ICMAT
INSTITUTO DE CIENCIAS MATEMÁTICAS

komorebi

Zero-shot learning in extremely large Transformer models (GPT and CLIP)

Víctor Gallego
Komorebi AI & ICMAT

Fact 1. One of the core problems of ML...

Is the **lack of labeled examples** for a given task.

Almost every interesting task in industry requires supervised data:

- Sentiment analysis: positive and negative examples.
- Recommendation systems: ranking of the item: ★ ★ ★ ★ ☆
    - Image classification: the classes of the objects: 🐶 (dog), 🐱(cat), 👩 (person) ...

One of the core problems of ML...

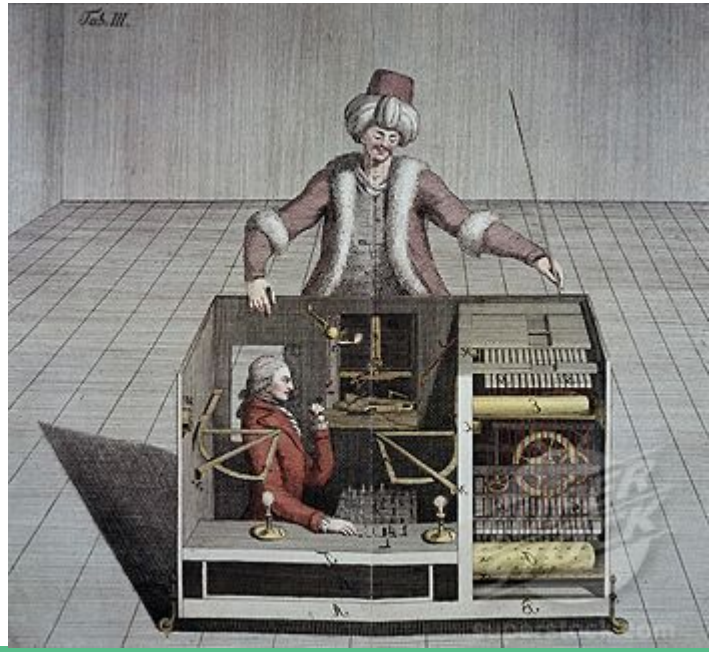
What can you do if you don’t have an annotated dataset?

- Human annotation & crowdsourcing:
  - Expensive (both in € and hours)
  - Boring
  - Doesn’t scale very well

- In this talk, we will focus on an alternative:

zero-shot learning

Mechanical Turk, 18th century:

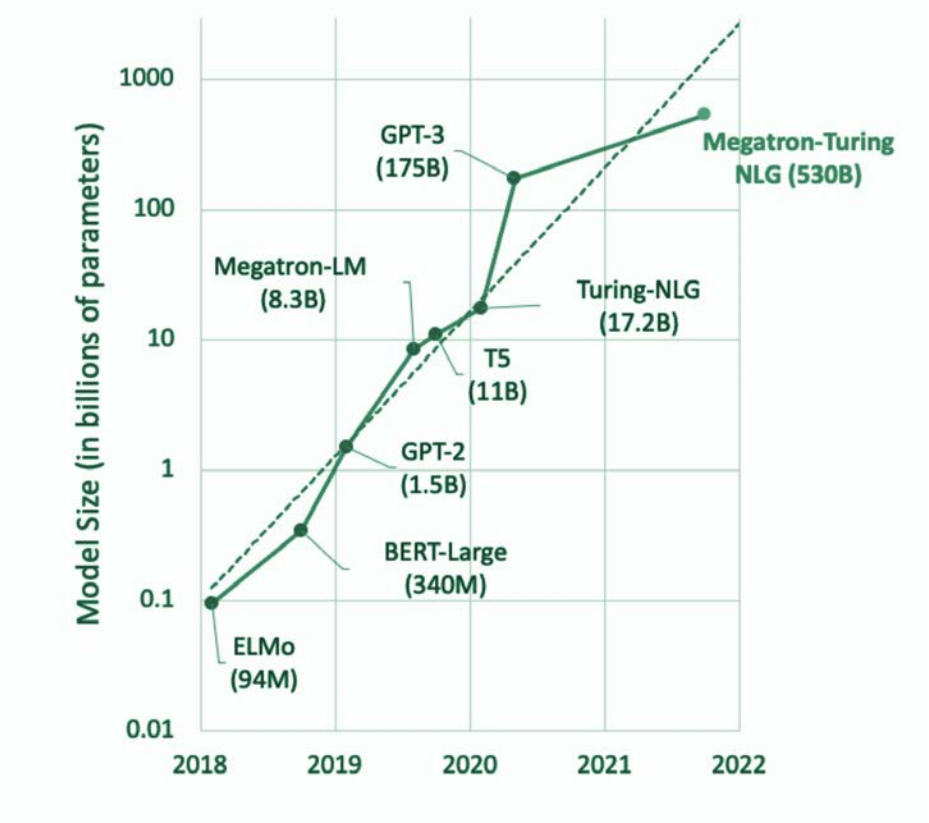


Fact 2. About the size of the models

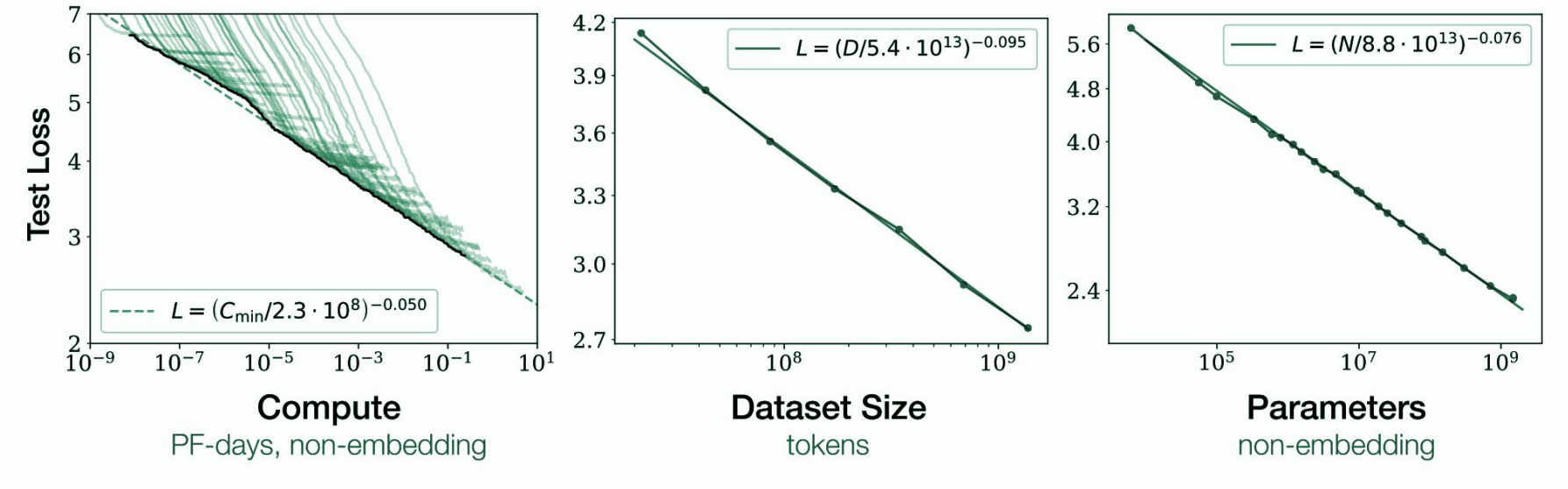
State-of-the-art models in NLP and CV **keep getting bigger and bigger** (in parameter count)

This is similar to the Moore's law of transistors.

Why is this?



The scaling law hypothesis of AI



See: “Scaling Laws for Neural Language Models”, https://arxiv.org/abs/2001.08361

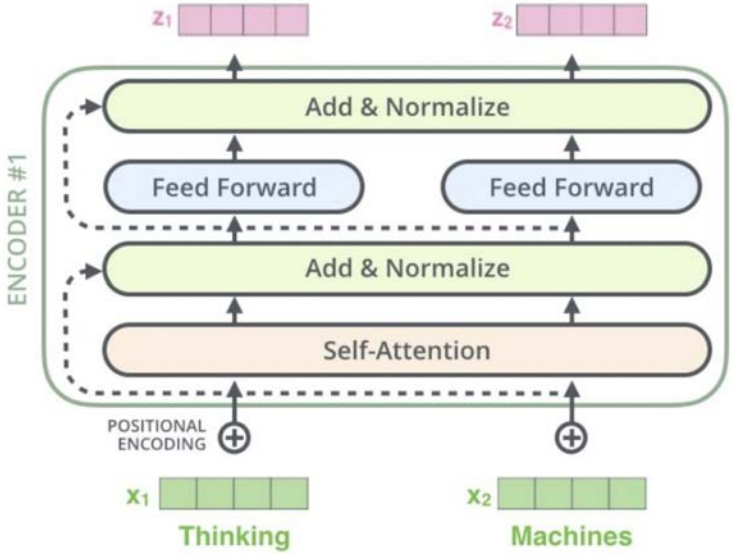
Some people believe this is the way towards true artificial intelligence:

“just make your model bigger”

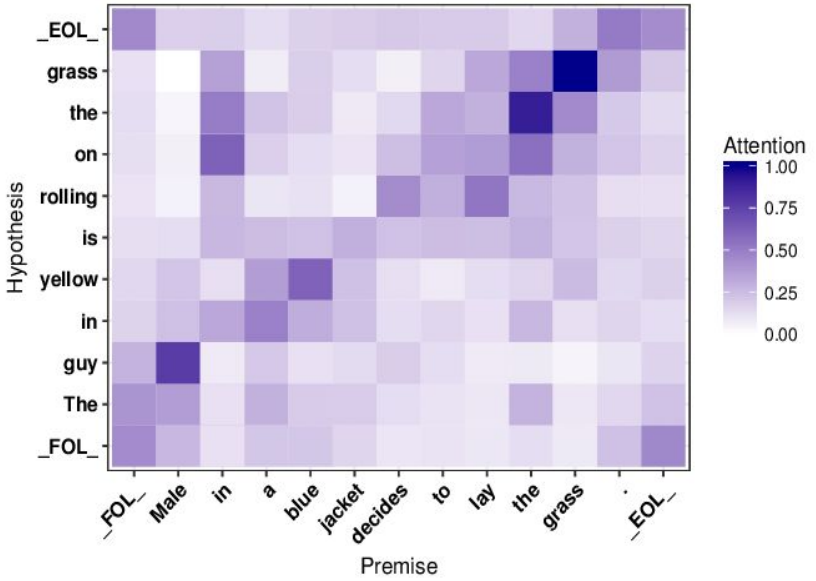
The core ingredient: the Transformer

Introduced in 2017: “Attention is all you need”, https://arxiv.org/abs/1706.03762

The main block is:



Self-attention is a way to correlate similar words in an example:



Can we use fact 2 to “solve” fact 1?

In this talk, we will explore several large-scale models that enable zero-shot learning

Contrastive learning

CLIP architecture, introduced in 2021, https://arxiv.org/abs/2103.00020

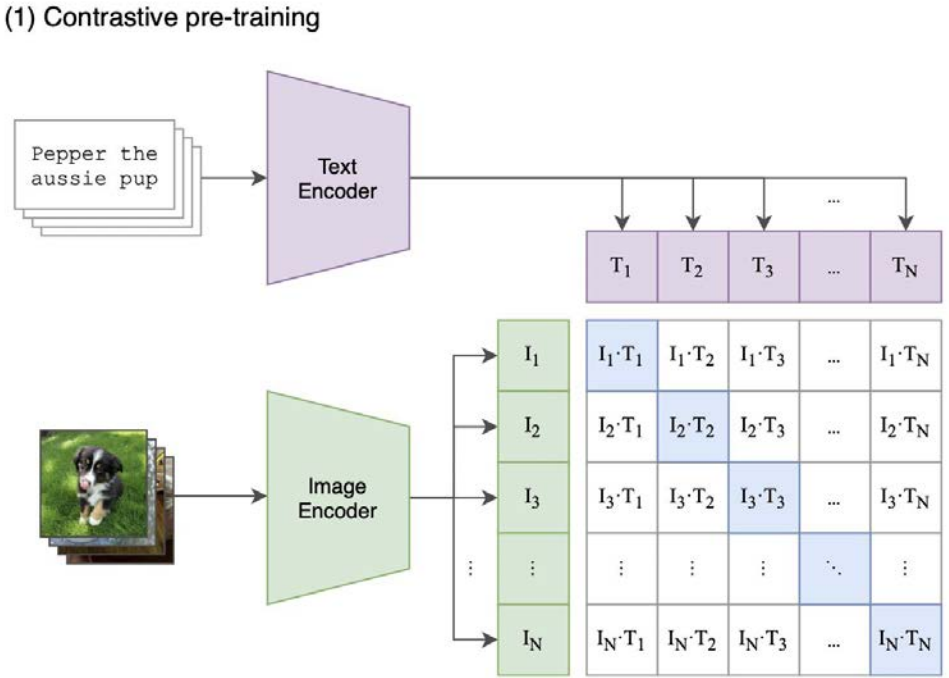
Trained over **400M pairs of (image, caption)** from the internet.

Consists in two components:

- Image encoder
- Text encoder

$$I_i = encoder_{\theta_1}(x_{i,img})$$
$$T_i = encoder_{\theta_2}(x_{i,text})$$

Loss function is **cross-correlation** between the embeddings of the images and the associated texts.



Zero-shot classification

This contrastive learning approach enables the specification of “labels” **only at inference time**

Just compute the embeddings for each class:

$$T_1 = encoder_{\theta_2}(\text{"}dog\text{"})$$

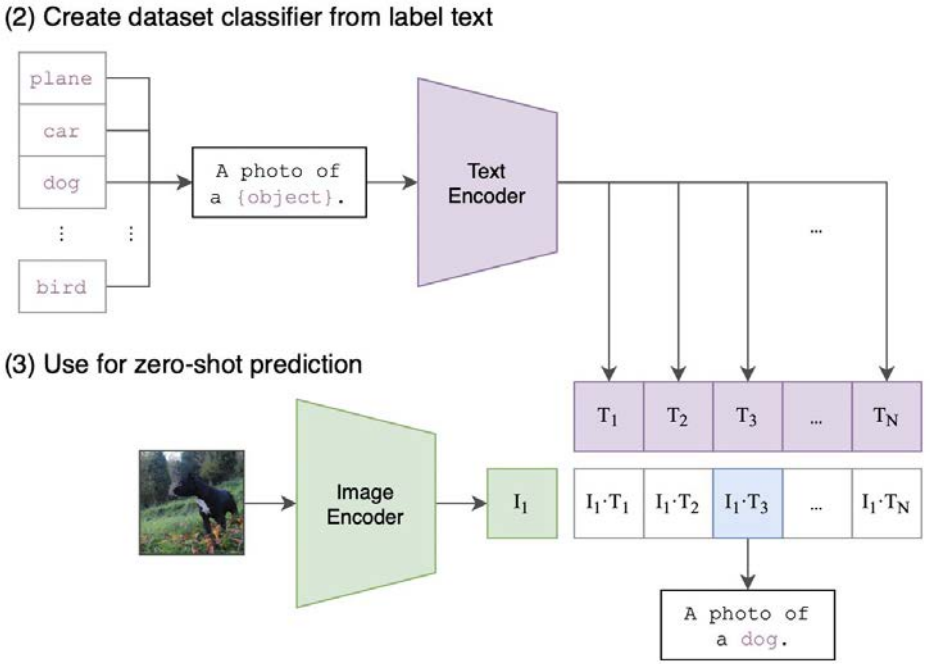
$$T_2 = encoder_{\theta_2}(\text{"}cat\text{"})$$

the embedding for the image:

$$I = encoder_{\theta_1}(x)$$

and a dot product to get the logits:

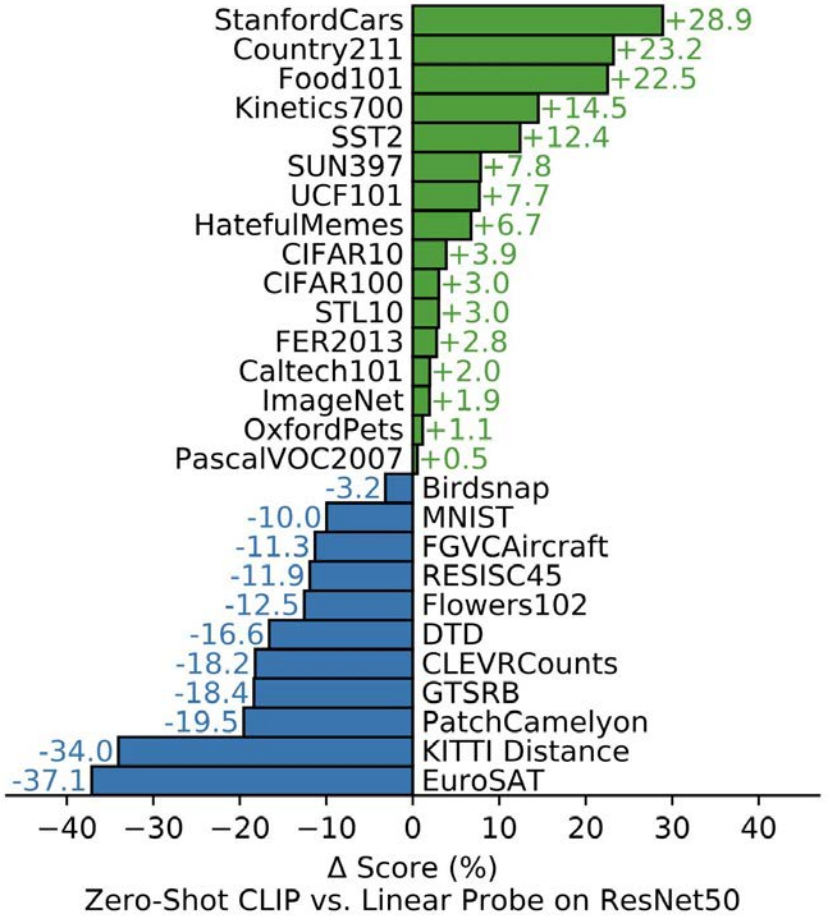
$$logits = I \begin{bmatrix} T_1 & T_2 & \dots \end{bmatrix}^{\intercal}$$



Zero-shot classification

It works very well for several standard computer vision datasets

But still gaps with SOTA models using lots of training data

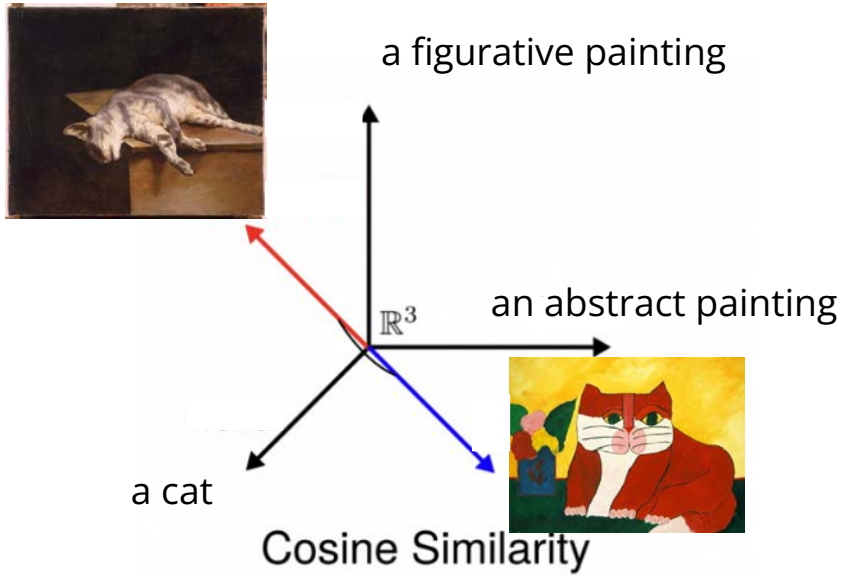


Zero-shot semantic search

Instead of fixing the textual labels and querying with images, we can do the opposite: **keep the images fixed (like a database) and query them with text**

Essentially, it is **a glorified nearest neighbors search.**

But in an extremely rich embedding space: $\mathbb{R}^{512}$



Demo time!

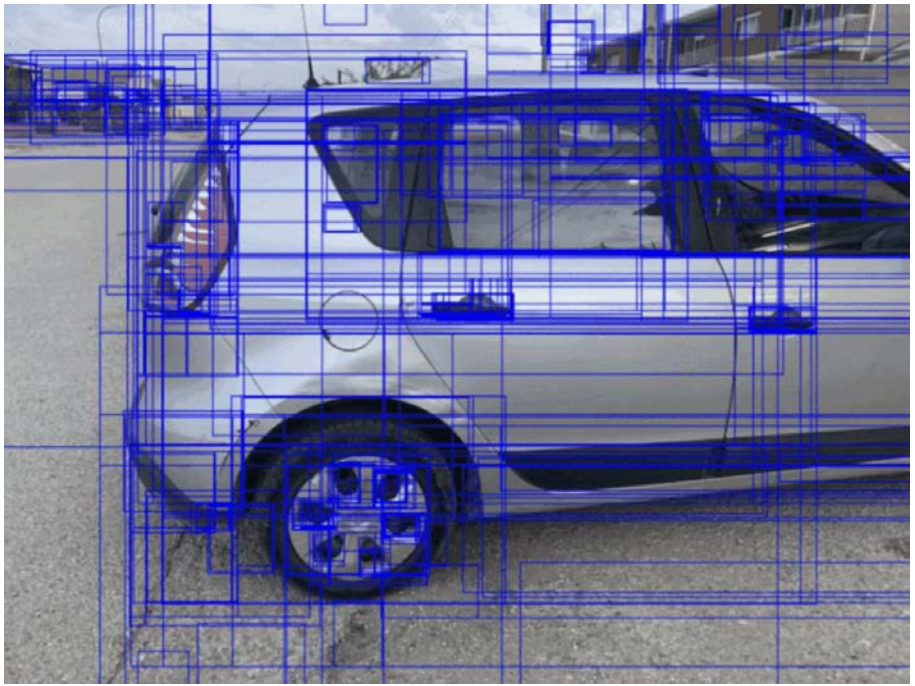
https://dev.komorebi.ai/art-explorer/

Zero-shot object detection

1. Using classical computer vision techniques, generate K **regions of interest**
2. Use CLIP to filter them, using:
   a. keywords (“dented metallic surface)
   b. similar defects

We are using it in car damage detection, to generate labeled datasets of car defects.

https://insurmapp.com



Zero-shot object detection

1. Using classical computer vision techniques, generate K **regions of interest**
2. Use CLIP to filter them, using:
   a. keywords (“dented metallic surface)
   b. similar defects

We are using it in car damage detection, to generate labeled datasets of car defects.

https://insurmapp.com

Language modeling

What if only textual data?

GPT-like language models consist in Transformer blocks trained using the following objective: **autorregressive language modelling**

$$p(x) = \prod_{i=1}^{n} p(s_n | s_1, ..., s_{n-1})$$

previous tokens

paragraph

current token / word

i.e. predict the next word given a sequence of words

Language modeling 2

We are using GPT-J, one of the biggest open-source models

- 6.7 billion parameters (around 32 GB of RAM)
- Trained over 800 GB of raw text from the internet
  The Pile dataset, https://arxiv.org/abs/2101.00027

Turns out, sufficiently big models like this **exhibit zero-shot capabilities**

See, e.g., the GPT-3 paper https://arxiv.org/abs/2005.14165

Zero-shot learning with language models

Zero-shot

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

```
1  Translate English to French:        ← task description
2  cheese =>                           ← prompt
```

One-shot

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

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← example
3  cheese =>                           ← prompt
```

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.

```
1  sea otter => loutre de mer          ← example #1
```

↓

gradient update

↓

```
1  peppermint => menthe poivrée        ← example #2
```

↓

gradient update

↓

• • •

↓

```
1  plush giraffe => girafe peluche     ← example #N
```

gradient update

Demo time!

https://github.com/vicgalle/gpt-j-api

What is the cost of a big language model?

Inference times (in seconds) for generating a sequence of length 32, 128, or 512 tokens

device	time_32 (s)	time_128 (s)	time_512 (s)	cost (USD/month)
CPU	16	72	300	500
GPU (V100)	1.6	8.4	35	2500
TPU (v2-8)	0.9	2.8	11	3200
TPU (v3-8)	0.7	2.3	8.9	6400

Stochastic Parrots

For the previous reasons, large scale language models are often called **stochastic parrots**.

⚠️ **Caution when deployed!** ⚠️

They will be reinforcing biases and social stereotypes reflected in the data they were pretrained on.

See: "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?"

Let’s assume we have a very small dataset (N < 100).
It’s very expensive to label more data

How do you fine-tune a large (+10^9 parameters) model?

Let’s assume we have a very small dataset (N < 100).
It’s very expensive to label more data

How do you fine-tune a large (+10^9 parameters) model?

You shouldn’t.

Current approaches

Prompt engineering: what we have just seen

Soft prompt tuning: requires the training of a subset of the parameters

https://arxiv.org/abs/2104.08691v2

Wouldn’t it be great if we could learn from a few examples without modifying the parameters of a large language model?

Functional Gradient Descent 1

The usual equation of gradient descent in ML:

$$\theta' = \theta - \alpha \nabla_\theta \ell(f_\theta(x), y)$$

Keep in mind that θ lives now in a really big space (+1 billion dimensions)

Very costly both in compute time and storage

Remember GPT-J takes around 32 GB of RAM, and you would have to host a different copy for each fine-tuned task

Functional Gradient Descent 2

We can assume the model function f lives in a
reproducing kernel Hilbert space (RKHS)

Under this assumption, f can be expressed as

$$f(\cdot) = \sum_{i=1}^{Q} \alpha_i K(\mathbf{x_i}, \cdot)$$

where the x_i are a set of training examples (< 100, for few-shot setting)

and we can differentiate wrt to it (in a functional sense):

$$f(x)' = f(x) - \alpha \nabla_f \ell(f(x), y)$$

Functional Gradient Descent 3

Thus instead of iterating like:

$$\theta' = \theta - \alpha \nabla_\theta \ell(f_\theta(x), y)$$

$\mathbb{R}^{1000000000}$

We iterate using this update rule:

similarity in last layers

can be cached

$$f(x)' = f(x) - \alpha \sum_i K(x, x_i) \ell'(f(x_i), y_i)$$

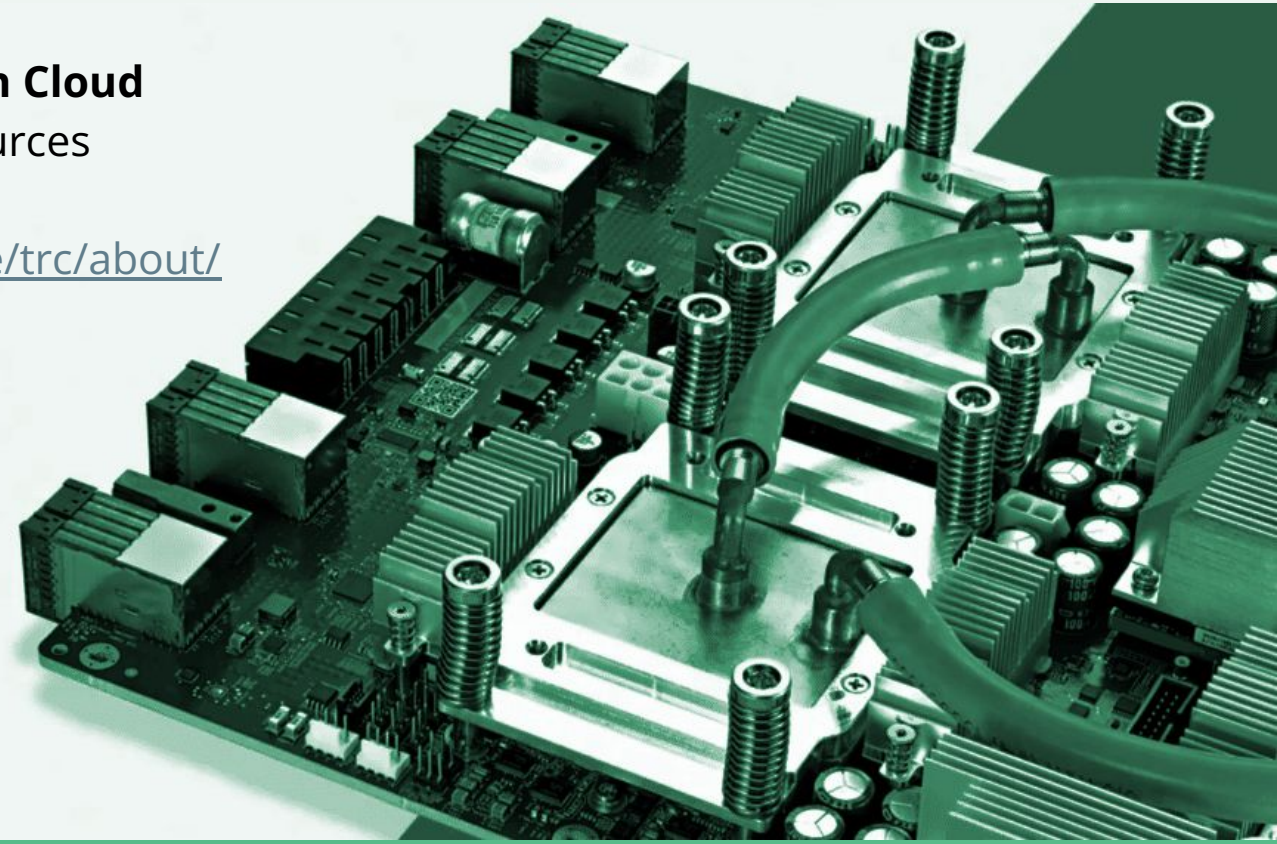
$\mathbb{R}^{512}$

Improved performance and storage cost for few-shot learning compared to finetuning/adapting in CLIP and language models: paper soon.

Acknowledgements

Thanks to the **TPU Research Cloud**
for providing compute resources

https://sites.research.google/trc/about/



Special thanks to the Komorebi AI team

Manuel Navarro

David Gordo

Alberto Torres

David Gómez-Ullate

We are open to collaborations!!

komorebi

https://komorebi.ai

Thank you!!

- victor.gallego@komorebi.ai
- Semantic search demo: https://dev.komorebi.ai/art-explorer/
- GPT-J API code and demos: https://github.com/vicgalle/gpt-j-api