Yandex

Petals: Platform for inference and fine-tuning of the world's largest language models

Alexander Borzunov,

Yandex Research

How to train SOTA in 2012

AlexNet

2x GTX 580 GPU (3 GB VRAM)

5-6 days

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Communications of the ACM* 60.6 (2017): 84-90.

How to train SOTA today

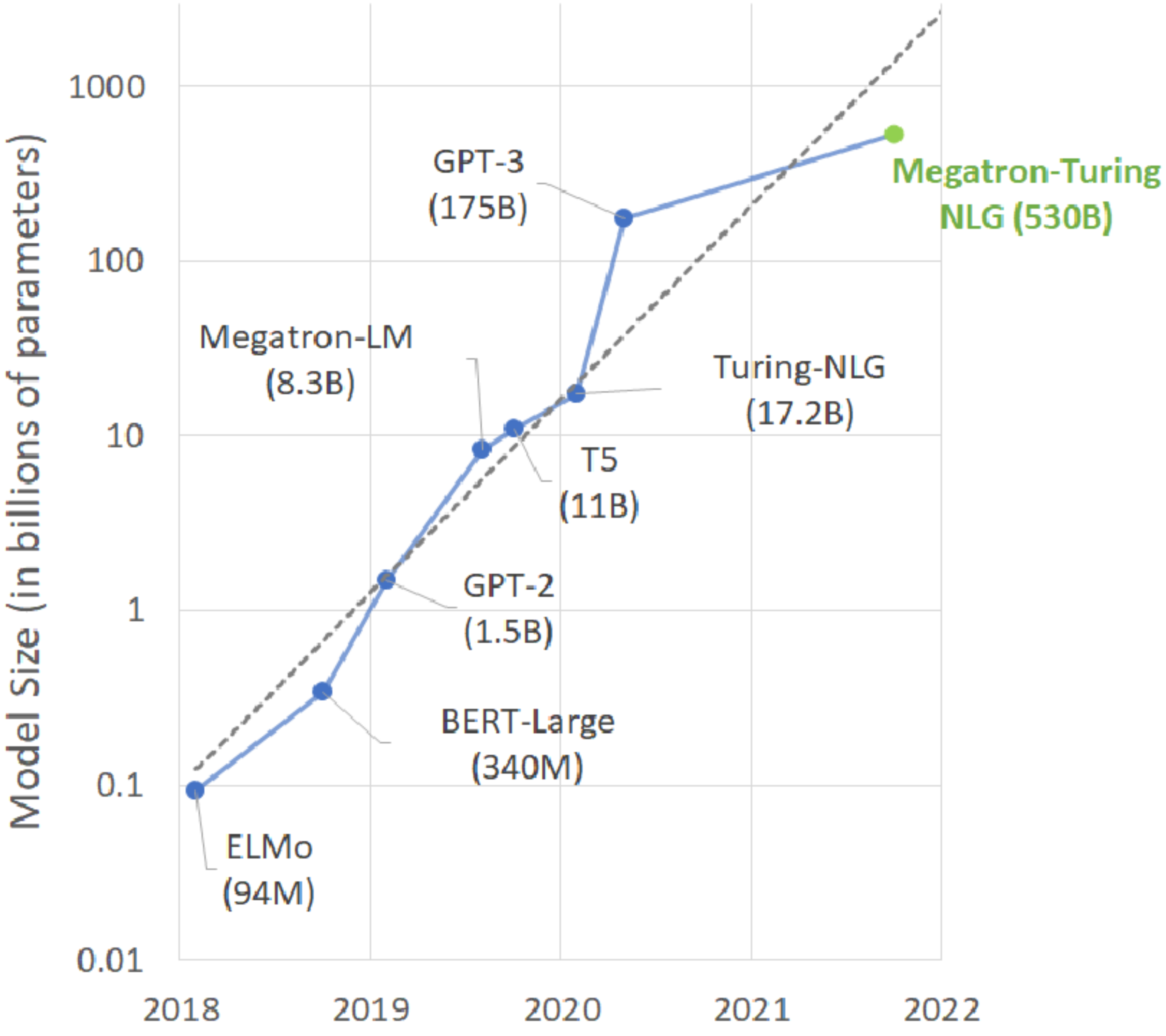
CoAtNet

TPU v3

20 000 days

Dai, Zihang, et al. "Coatnet: Marrying convolution and attention for all data sizes." *Advances in Neural Information Processing Systems* 34 (2021): 3965-3977.

Model size grows exponentially



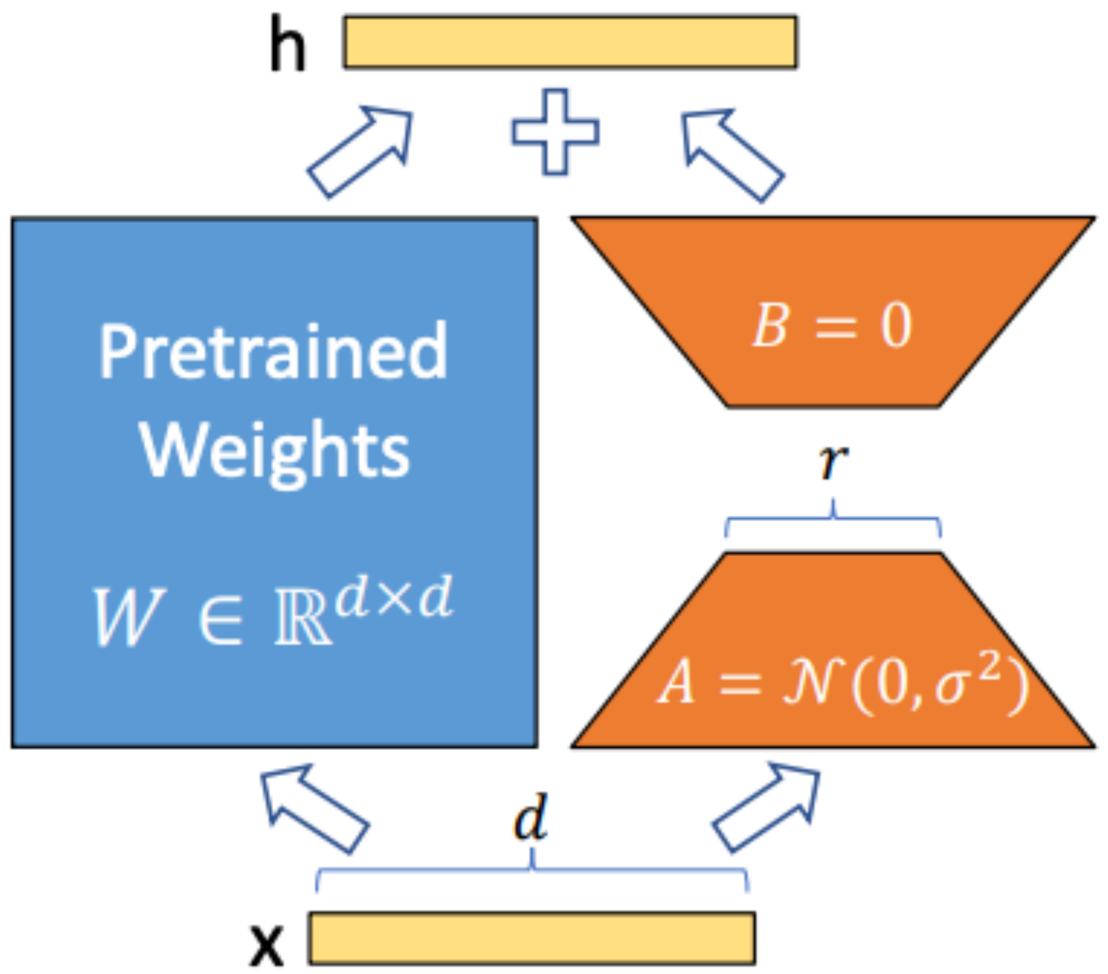
Smith, Shaden, et al. "Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model." *arXiv preprint arXiv:2201.11990* (2022).

Why large models are useful?

- Deep learning benefits from lots of data
  - Getting lots of **labeled** data is hard
  - But getting lots of **unlabeled** data is easy
- What to do?
  - Researchers train a large “foundation model” on unlabeled data, so it “understands” this kind of data
  - Like BERT, ViT, data2vec, etc.
  - Practicioners use this model for **downstream** tasks
  - They don’t need much data for that (sometimes no data at all)

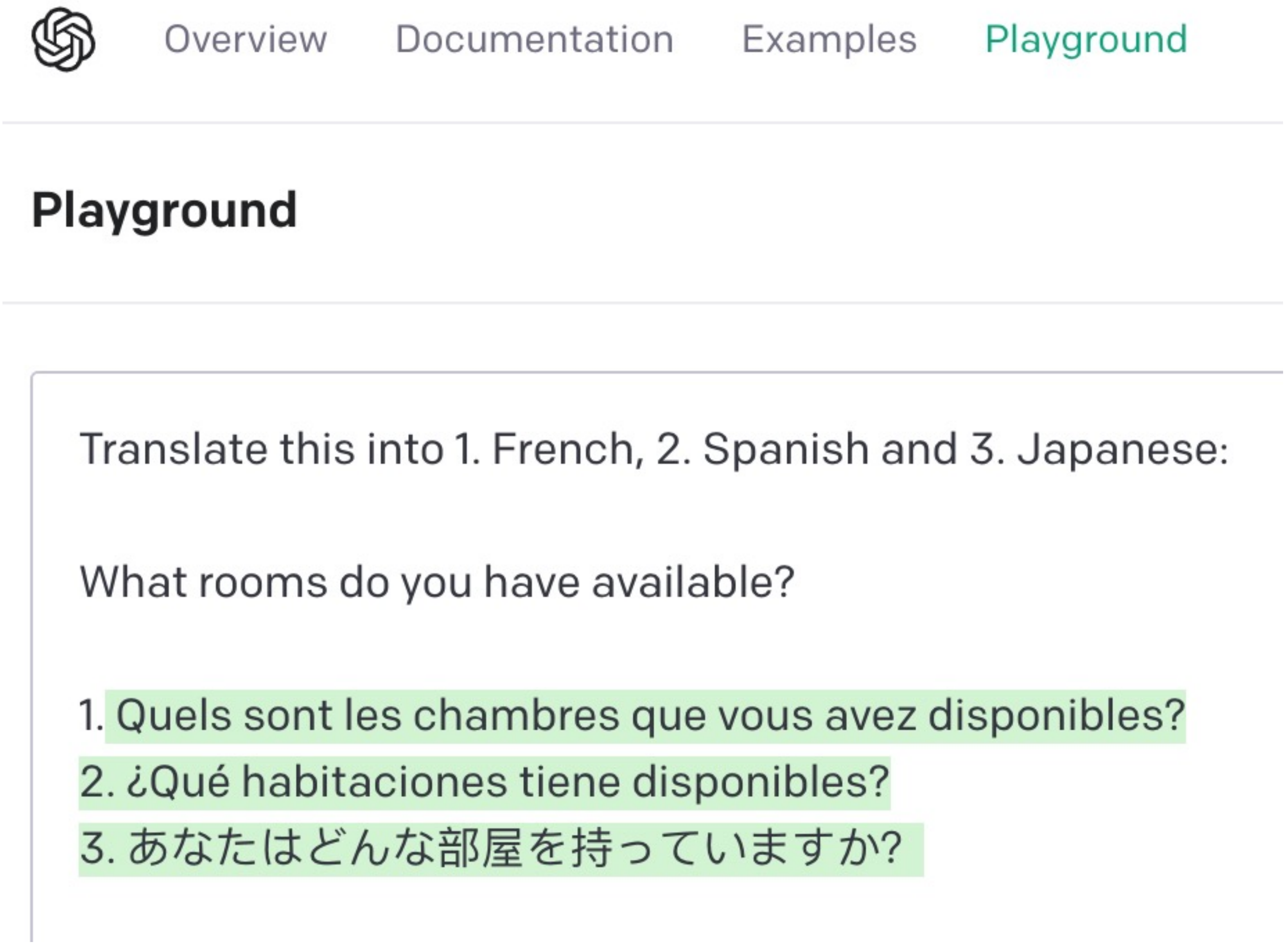
Using pretrained model: Fine-tuning

- Fine-tuning the full model is hard
- We can use parameter-efficient adapters (e.g., low-rank adapters)



Hu, Edward J., et al. "LoRA: Low-rank adaptation of large language models." *arXiv preprint arXiv:2106.09685* (2021).

Using pretrained model: Zero-shot



Brown, Tom, et al. "Language models are few-shot learners." *Advances in neural information processing systems* 33 (2020): 1877-1901.

Using pretrained model: Few-shot

Q: хочу купить 2 капучино, одно латте и 3 пончика с глазурью

A: [{'prod': 'капучино', 'amount': 2}, {'prod': 'латте', 'amount': 1}, {'prod': 'пончика с глазурью', 'amount': 3}]

Q: дайте, пожалуйста, один круассан, два латте, жвачку и пиво

A: [{ 'prod': 'круассан', 'amount': 1}, {'prod': 'латте', 'amount': 2}, {'prod': 'жвачку', 'amount': 1}, {'prod': 'пиво', 'amount': 1}]

Q: привезите чипсы и пиво

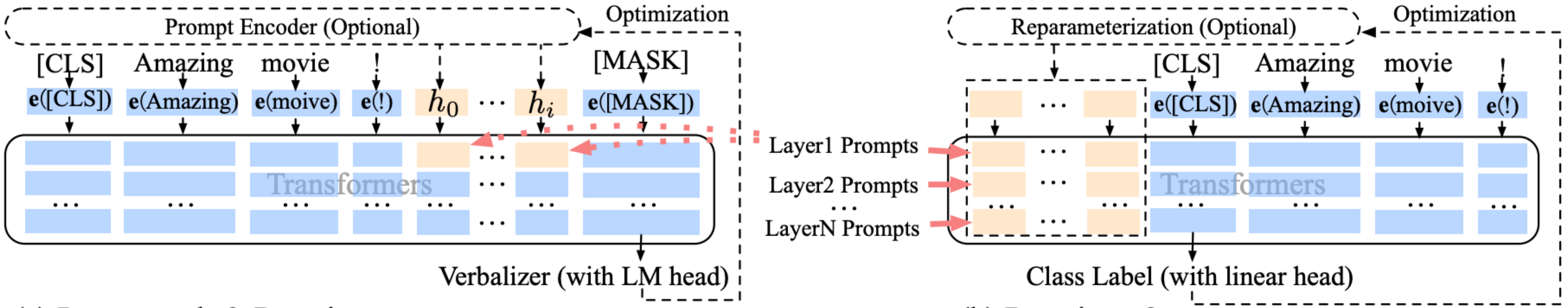
A: [{ 'prod': 'чипсы', 'amount': 1}, {'prod': 'пиво', 'amount': 1}]

Q: можете дать два круассана, 3 капучино и булочку с вареньем?

A: **[{ 'prod': 'круассана', 'amount': 2}, {'prod': 'капучино', 'amount': 3}, {'prod': 'булочку с вареньем', 'amount': 1}]**

Brown, Tom, et al. "Language models are few-shot learners." *Advances in neural information processing systems* 33 (2020): 1877-1901.

Using pretrained model: Prompt tuning



(a) Lester et al. & P-tuning (Frozen, 10-billion-scale, simple tasks)

(b) P-tuning v2 (Frozen, most scales, most tasks)

Figure 2: From Lester et al. (2021) & P-tuning to P-tuning v2. Orange blocks (i.e., $h_0, ..., h_i$) refer to trainable prompt embeddings; blue blocks are embeddings stored or computed by frozen pre-trained language models.

Li, Xiang Lisa, and Percy Liang. "Prefix-tuning: Optimizing continuous prompts for generation." *arXiv preprint arXiv:2101.00190* (2021).

Lester, Brian, Rami Al-Rfou, and Noah Constant. "The power of scale for parameter-efficient prompt tuning." *arXiv preprint arXiv:2104.08691* (2021).

Liu, Xiao, et al. "P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks." *arXiv preprint arXiv:2110.07602* (2021).

Using pretrained model: Method comparison

	Few-shot	P-tuning	Finetuning
Необходимый размер выборки для обучения (# примеров)	~10	≧100-1000	≧10 000
Время инженера	Много времени на подбор подводки	ε на разметку ~100 примеров	0
Итоговое качество (на соотв. объёме данных)	Частые артефакты даже в простых задачах. Непригоден для сложных	Хорошее качество на малых объёмах данных. Часто не отстаёт от finetuning на больших объёмах	Не работает на малых объёмах данных. Максимальное качество на больших объёмах
Время обучения	0	Часы	Дни
Вычислительные ресурсы	•	••	••••

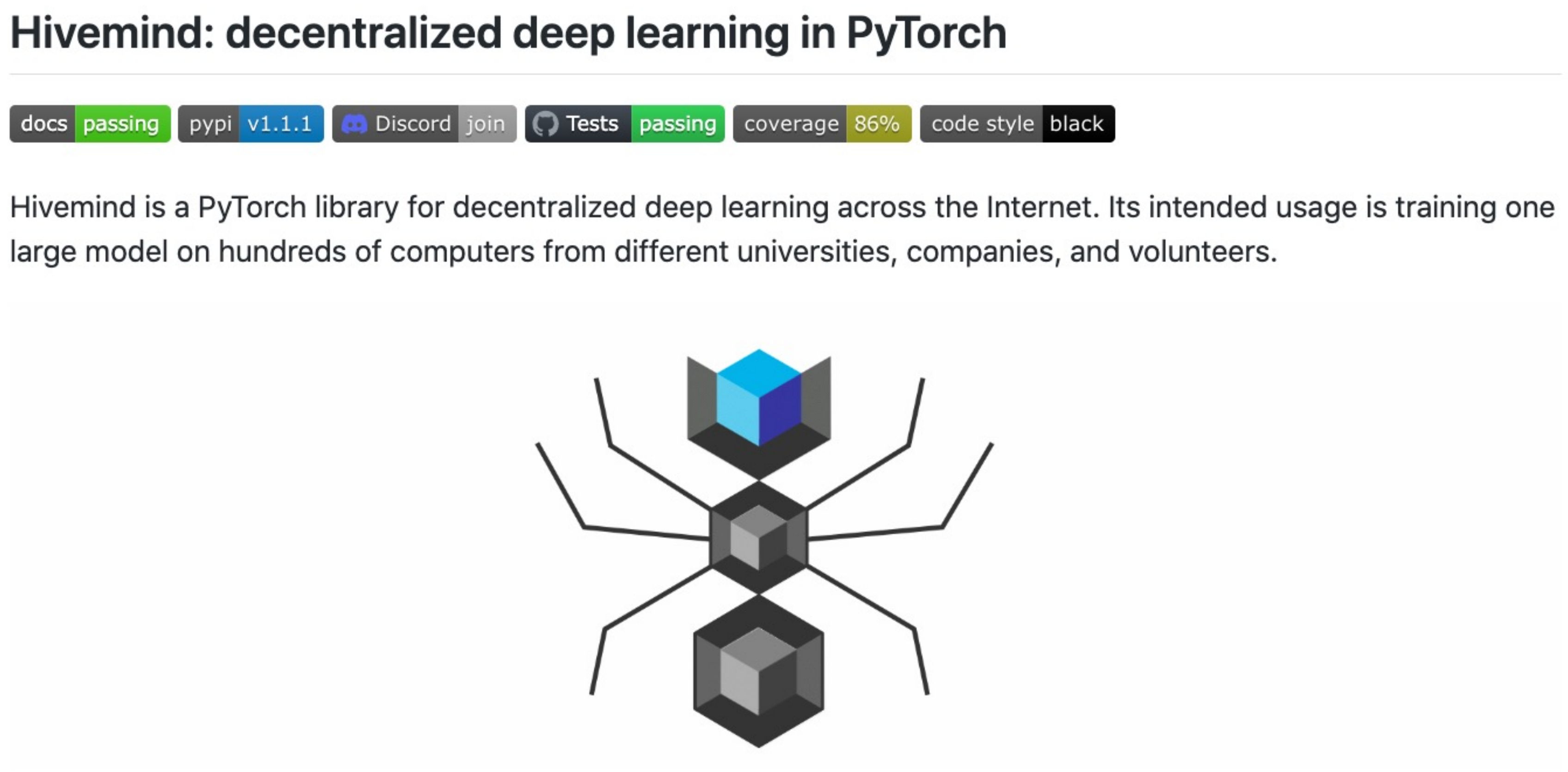
Нейросеть, способная объяснить себе задачу: P-tuning для YaLM
https://habr.com/ru/company/yandex/blog/588214/

Nice! But where to get these models?

- Until mid-2021, large foundation models were mostly closed due to:
  - Ethical concerns (NSFW images, fake news, etc.)
  - Keeping competitive advantage
  - Earning money via proprietary APIs
- Examples: GPT-3, DALL-E, Codex, PALM, etc.

That’s not good!

- Researchers from universities and small companies don’t have access to these models
  - Slows down scientific progress
  - Underrepresented communities
- **Our idea:** Researchers can collaborate to train a large model!



Diskin, Michael, et al. "Distributed deep learning in open collaborations." *Advances in Neural Information Processing Systems* 34 (2021): 7879-7897.

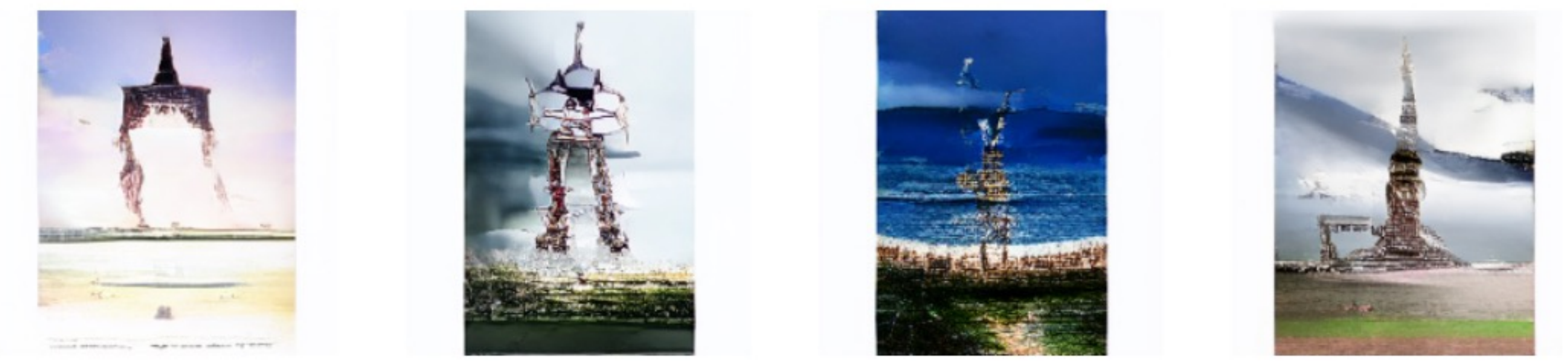
Long story short…

- We did a few projects in 2021 (Bengali and Arabic BERTs, a small DALL-E)
- Wrote papers on what to do with slow communication, unreliable and malicious participants

পথের দেবতা প্রসন্ন হাসিয়া বলেন-মূর্খ বালক, পথ তো
আমার শেষ হয়নি তোমাদের গ্রামে, বাঁশের বনে, ঠ্যাঙাড়ে
বীরু রায়ের বটতলায় কি ধলচিতের খেয়াঘাটের সীমানায়.
তোমাদের সোনাডাঙা মাঠ ছাড়িয়ে ইচ্ছামতী পার হয়ে
পদ্মফুলে ভরা মধূখালি বিলের পাশ কাটিয়া বেত্রবতীর
খেয়ায় পাড়ি দিয়ে, পথ আমার চলে গেল সামনে, সামনে,
শুধুই সামনে...দেশ ছেড়ে দেশান্তরের দিকে, সূর্যোদয় ছেড়ে
সুর্যাস্তের দিকে, জানার গন্ডী এড়িয়ে অপরিচয়ের উদ্দেশে..

Collaborative Arabic Language Model
CALM
نموذج اللغة العربية التعاوني

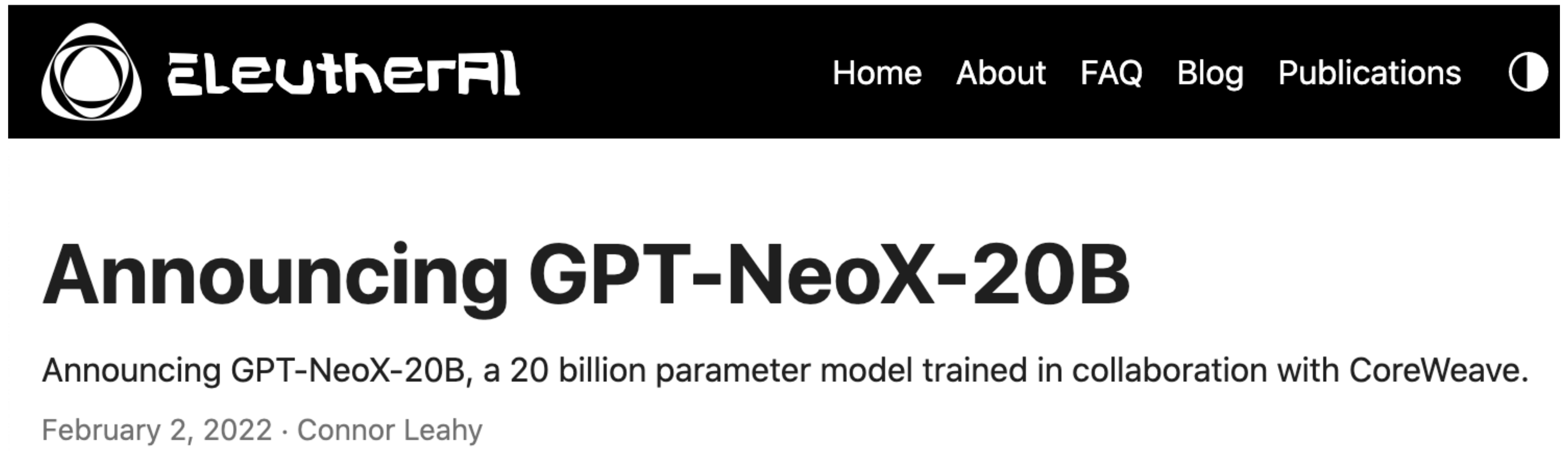
(h) half human half Eiffel tower

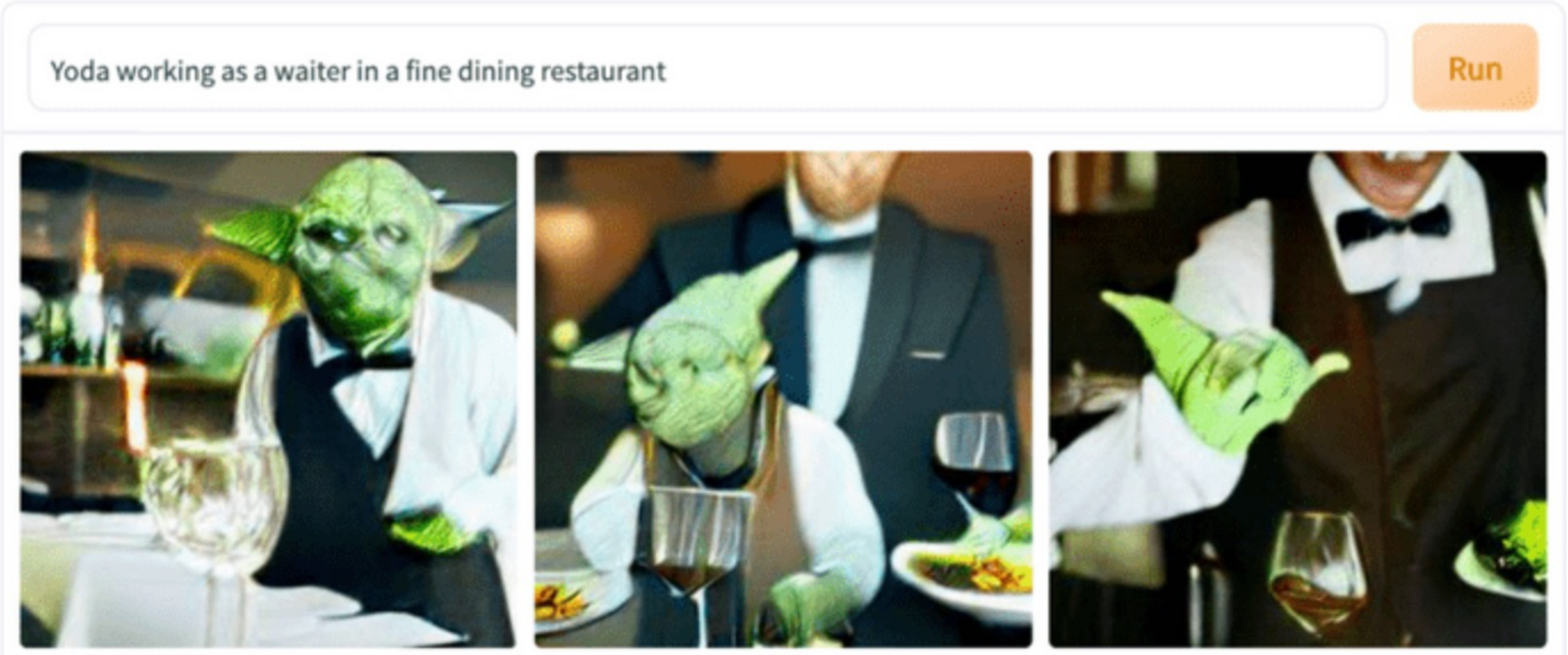


Hivemind: decentralized deep learning in PyTorch, Example Use Cases
https://github.com/learning-at-home/hivemind#example-use-cases

Long story short…

- But people found easier ways to do that
- Lots of foundation models have been published in late 2021 - 2022





Long story short…

- Turns out training new foundation models from scratch is not that relevant anymore
  - Maybe this will change in future

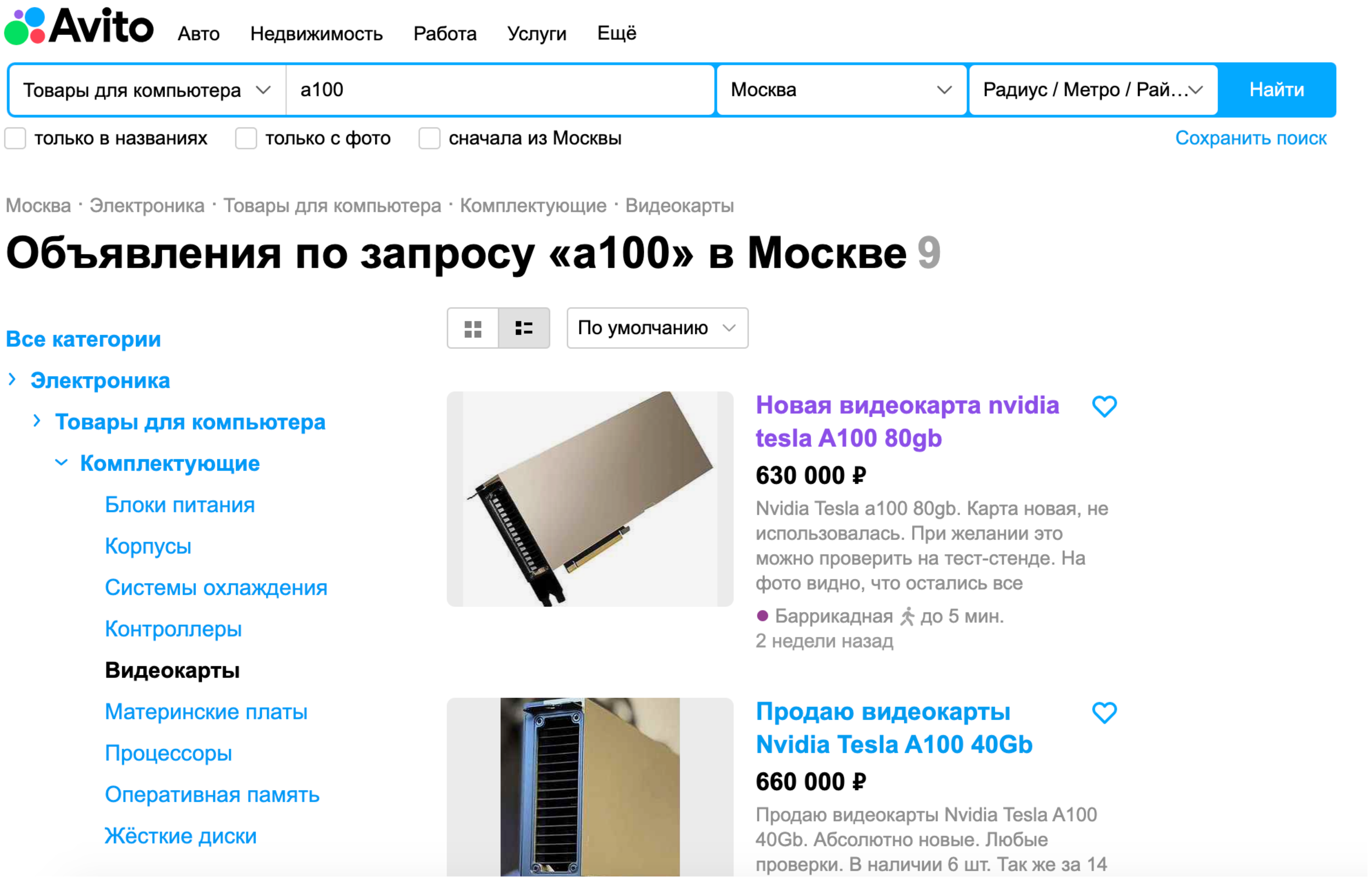
- However, there is another problem!
  - Even if a foundation model has been publicly release, how can people run it?

How to run a publicly released model?

- The models are huge
  - Example: 176B model in float16 requires 352 GiB
- You can compress it to 8-bit
- But you still need several high-end GPUs to fit it
  - For the model above, it’s 3x A100 with 80 GiB each

Dettmers, Tim, et al. "LLM. int8 (): 8-bit Matrix Multiplication for Transformers at Scale." *arXiv preprint arXiv:2208.07339* (2022).

How to run a publicly released model?



Introducing Petals

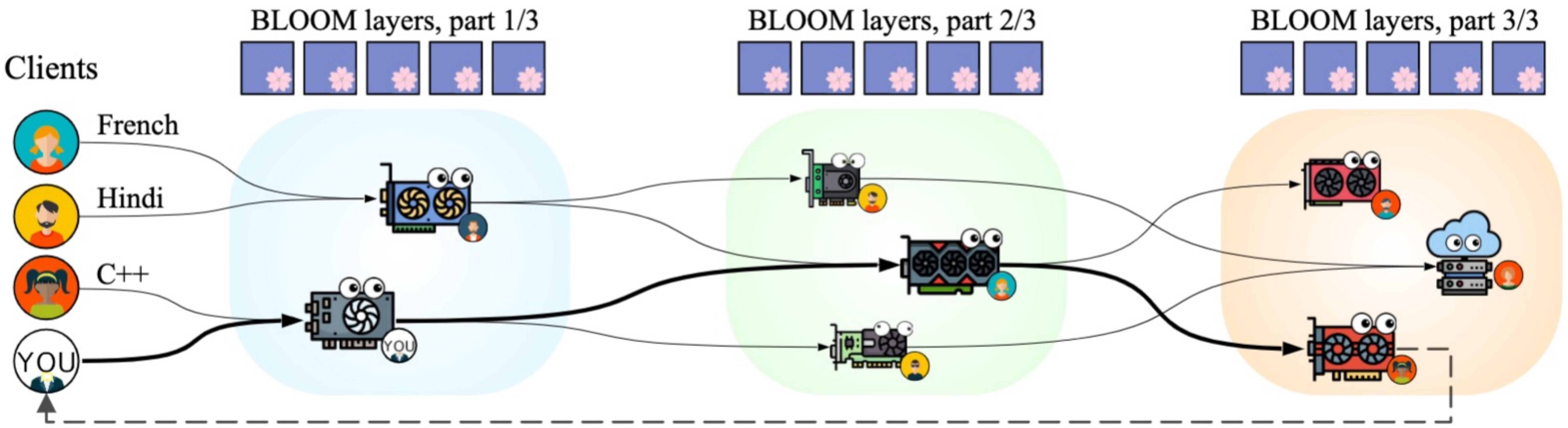


Figure 1: An overview of PETALS. Some participants (*clients*) want to use a pretrained language model to solve various tasks involving processing texts in natural (e.g., French, Hindi) or programming (e.g., C++) languages. They do it with help of other participants (*servers*), who hold various subsets of model layers on their GPUs. Each client chooses a sequence of servers so that it performs an inference or fine-tuning step in the least amount of time.

Borzunov, Alexander, et al. "Petals: Collaborative Inference and Fine-tuning of Large Models." *arXiv preprint arXiv:2209.01188* (2022).

Surprise 1: Inference is fast

- In large LMs, **hidden states** are much smaller than transformer block's **parameters**
- Internet is slower than GPU bus
- However, sending small hidden states over the Internet is **~10x faster** than sending large block parameters over GPU bus

Surprise 2: Fine-tuning is possible, no need for changes on servers

- Actually, you can impelement **almost any** parameter-efficient fine-tuning or sampling method
- Each client can store trained parameters locally
- You can see internal states & probabilities for your research

How to make it efficient?

- Compressing communication buffers
  - Block-wise quantization

- Compressing model weights
  - 8-bit mixed decomposition
  - This decomposition separates hidden states and weights into two portions: about 0.1% of 16-bit outlier and 99.9% of 8-bit regular values

Dettmers, Tim, et al. "8-bit Optimizers via Block-wise Quantization." *arXiv preprint arXiv:2110.02861* (2021).

Dettmers, Tim, et al. "LLM. int8 (): 8-bit Matrix Multiplication for Transformers at Scale." *arXiv preprint arXiv:2208.07339* (2022).

How to make it efficient?

- Server-side load balancing
- Client-side routing

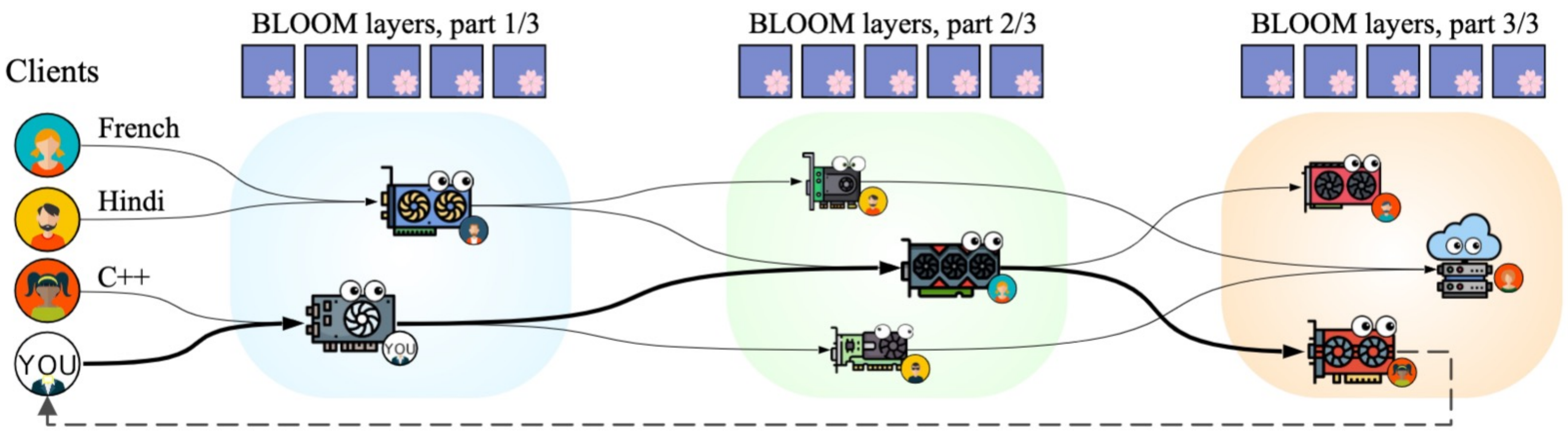


Figure 1: An overview of PETALS. Some participants (*clients*) want to use a pretrained language model to solve various tasks involving processing texts in natural (e.g., French, Hindi) or programming (e.g., C++) languages. They do it with help of other participants (*servers*), who hold various subsets of model layers on their GPUs. Each client chooses a sequence of servers so that it performs an inference or fine-tuning step in the least amount of time.

Benchmarks

Table 3: Performance of sequential inference steps and training-time forward passes.

Network		Inference (steps/s)		Forward (tokens/s)	
		Sequence length		Batch size	
Bandwidth	**Latency**	128	2048	1	64
Offloading, max. speed on 1x A100					
256 Gbit/s	–	0.18	0.18	2.7	170.3
128 Gbit/s	–	0.09	0.09	2.4	152.8
Offloading, max. speed on 3x A100					
256 Gbit/s	–	0.09	0.09	5.1	325.1
128 Gbit/s	–	0.05	0.05	3.5	226.3
PETALS on 3 physical servers, with one A100 each					
1 Gbit/s	< 5 ms	1.22	1.11	70.0	253.6
100 Mbit/s	< 5 ms	1.19	1.08	56.4	182.0
100 Mbit/s	100 ms	0.89	0.8	19.7	112.2
PETALS on 12 virtual servers, simulated on 3x A100					
1 Gbit/s	< 5 ms	0.97	0.86	37.9	180.0
100 Mbit/s	< 5 ms	0.97	0.86	25.6	66.6
100 Mbit/s	100 ms	0.44	0.41	5.8	44.3
PETALS on 14 real servers in Europe and North America					
Real world		0.68	0.61	32.6	179.4

Current status and future work

- Alpha version of the code is released
- Public swarm will be open in November

Future work

- Incentives
  - Demand/supply imbalance
  - Bloom points can be spent on high-priority inference or other rewards
  - Centralized/decentralized
- Security
- Privacy

Thank you! Questions?

Petals

Decentralized platform for running 100B+ language models

https://petals.ml