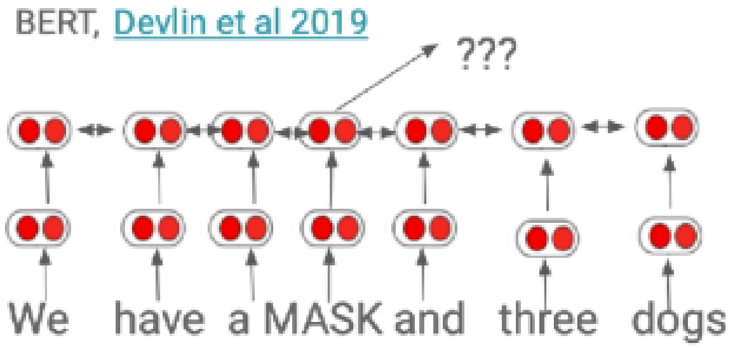




https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

BERT (2019)

- *Bidirectional* Encoder Representations from Transformers
- Encoder-only (no attention masking)
- 110M params
- 15% of all words masked out

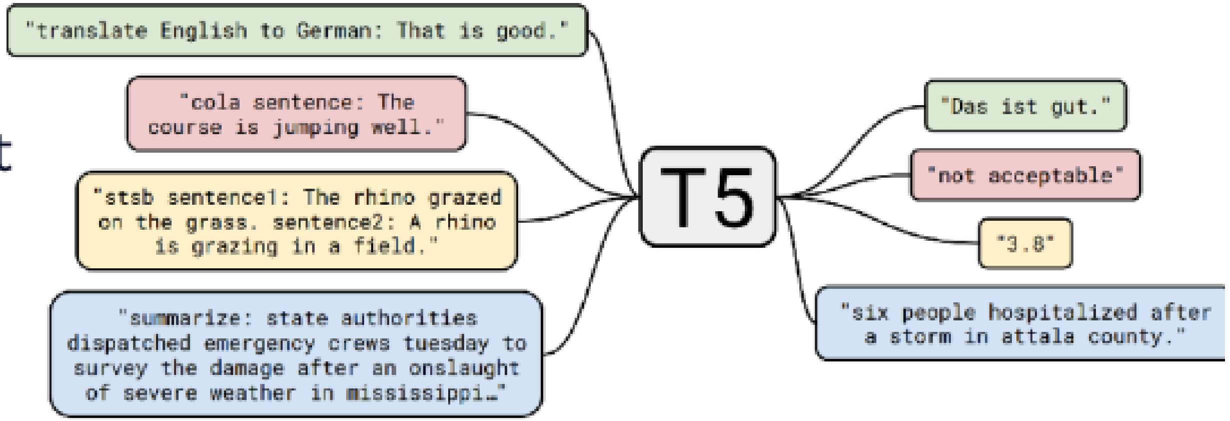


BERT, Devlin et al 2019

https://docs.google.com/presentation/d/1fIhGikFPnb7G5kr58OvYC3GN4io7MznnM0aAgadvJfc

T5: Text-to-Text Transfer Transformer (2020)

- Input and output are both text strings
- Encoder-Decoder architecture
- Trained on C4 (Colossal Clean Crawled Corpus) - 100x larger than Wikipedia
- 11B parameters, open source!



"translate English to German: That is good."

"cola sentence: The course is jumping well."

"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."

"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"

T5

"Das ist gut."

"not acceptable"

"3.8"

"six people hospitalized after a storm in attala county."

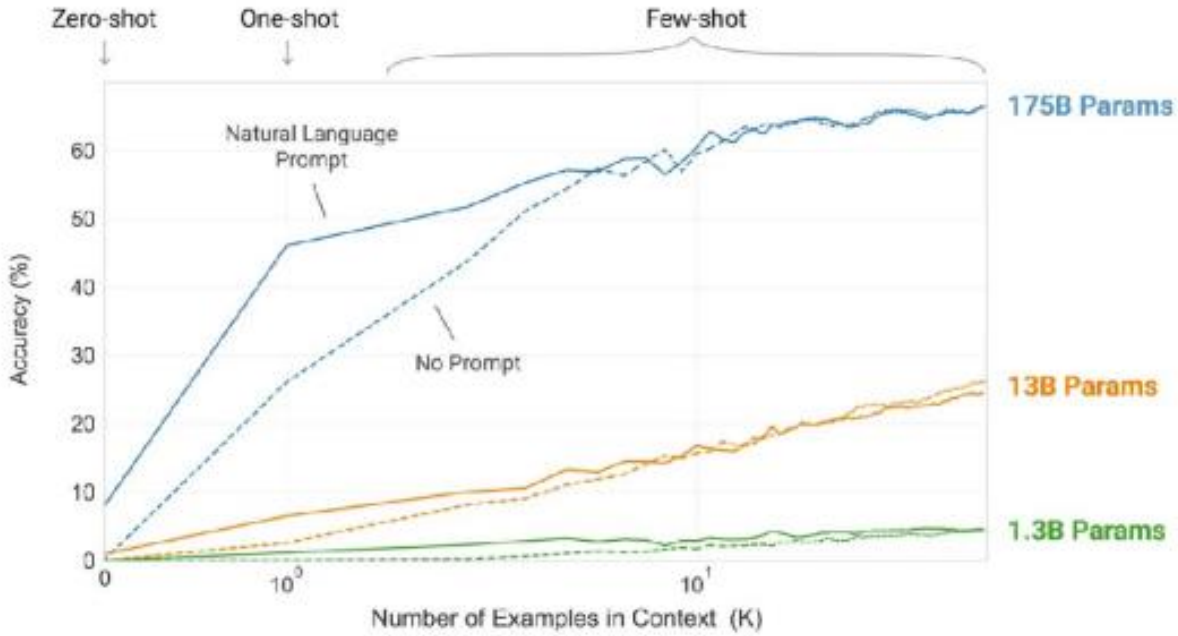
t5-11b

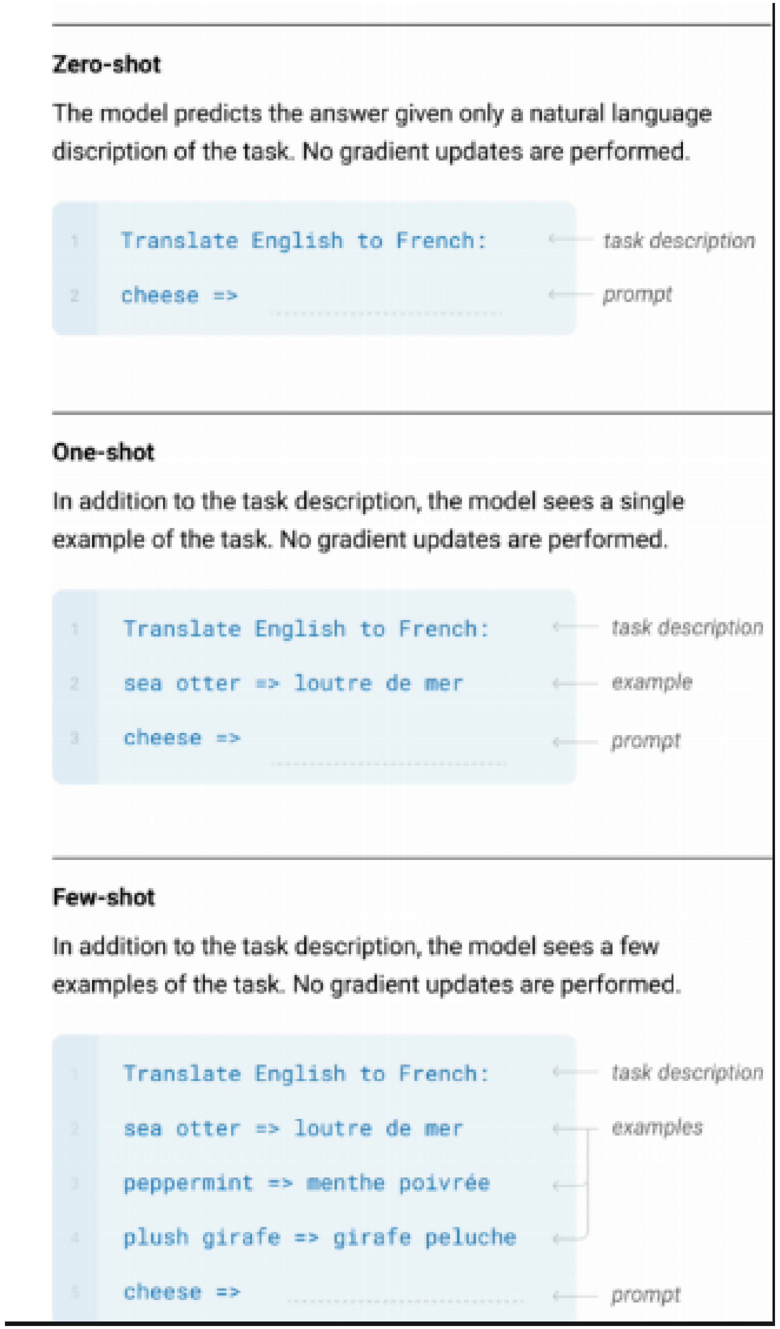
Model Card for T5 11B

https://ai.googleblog.com/2020/02/exploring-transfer-learning-with-t5.html

GPT-3 (2020)

- Just like GPT/GPT-2, but 100x larger (175B params)
- Exhibits unprecedented few-shot and zero-shot learning
- Not yet overfitting!
- Available via API





Zero-shot

The model predicts the answer given only a natural language discription of the task. No gradient updates are performed.

```
Translate English to French:
cheese =>
```

task description
prompt

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
Translate English to French:
sea otter => loutre de mer
cheese =>
```

task description
example
prompt

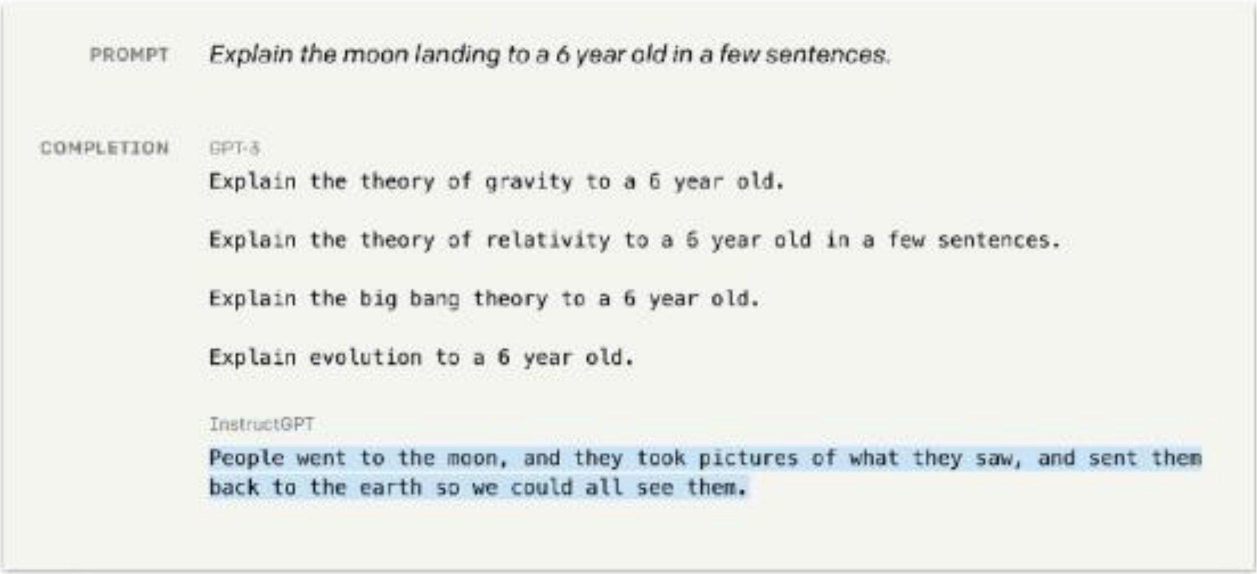
Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>
```

Instruct-GPT (2022)

- Had humans rank different GPT-3 outputs, and used RL to fine-tune the model
- **Much** better at following instructions
- `text-davinci-002` in the API
  - Unclear how big the model is. Could be ~10x smaller than 175B?



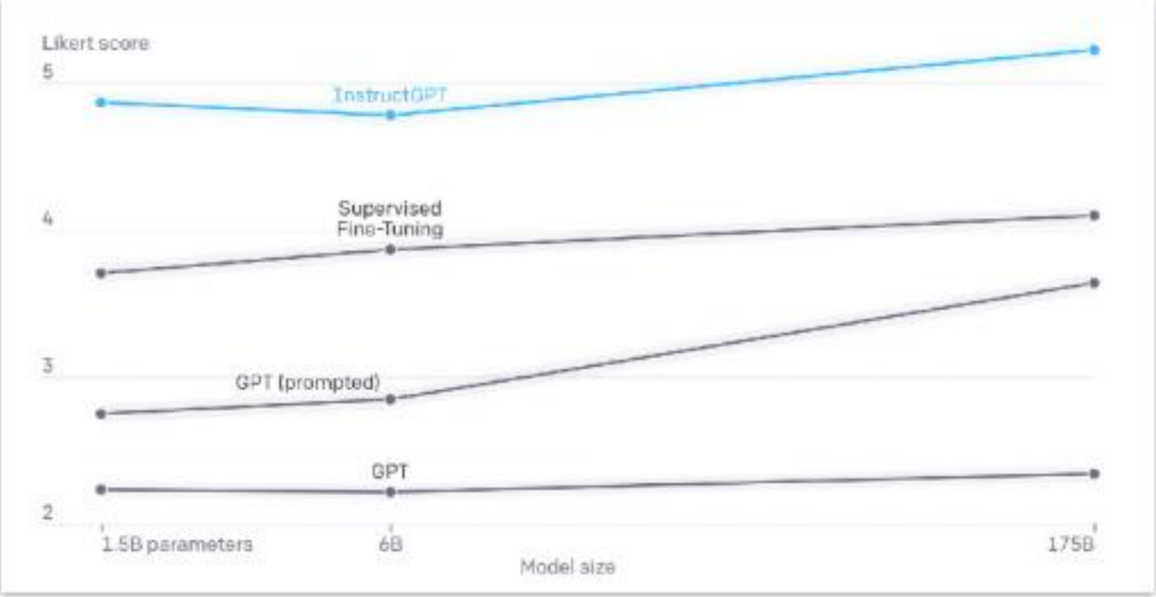
PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

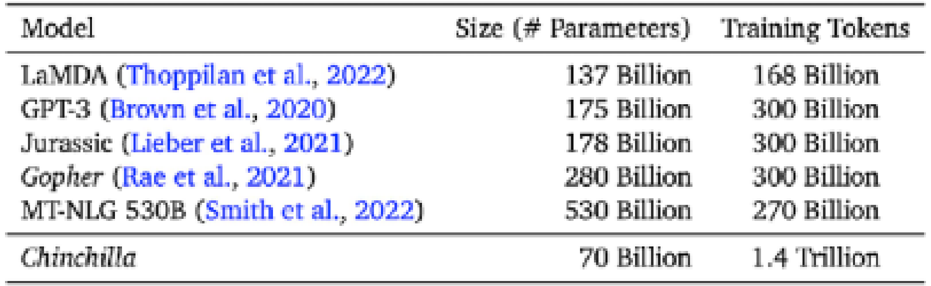
InstructGPT

```
People went to the moon, and they took pictures of what they saw, and sent them
back to the earth so we could all see them.
```

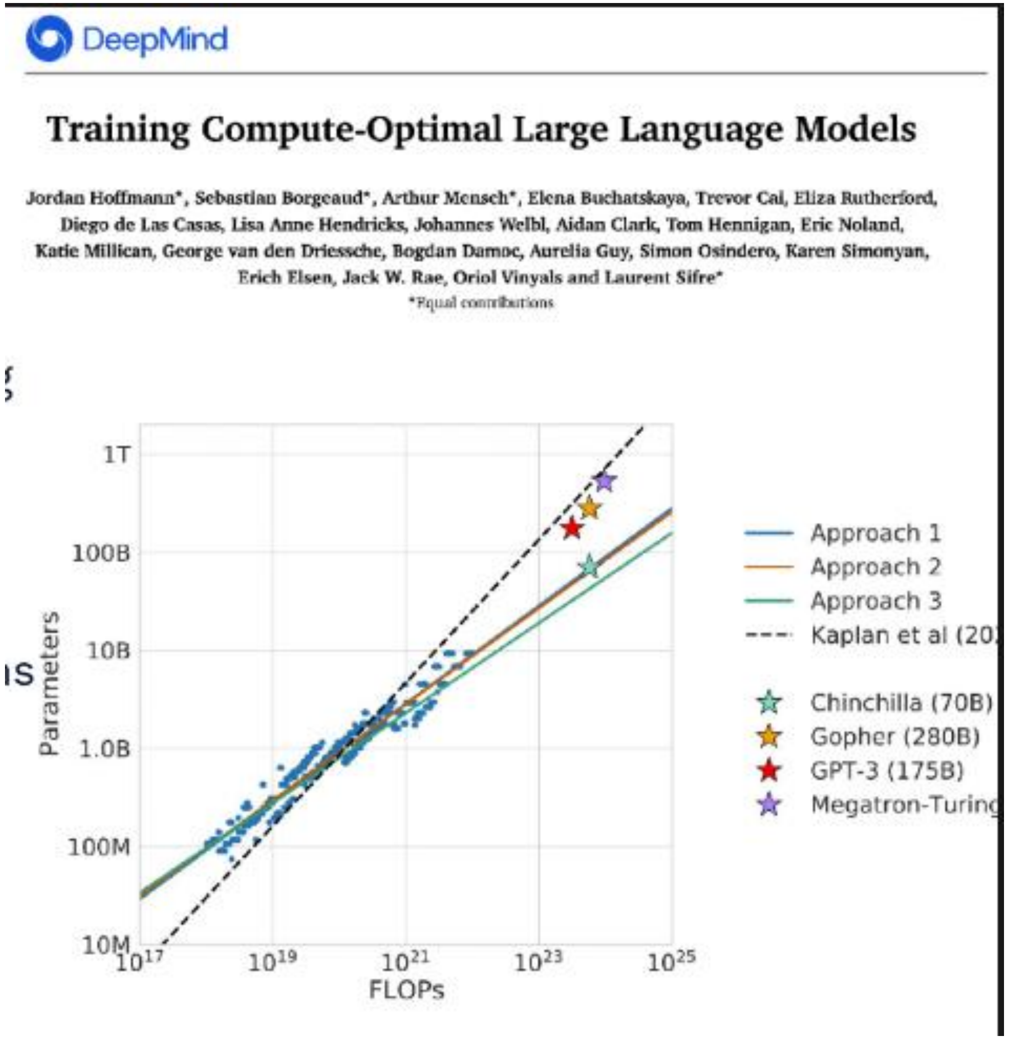


Chinchilla (2022)

- Trained over 400 LM's from 70M to 16B params on 5B to 500B tokens
- Derived formulas for optimal model and training set size given a fixed compute budget
- Found that most LLMs are "undertrained"
- Trained Chinchilla (70B) vs Gopher (280B) at the same compute budget, by using 4x fewer params and 4x more data



Model	Size (# Parameters)	Training Tokens
LaMDA (Thoppilan et al., 2022)	137 Billion	168 Billion
GPT-3 (Brown et al., 2020)	175 Billion	300 Billion
Jurassic (Lieber et al., 2021)	178 Billion	300 Billion
Gopher (Rae et al., 2021)	280 Billion	300 Billion
MT-NLG 530B (Smith et al., 2022)	530 Billion	270 Billion
Chinchilla	70 Billion	1.4 Trillion



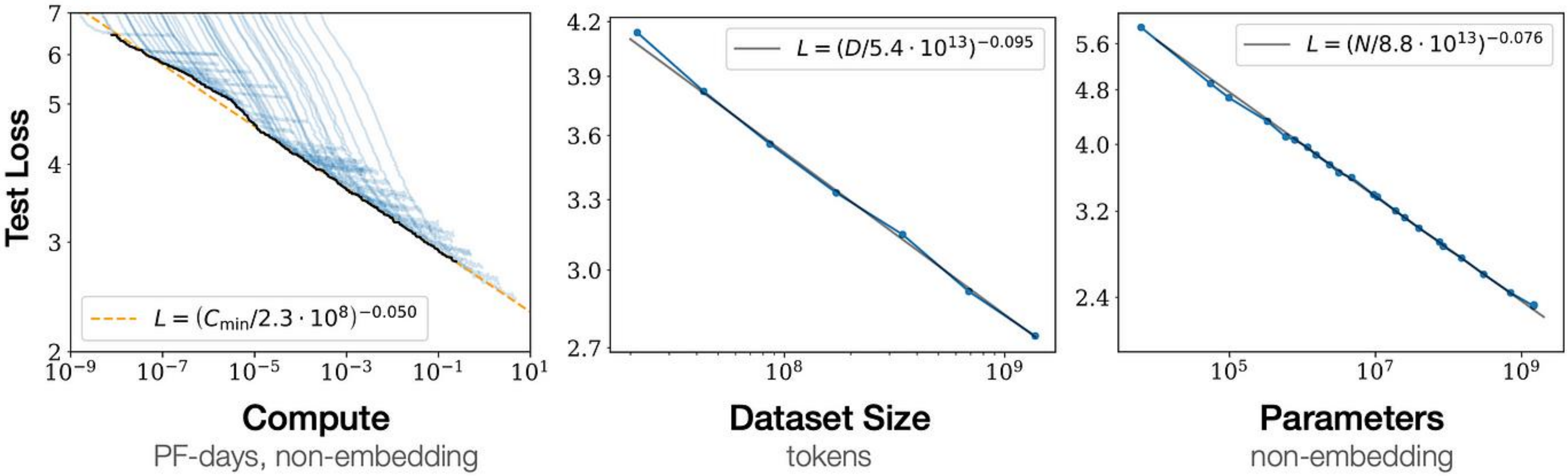
DeepMind

Training Compute-Optimal Large Language Models

Jordan Hoffmann*, Sebastian Borgeaud*, Arthur Mensch*, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals and Laurent Sifre*

*Equal contributions

Scaling Law

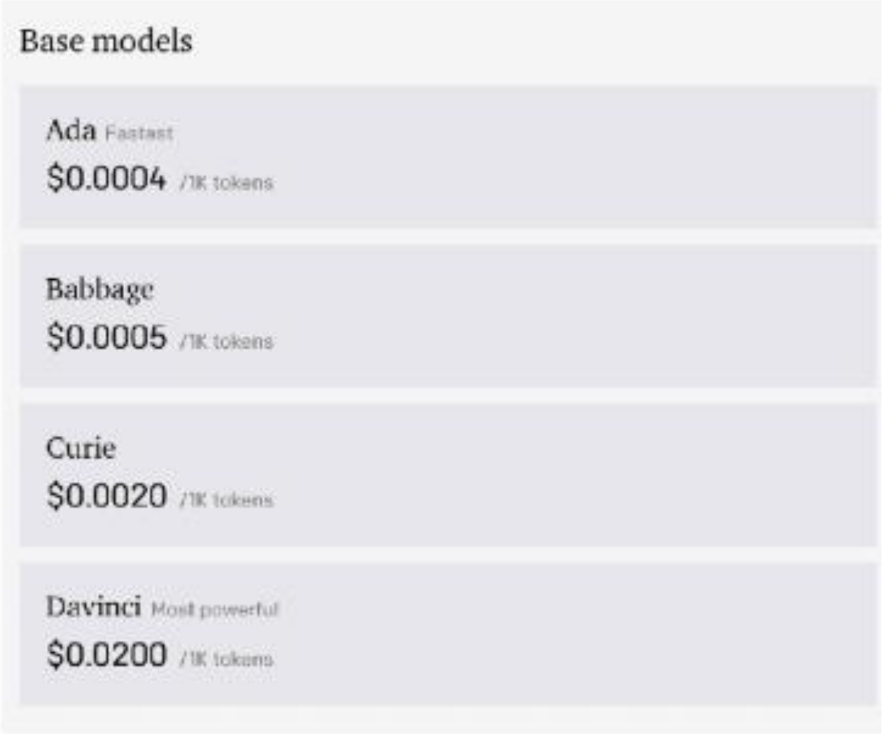

Test Loss
7
6
5
4
3
2
10^{-9}
10^{-7}
10^{-5}
10^{-3}
10^{-1}
10^{1}
$L = (C_{\min}/2.3 \cdot 10^{8})^{-0.050}$
Compute
PF-days, non-embedding
4.2
3.9
3.6
3.3
3.0
2.7
10^{8}
10^{9}
$L = (D/5.4 \cdot 10^{13})^{-0.095}$
Dataset Size
tokens
5.6
4.8
4.0
3.2
2.4
10^{5}
10^{7}
10^{9}
$L = (N/8.8 \cdot 10^{13})^{-0.076}$
Parameters
non-embedding

LLM Vendors

LLM Vendor: OpenAI

- Four model sizes
  - Most GPT-3 results you've seen are from Davinci
  - Probably 350M, 1.3B, 6.7B, and 175B

    https://blog.eleuther.ai/gpt3-model-sizes/
  - 1000 tokens ~= 750 words
  - I have ~800 tweets, about 40K words, so it would cost ~$1 to process all my tweets
- Ability to fine-tune models (for extra cost)
- Quota is pretty small to start, over time can raise
- Apply for review before going into production



Base models

Ada Fastest
$0.0004 /1K tokens

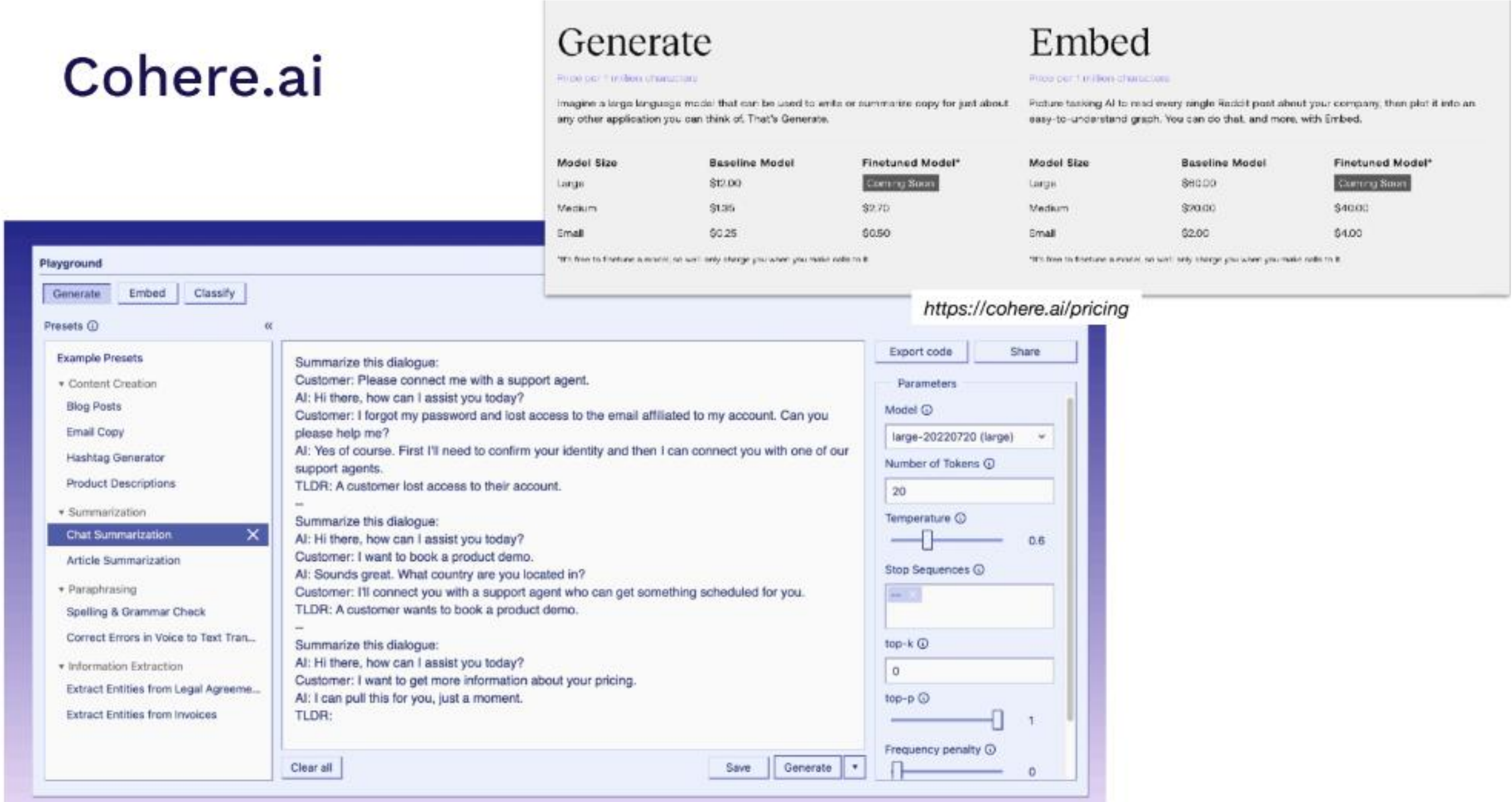
Babbage
$0.0005 /1K tokens

Curie
$0.0020 /1K tokens

Davinci Most powerful
$0.0200 /1K tokens

Cohere.ai

Cohere.ai



Generate

Price per 1 million characters

Imagine a large language model that can be used to write or summarize copy for just about any other application you can think of. That's Generate.

Model Size	Baseline Model	Finetuned Model*
Large	$12.00	Coming Soon
Medium	$1.35	$2.70
Small	$0.25	$0.50

*It's free to finetune a model, so we'll only charge you when you make calls to it

Embed

Price per 1 million characters

Picture tasking AI to read every single Reddit post about your company, then plot it into an easy-to-understand graph. You can do that, and more, with Embed.

Model Size	Baseline Model	Finetuned Model*
Large	$80.00	Coming Soon
Medium	$20.00	$40.00
Small	$2.00	$4.00

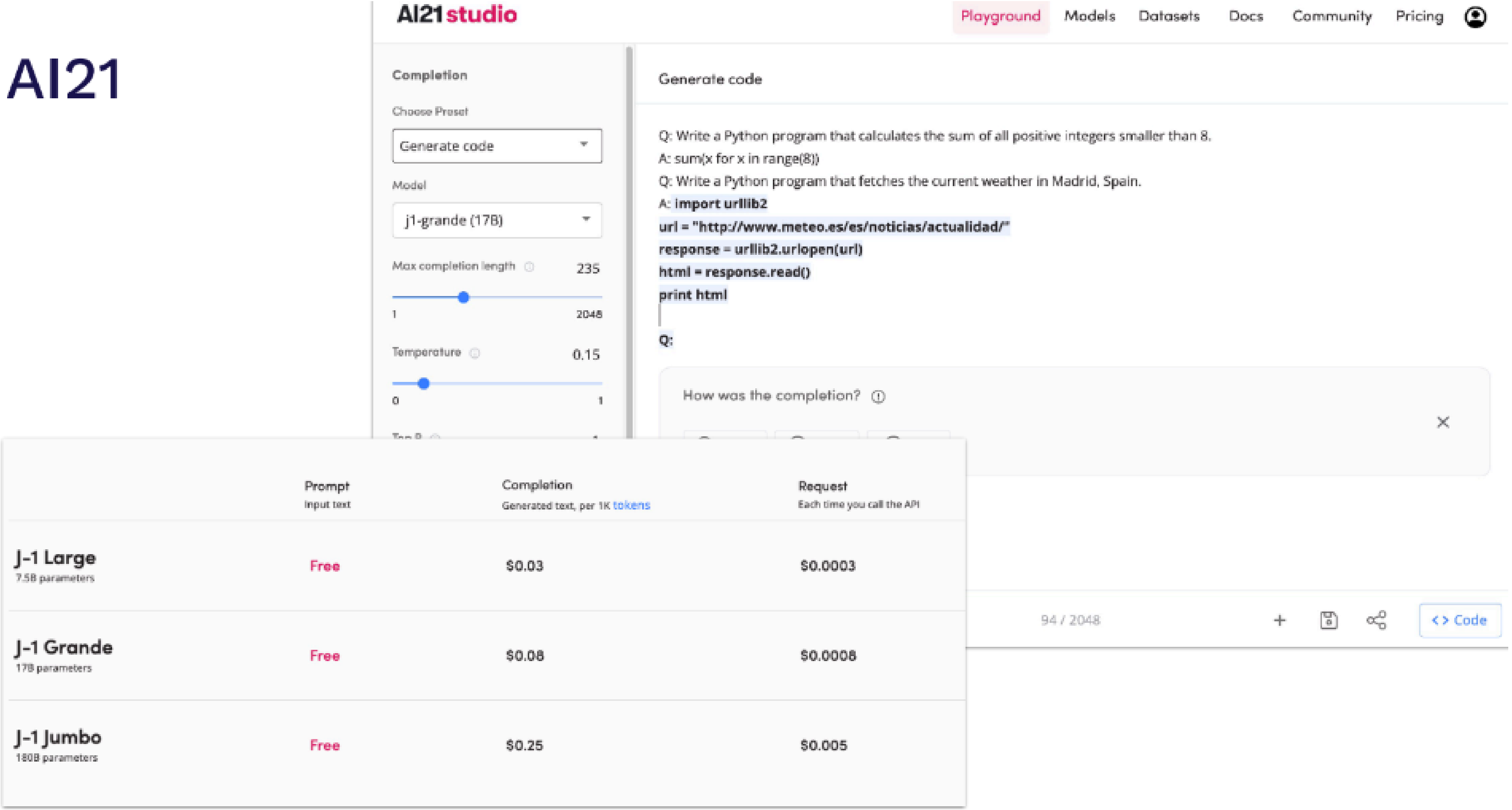
*It's free to finetune a model, so we'll only charge you when you make calls to it

https://cohere.ai/pricing

Summarize this dialogue:
Customer: Please connect me with a support agent.
AI: Hi there, how can I assist you today?
Customer: I forgot my password and lost access to the email affiliated to my account. Can you please help me?
AI: Yes of course. First I'll need to confirm your identity and then I can connect you with one of our support agents.
TLDR: A customer lost access to their account.
--
Summarize this dialogue:
AI: Hi there, how can I assist you today?
Customer: I want to book a product demo.
AI: Sounds great. What country are you located in?
Customer: I'll connect you with a support agent who can get something scheduled for you.
TLDR: A customer wants to book a product demo.
--
Summarize this dialogue:
AI: Hi there, how can I assist you today?
Customer: I want to get more information about your pricing.
AI: I can pull this for you, just a moment.
TLDR:

Parameters
Model: large-20220720 (large)
Number of Tokens: 20
Temperature: 0.6
Stop Sequences: --
top-k: 0
top-p: 1
Frequency penalty: 0

AI21



Completion

Choose Preset: Generate code

Model: j1-grande (17B)

Max completion length: 235

Temperature: 0.15

Generate code

Q: Write a Python program that calculates the sum of all positive integers smaller than 8.
A: sum(x for x in range(8))
Q: Write a Python program that fetches the current weather in Madrid, Spain.
A: **import urllib2**

```
url = "http://www.meteo.es/es/noticias/actualidad/"
response = urllib2.urlopen(url)
html = response.read()
print html
```

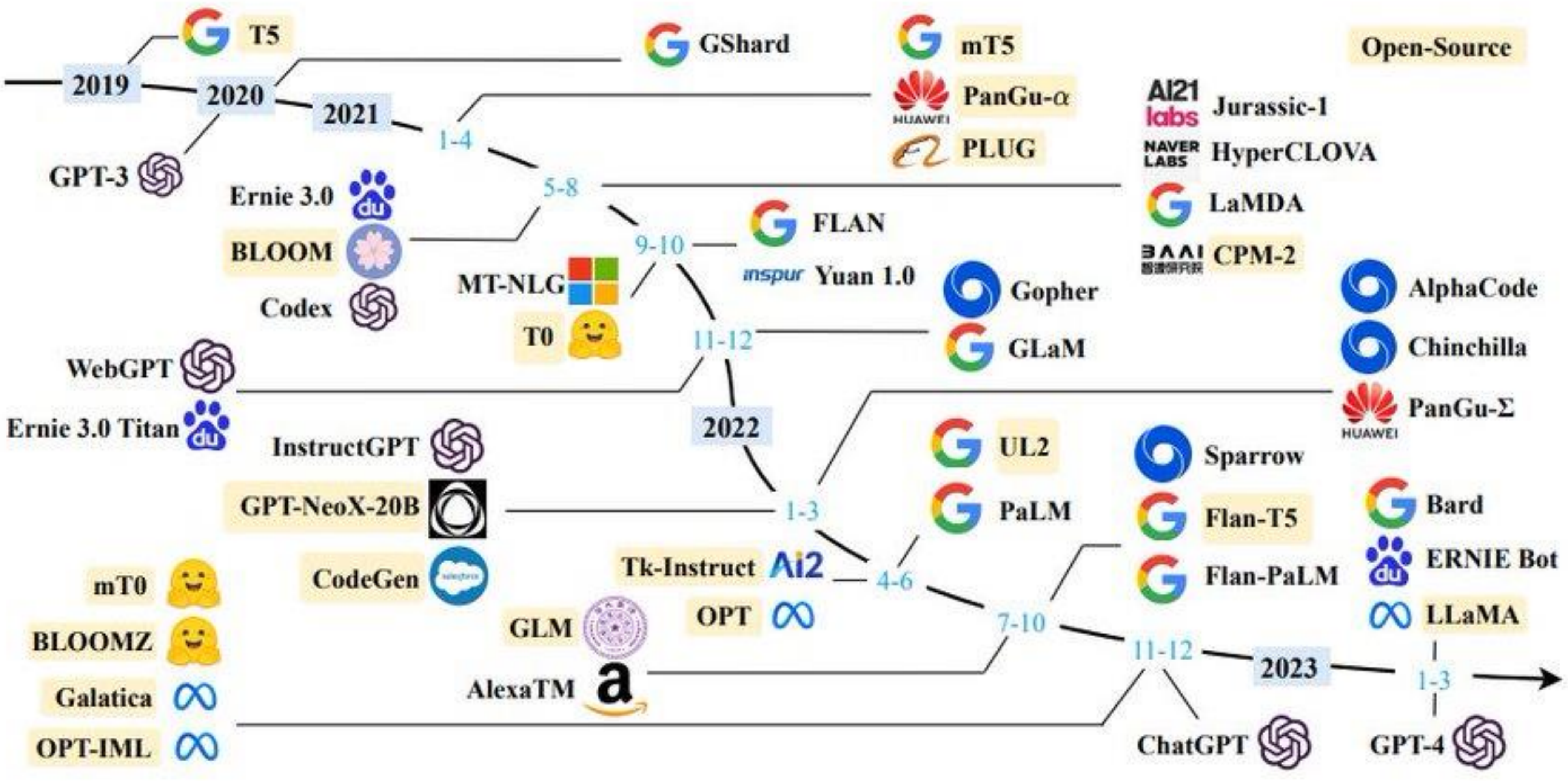
Q:

How was the completion?

94 / 2048

	Prompt Input text	Completion Generated text, per 1K tokens	Request Each time you call the API
J-1 Large 7.5B parameters	Free	$0.03	$0.0003
J-1 Grande 17B parameters	Free	$0.08	$0.0008
J-1 Jumbo 180B parameters	Free	$0.25	$0.005

Open-source LLMs



https://x.com/DrJimFan/status/1669025412138299392

Prompt Engineering

Alien Technology

- The most recent version of GPT-3 (text-davinci-002, based on InstructGPT) is alien technology
- People are finding out how it works by playing with it
- Here we cover some notable examples
- Play around and you're likely to discover something new!

Riley Goodside

@goodside Follows you

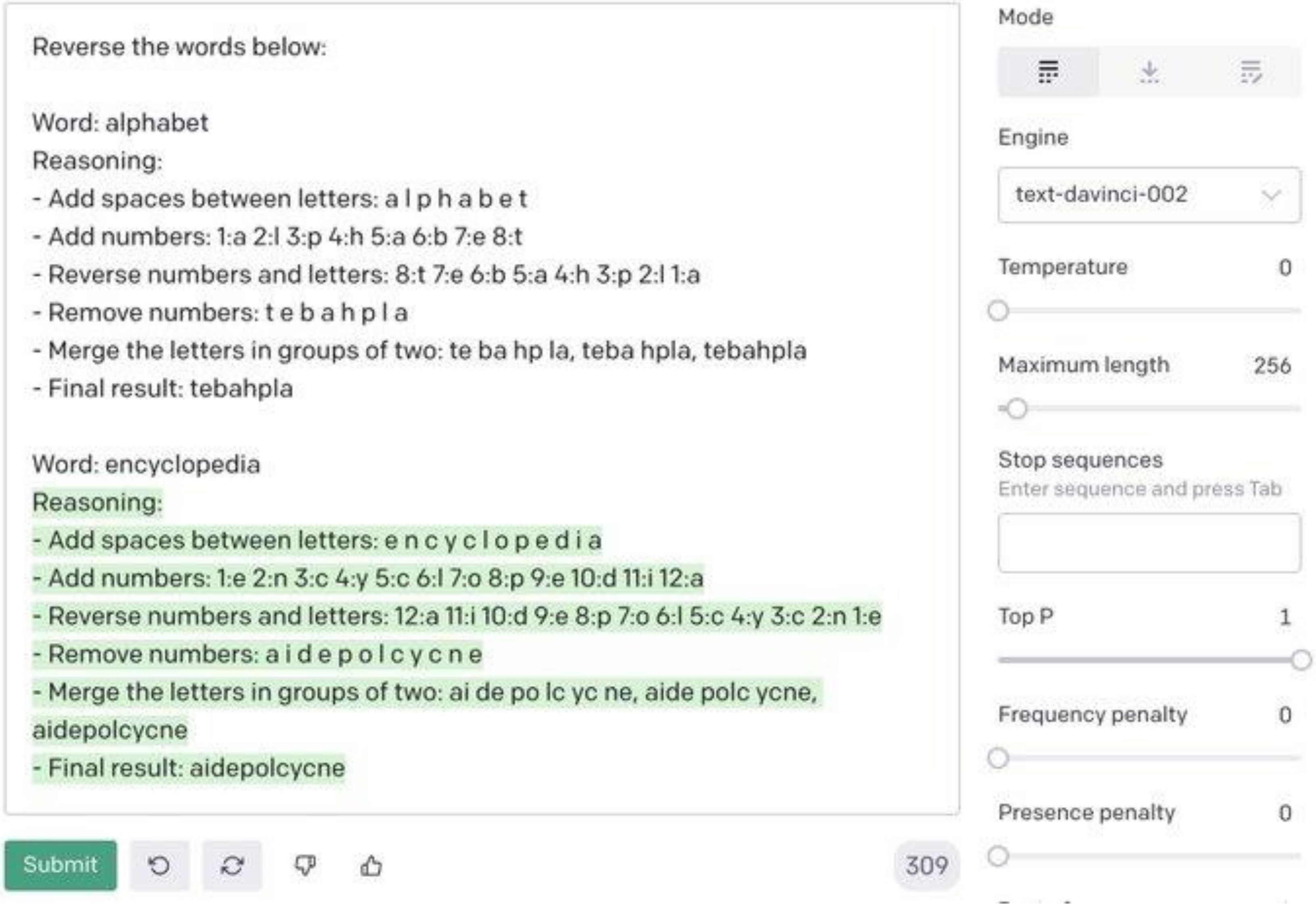
Data Scientist @copy_ai, previously @Grindr, @OkCupid. I'm good at talking to GPT-3.

https://twitter.com/goodside

Prompt Engineering

- GPT-3 is surprisingly bad at reversing words due to **tokenization**:
- It doesn’t see letters and words as humans do.
- Instead, it sees “tokens,” which are chunks of characters.
- Furthermore, it gets confused with long-ish sequences.
- Finally, it has trouble merging characters.
- For it to work, you have to teach GPT-3 the algorithm to use to get around its limitations.

Prompt Engineering



Reverse the words below:

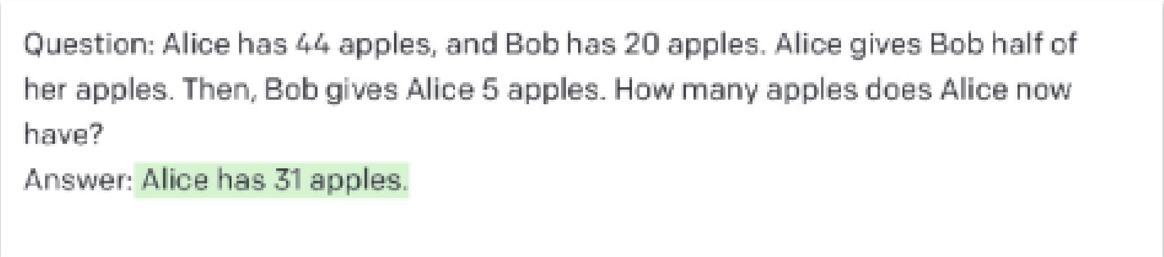
Word: alphabet
Reasoning:
- Add spaces between letters: a l p h a b e t
- Add numbers: 1:a 2:l 3:p 4:h 5:a 6:b 7:e 8:t
- Reverse numbers and letters: 8:t 7:e 6:b 5:a 4:h 3:p 2:l 1:a
- Remove numbers: t e b a h p l a
- Merge the letters in groups of two: te ba hp la, teba hpla, tebahpla
- Final result: tebahpla

Word: encyclopedia
Reasoning:
- Add spaces between letters: e n c y c l o p e d i a
- Add numbers: 1:e 2:n 3:c 4:y 5:c 6:l 7:o 8:p 9:e 10:d 11:i 12:a
- Reverse numbers and letters: 12:a 11:i 10:d 9:e 8:p 7:o 6:l 5:c 4:y 3:c 2:n 1:e
- Remove numbers: a i d e p o l c y c n e
- Merge the letters in groups of two: ai de po lc yc ne, aide polc ycne, aidepolcycne
- Final result: aidepolcycne

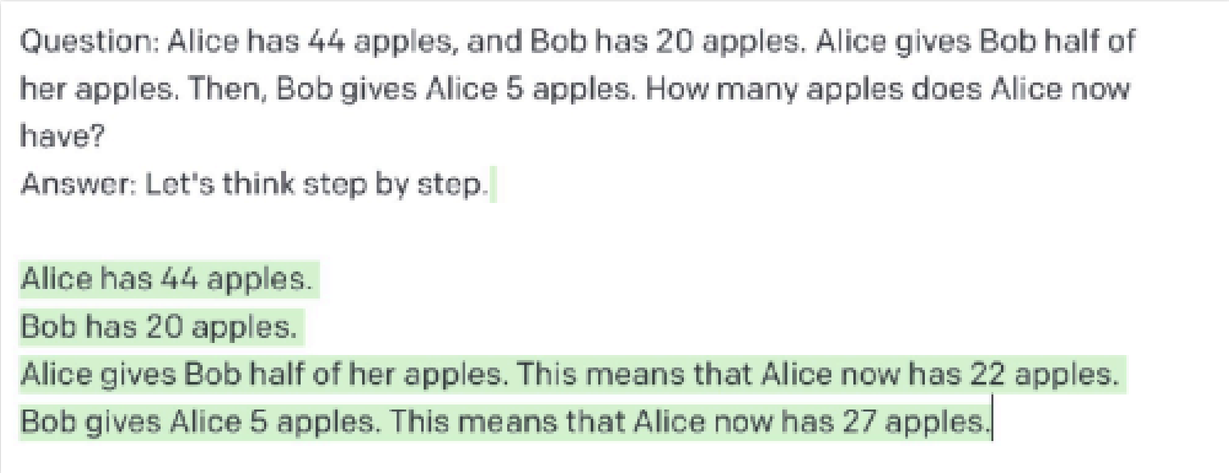
Engine: text-davinci-002
Temperature: 0
Maximum length: 256
Stop sequences: Enter sequence and press Tab
Top P: 1
Frequency penalty: 0
Presence penalty: 0

Let's think step by step

- Simply adding "Let's think step by step." Increases the accuracy on MultiArithm from 17% to 78%, and on GSM8K from 10% to 40%.
- That's it! 🤯



Question: Alice has 44 apples, and Bob has 20 apples. Alice gives Bob half of her apples. Then, Bob gives Alice 5 apples. How many apples does Alice now have?
Answer: Alice has 31 apples.



Question: Alice has 44 apples, and Bob has 20 apples. Alice gives Bob half of her apples. Then, Bob gives Alice 5 apples. How many apples does Alice now have?
Answer: Let's think step by step.

Alice has 44 apples.
Bob has 20 apples.
Alice gives Bob half of her apples. This means that Alice now has 22 apples.
Bob gives Alice 5 apples. This means that Alice now has 27 apples.

ChatGPT Prompt Engineering for Developers

- https://www.deeplearning.ai/short-courses/chatgpt-prompt-engineering-for-developers/

SHORT COURSE

ChatGPT Prompt
Engineering for Developers

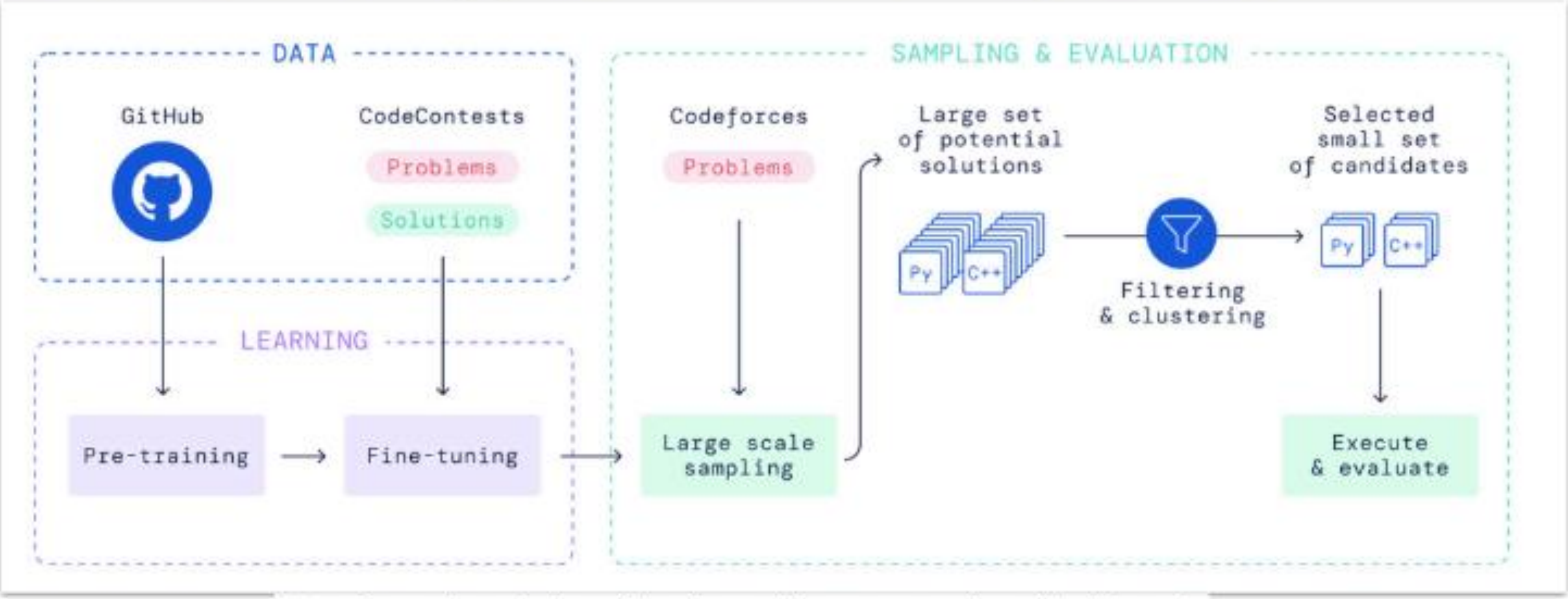
IN COLLABORATION WITH

OpenAI

Code

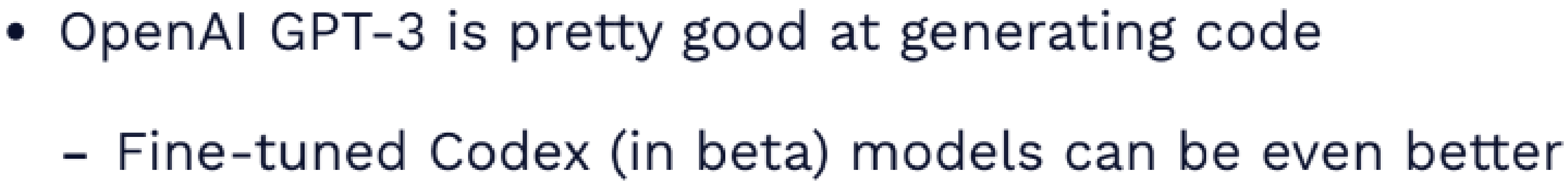
Deepmind Alphacode (2022)

- Pre-trained on Github (715GB) + CodeContests
- Encoder-only 41B model, with filtering
- Top 54% (above average) in Codeforces competitions!



https://www.deepmind.com/blog/competitive-programming-with-alphacode

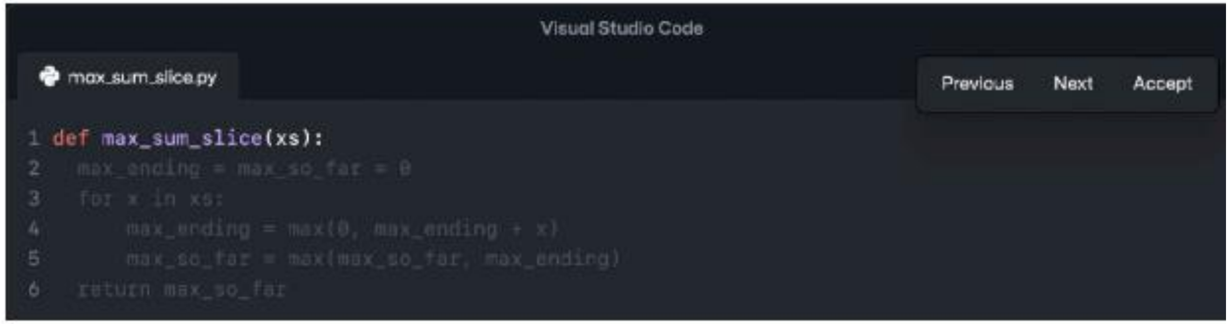
Generating Code



- OpenAI GPT-3 is pretty good at generating code
  - Fine-tuned Codex (in beta) models can be even better

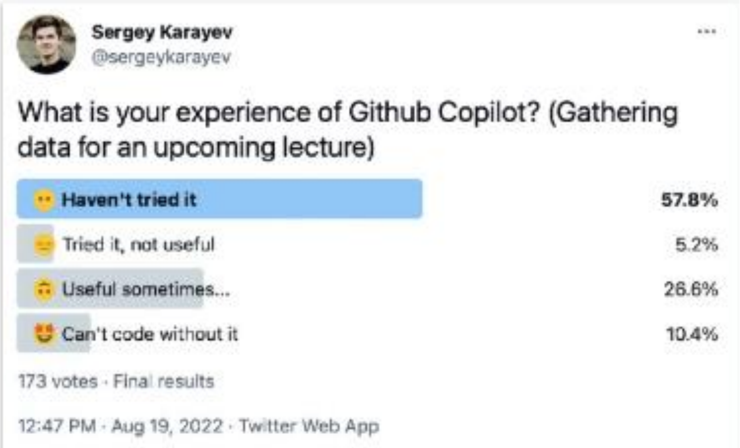
Github Copilot

Github Copilot



```
def max_sum_slice(xs):
  max_ending = max_so_far = 0
  for x in xs:
      max_ending = max(0, max_ending + x)
      max_so_far = max(max_so_far, max_ending)
  return max_so_far
```

- Unobtrusive Codex-powered completions in your code editor
- Try it!



Sergey Karayev
@sergeykarayev

What is your experience of Github Copilot? (Gathering data for an upcoming lecture)

Haven't tried it	**57.8%**
Tried it, not useful	5.2%
Useful sometimes...	26.6%
Can't code without it	10.4%

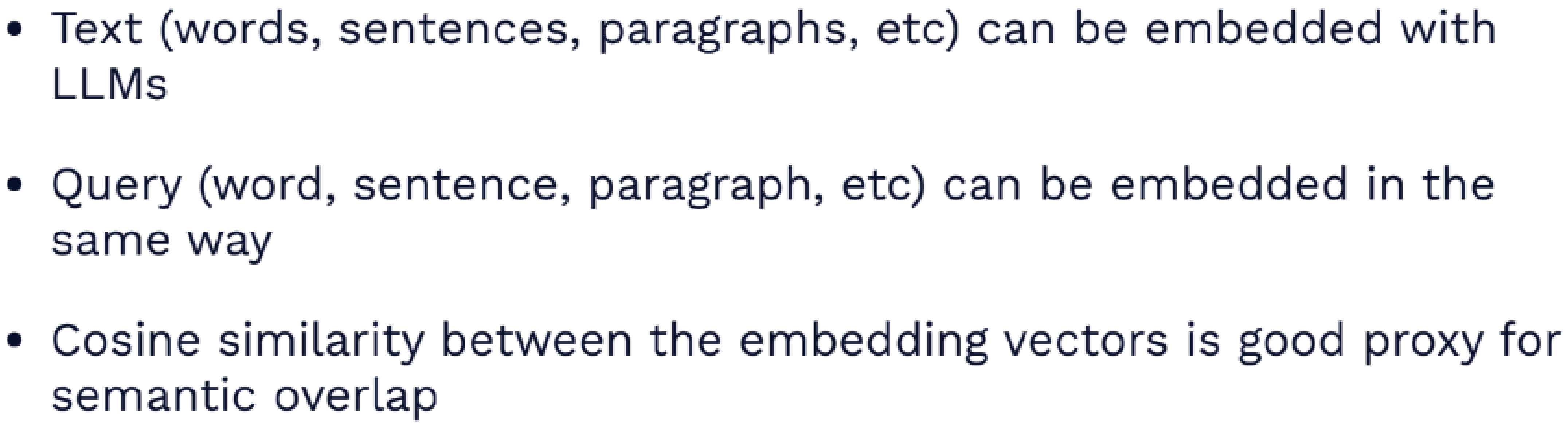
173 votes · Final results

12:47 PM · Aug 19, 2022 · Twitter Web App

@sergeykarayev

Semantic Search

Semantic Search

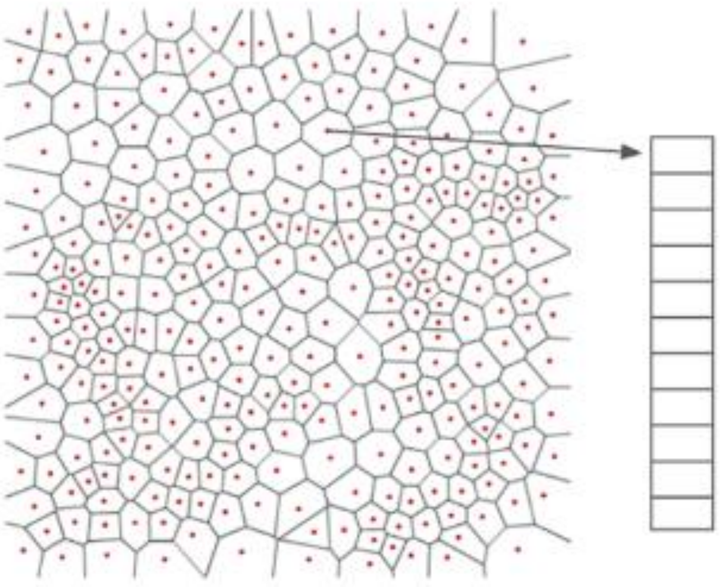


- Text (words, sentences, paragraphs, etc) can be embedded with LLMs
- Query (word, sentence, paragraph, etc) can be embedded in the same way
- Cosine similarity between the embedding vectors is good proxy for semantic overlap

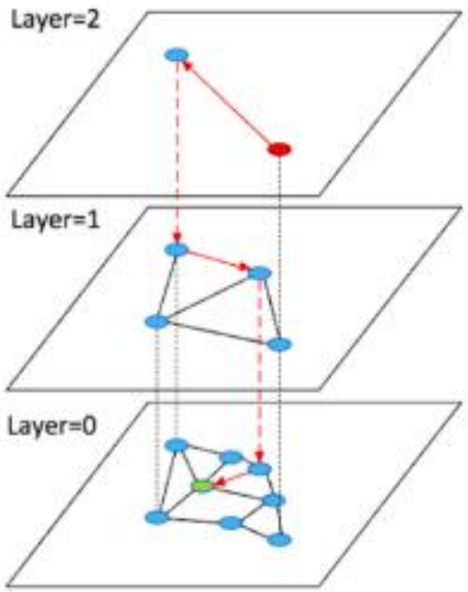
Semantic Search Implementation

- The challenge is the computation required. Dense float vectors of even 512 dimensions don't scale past ~10K.
- Libraries like FAISS and ScaNN make such searching feasible
- Great read from Google

Segmenting the search space

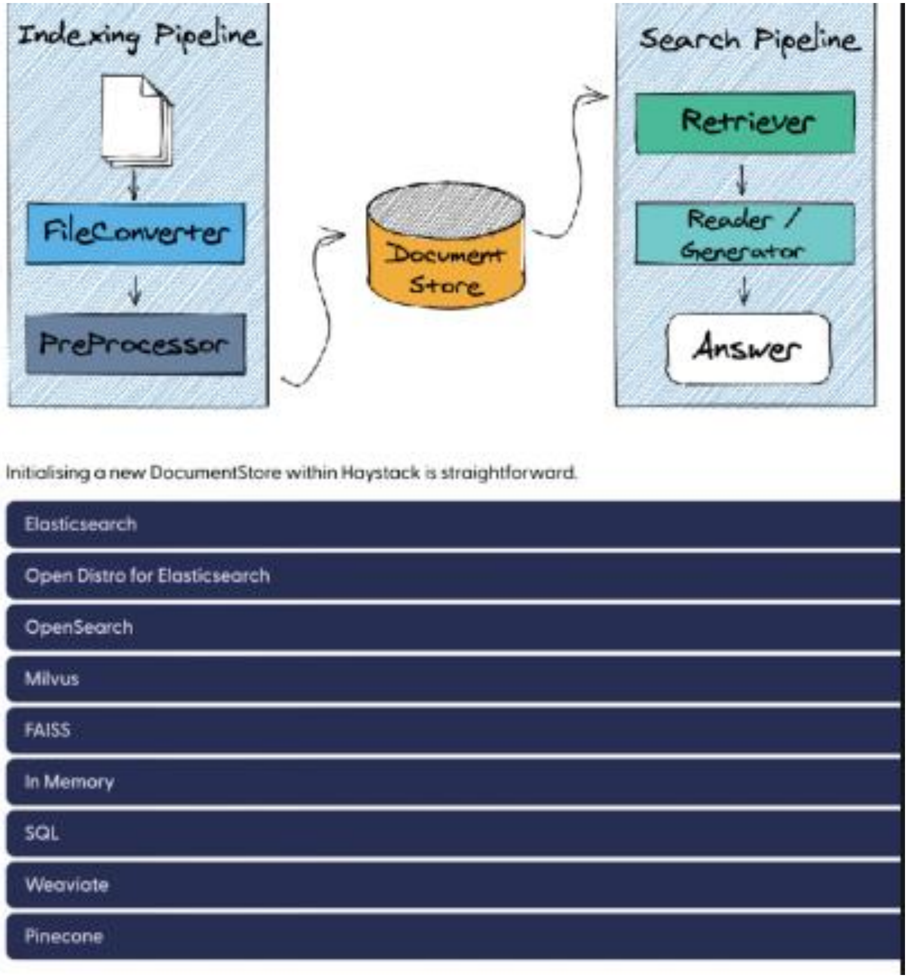


Hierarchical Navigable Small World (average path length loglogN)



Vector Search Open-source

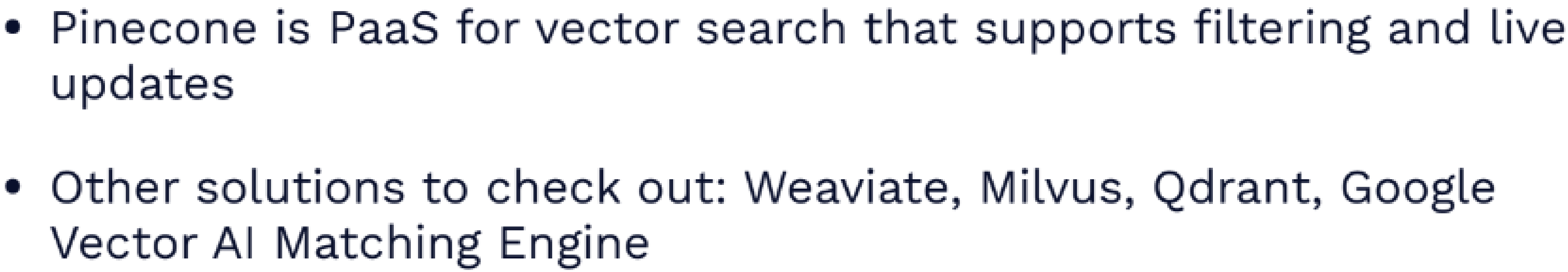
- DeepSet Haystack Python library is a high-level
- Another interesting Python open-source: Jina.ai



Initialising a new DocumentStore within Haystack is straightforward.

- Elasticsearch
- Open Distro for Elasticsearch
- OpenSearch
- Milvus
- FAISS
- In Memory
- SQL
- Weaviate
- Pinecone

Vector Search Vendors

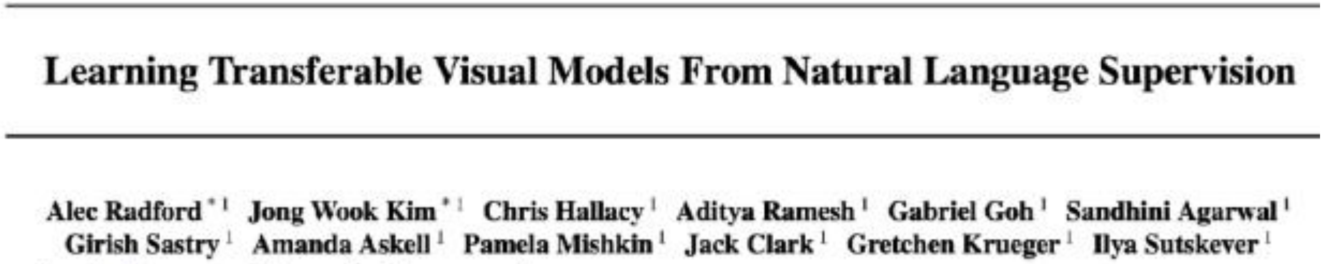


- Pinecone is PaaS for vector search that supports filtering and live updates
- Other solutions to check out: Weaviate, Milvus, Qdrant, Google Vector AI Matching Engine

Going Cross-Modal

CLIP: Contrastive Language-Image Pre-Training

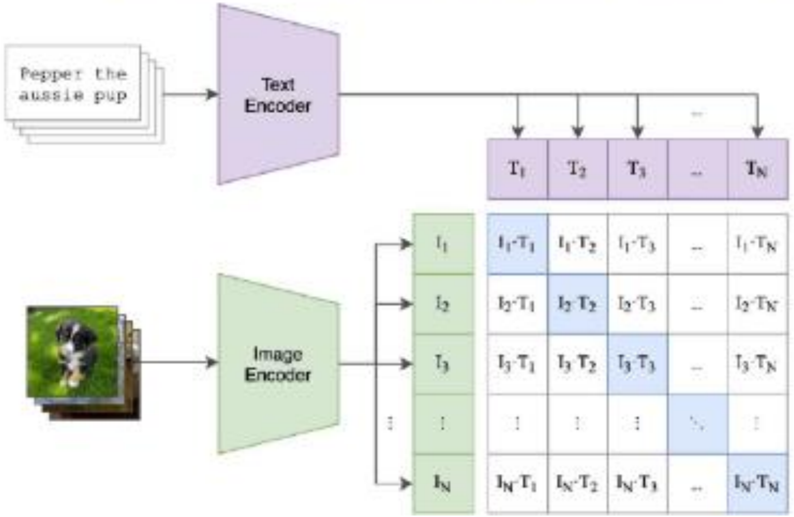
CLIP



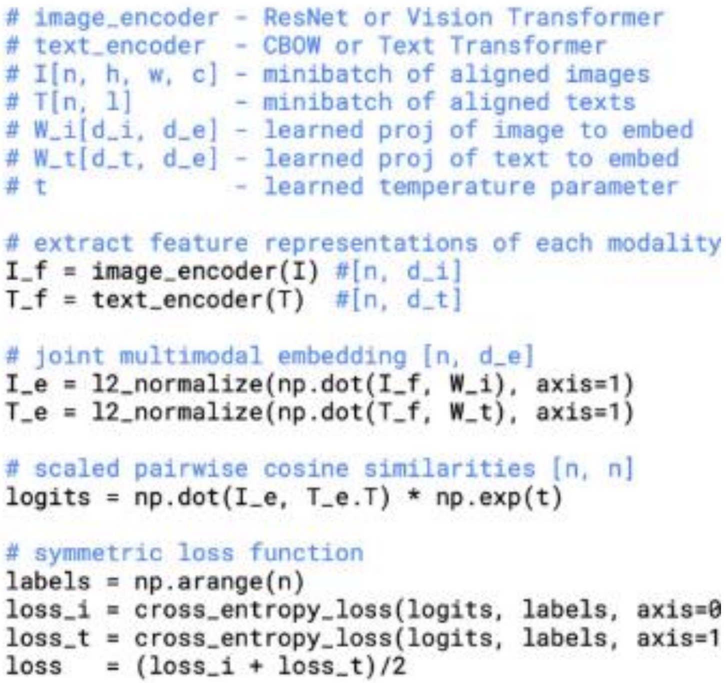
Learning Transferable Visual Models From Natural Language Supervision

Alec Radford [*1] Jong Wook Kim [*1] Chris Hallacy [1] Aditya Ramesh [1] Gabriel Goh [1] Sandhini Agarwal [1]
Girish Sastry [1] Amanda Askell [1] Pamela Mishkin [1] Jack Clark [1] Gretchen Krueger [1] Ilya Sutskever [1]

- 400M image-text pairs crawled from the Internet
- Transformer to encode text, ResNet or Visual Transformer to encode image
- Contrastive training: maximize cosine similarity of correct image-text pairs (32K pairs per batch)



	T_1	T_2	T_3	...	T_N
I_1	$I_1 \cdot T_1$	$I_1 \cdot T_2$	$I_1 \cdot T_3$	...	$I_1 \cdot T_N$
I_2	$I_2 \cdot T_1$	$I_2 \cdot T_2$	$I_2 \cdot T_3$	...	$I_2 \cdot T_N$
I_3	$I_3 \cdot T_1$	$I_3 \cdot T_2$	$I_3 \cdot T_3$	...	$I_3 \cdot T_N$
⋮	⋮	⋮	⋮	⋱	⋮
I_N	$I_N \cdot T_1$	$I_N \cdot T_2$	$I_N \cdot T_3$	...	$I_N \cdot T_N$

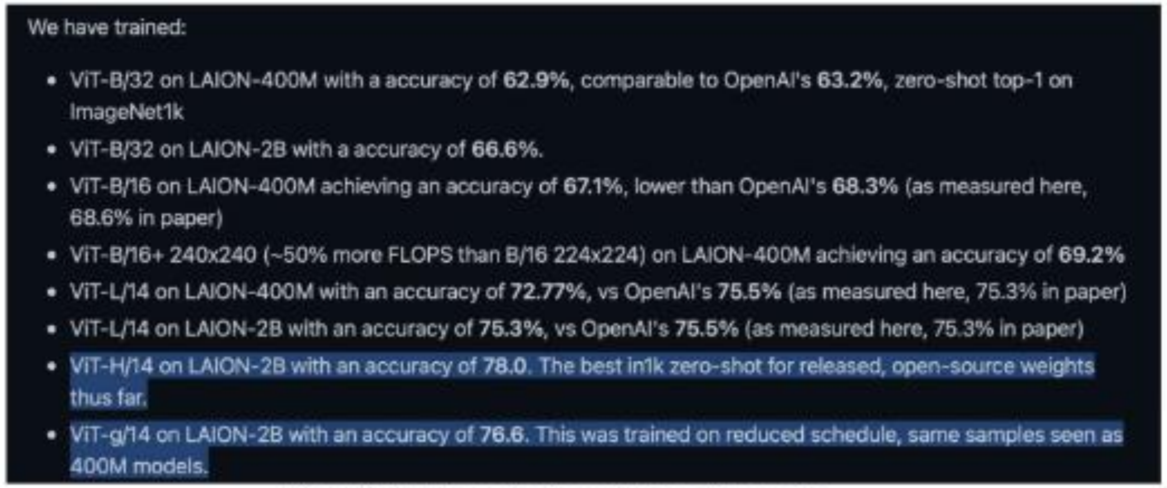


```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0
loss_t = cross_entropy_loss(logits, labels, axis=1
loss   = (loss_i + loss_t)/2
```

https://arxiv.org/pdf/2103.00020.pdf

CLIP Open-source

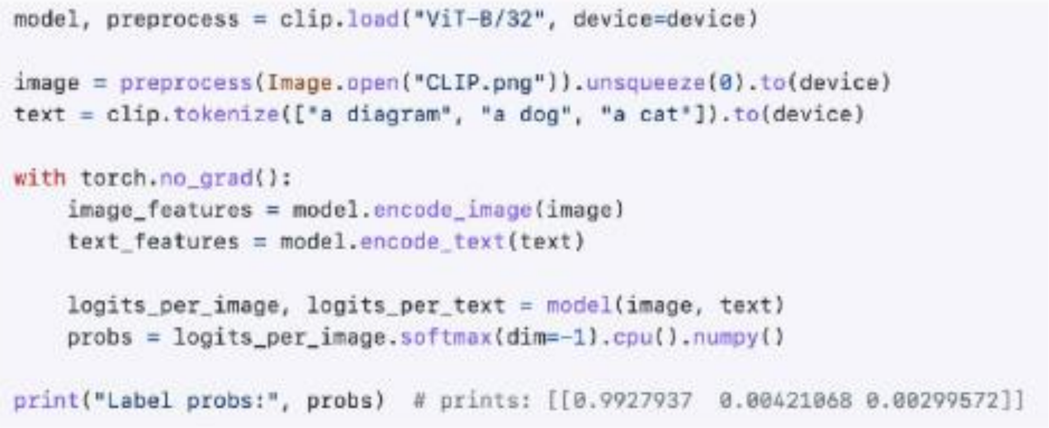
- OpenAI released all trained CLIP models
- OpenCLIP re-trained them on LAION, and published even bigger ones
- Note that CLIP is image → embedding and text → embedding, not image → text or text → image!



We have trained:

- ViT-B/32 on LAION-400M with a accuracy of **62.9%**, comparable to OpenAI's **63.2%**, zero-shot top-1 on ImageNet1k
- ViT-B/32 on LAION-2B with a accuracy of **66.6%**.
- ViT-B/16 on LAION-400M achieving an accuracy of **67.1%**, lower than OpenAI's **68.3%** (as measured here, 68.6% in paper)
- ViT-B/16+ 240x240 (~50% more FLOPS than B/16 224x224) on LAION-400M achieving an accuracy of **69.2%**
- ViT-L/14 on LAION-400M with an accuracy of **72.77%**, vs OpenAI's **75.5%** (as measured here, 75.3% in paper)
- ViT-L/14 on LAION-2B with an accuracy of **75.3%**, vs OpenAI's **75.5%** (as measured here, 75.3% in paper)
- ViT-H/14 on LAION-2B with an accuracy of **78.0**. The best in1k zero-shot for released, open-source weights thus far.
- ViT-g/14 on LAION-2B with an accuracy of **76.6**. This was trained on reduced schedule, same samples seen as 400M models.

https://github.com/mlfoundations/open_clip

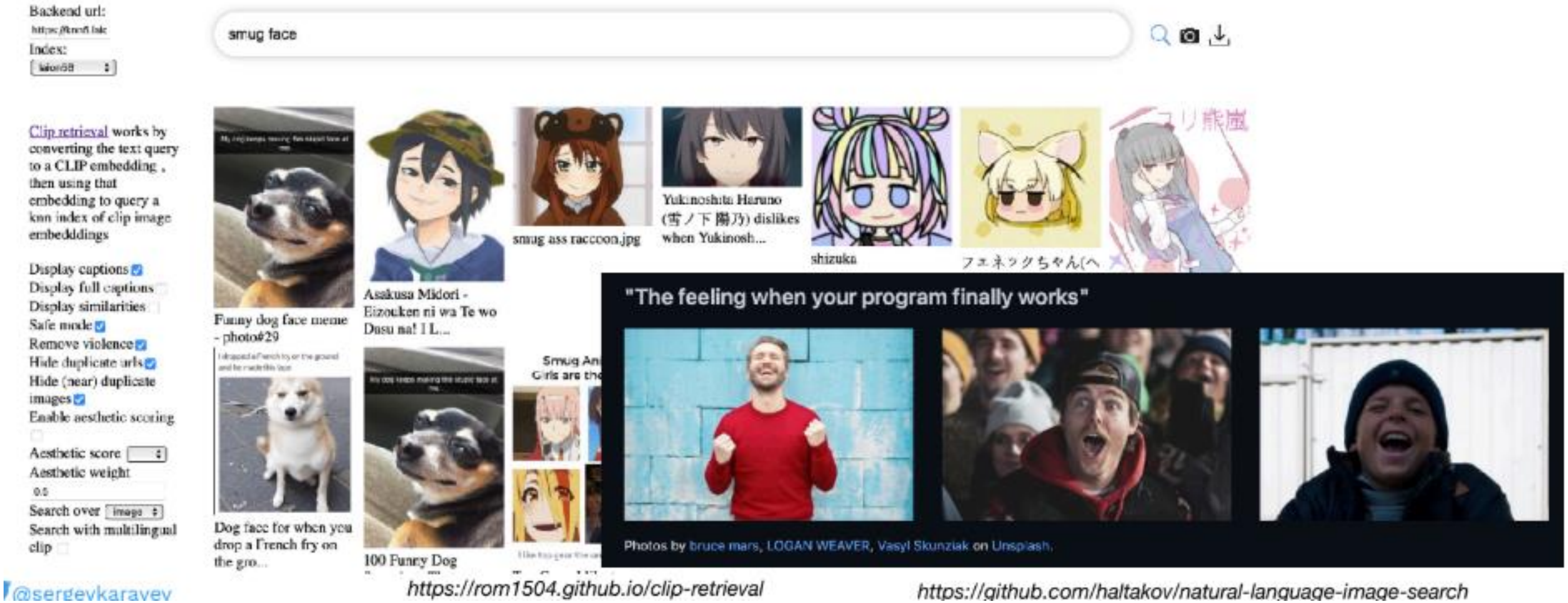


```
model, preprocess = clip.load("ViT-B/32", device=device)

image = preprocess(Image.open("CLIP.png")).unsqueeze(0).to(device)
text = clip.tokenize(["a diagram", "a dog", "a cat"]).to(device)

with torch.no_grad():
    image_features = model.encode_image(image)
    text_features = model.encode_text(text)

    logits_per_image, logits_per_text = model(image, text)
    probs = logits_per_image.softmax(dim=-1).cpu().numpy()

print("Label probs:", probs)  # prints: [[0.9927937  0.00421068 0.00299572]]
```

https://github.com/openai/CLIP

Cross-modal search

- Since CLIP embeds images and text into a shared space, can search images by text and vice versa...



Backend url:

Index: laion5B

smug face

Clip retrieval works by converting the text query to a CLIP embedding , then using that embedding to query a knn index of clip image embedddings

Display captions ☑
Display full captions ☐
Display similarities ☐
Safe mode ☑
Remove violence ☑
Hide duplicate urls ☑
Hide (near) duplicate images ☑
Enable aesthetic scoring ☐
Aesthetic score
Aesthetic weight 0.5
Search over image
Search with multilingual clip ☐

Funny dog face meme - photo#29

Asakusa Midori - Eizouken ni wa Te wo Dasu na! I L...

smug ass raccoon.jpg

Yukinoshita Haruno (雪ノ下陽乃) dislikes when Yukinosh...

shizuka

フェネックちゃん(へ

Dog face for when you drop a French fry on the gro...

100 Funny Dog

"The feeling when your program finally works"

Photos by bruce mars, LOGAN WEAVER, Vasyl Skunziak on Unsplash.

@sergeykarayev

https://rom1504.github.io/clip-retrieval

https://github.com/haltakov/natural-language-image-search

Building Multimodal Search

- https://www.deeplearning.ai/short-courses/building-multimodal-search-and-rag/

DeepLearning.AI

SHORT COURSE

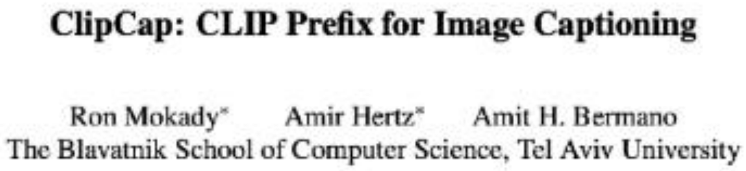
Building Multimodal Search and RAG

IN COLLABORATION WITH

Weaviate

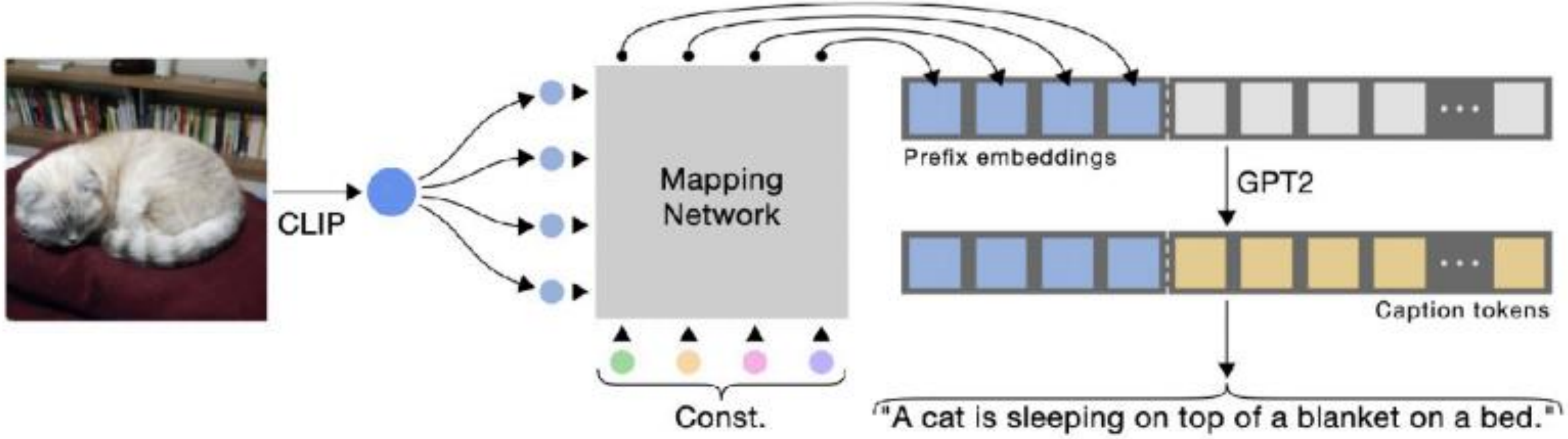
CLIP Image Captioning (image → text)

- One way is ClipCap: train a network to go from CLIP image embedding to a sequence of "word" embeddings that a LLM like GPT-2 can continue
- Training data is (image, caption) pairs, mapping network is a transformer; CLIP and GPT-2 are frozen



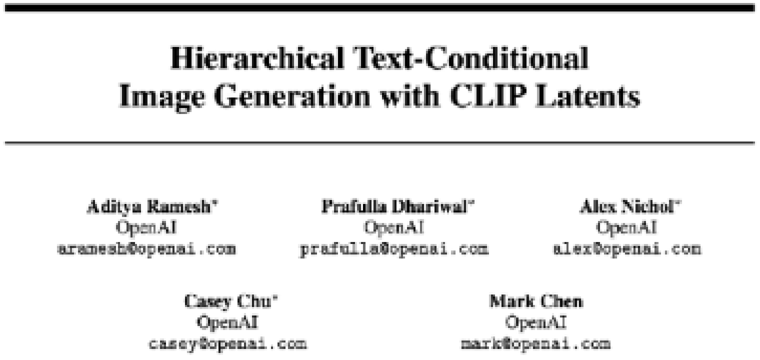
ClipCap: CLIP Prefix for Image Captioning

Ron Mokady* Amir Hertz* Amit H. Bermano
The Blavatnik School of Computer Science, Tel Aviv University



CLIP Image Generation (text → image)

- unCLIP (DALL-E 2)
  - CLIP: text encoder + image encoder
  - *Prior*: mapping from text embedding to image embedding
  - *Decoder*: mapping from image embedding to image
- Unclear training data



Hierarchical Text-Conditional Image Generation with CLIP Latents

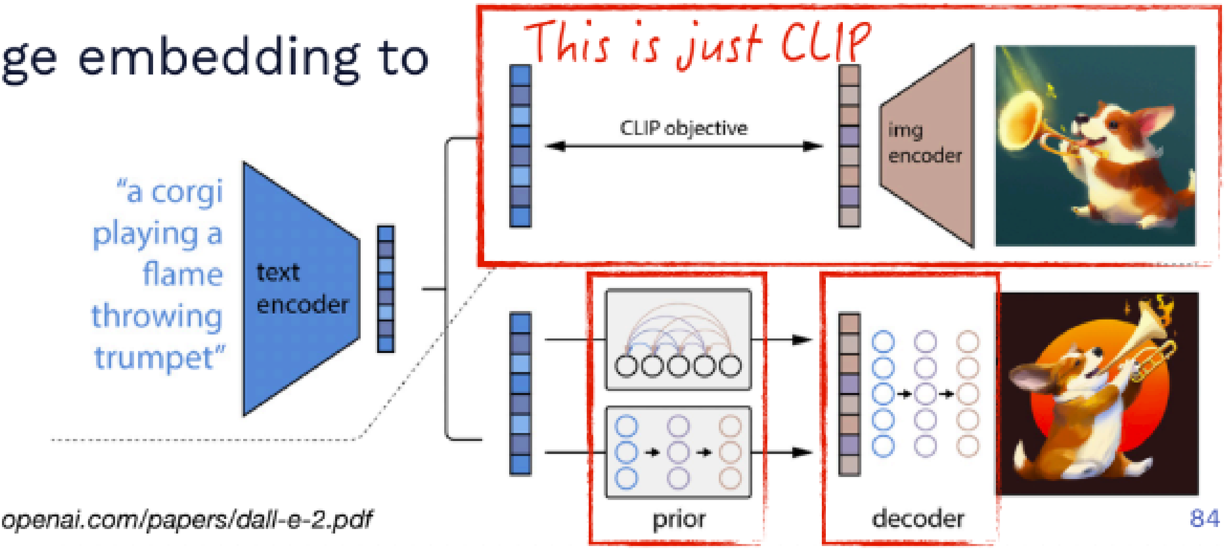
Aditya Ramesh*
OpenAI
`aramesh@openai.com`

Prafulla Dhariwal*
OpenAI
`prafulla@openai.com`

Alex Nichol*
OpenAI
`alex@openai.com`

Casey Chu*
OpenAI
`casey@openai.com`

Mark Chen
OpenAI
`mark@openai.com`



@sergeykarayev

https://cdn.openai.com/papers/dall-e-2.pdf

More ???