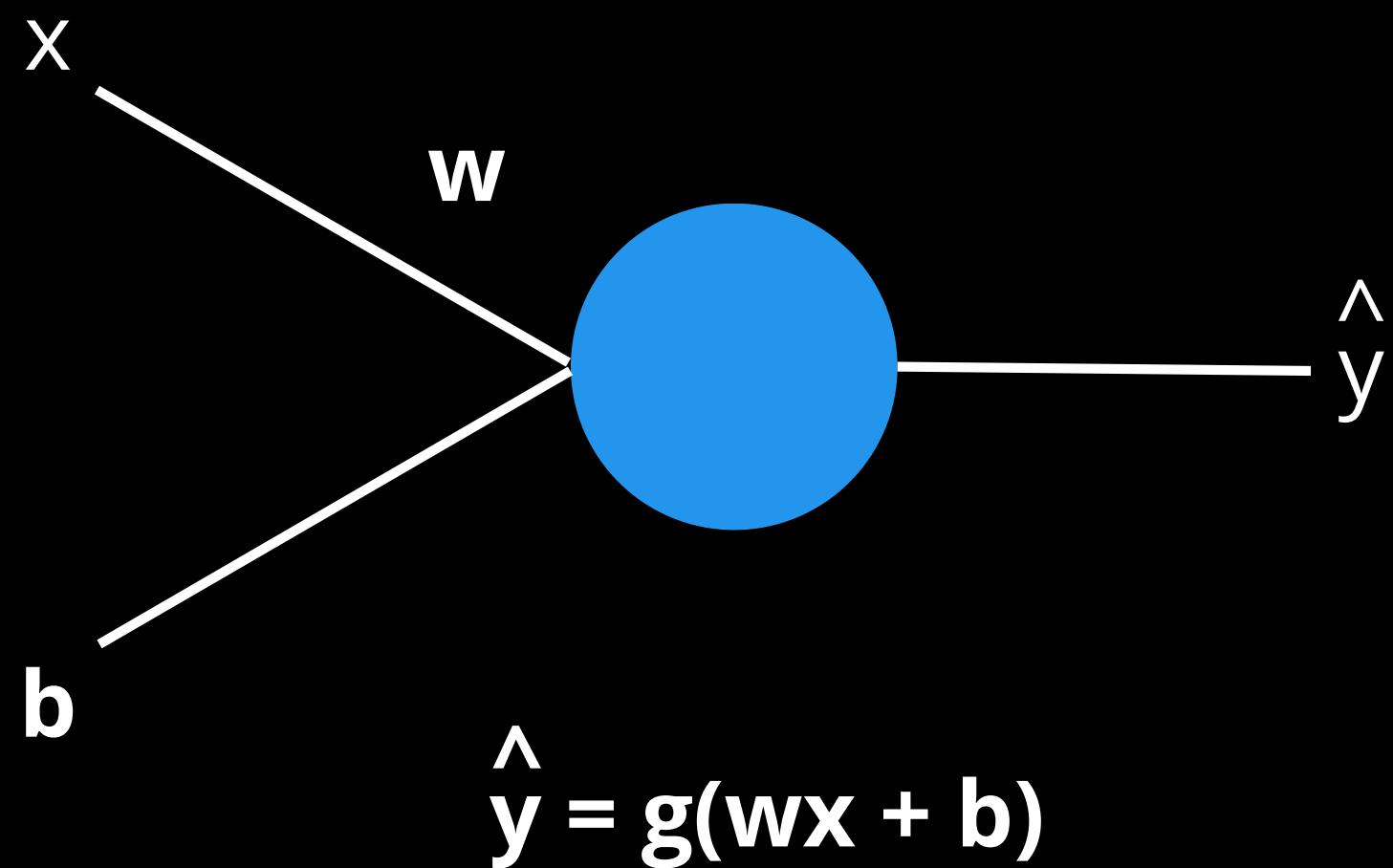


Fundamentos de **vLLM**: Virtual Large Language Model

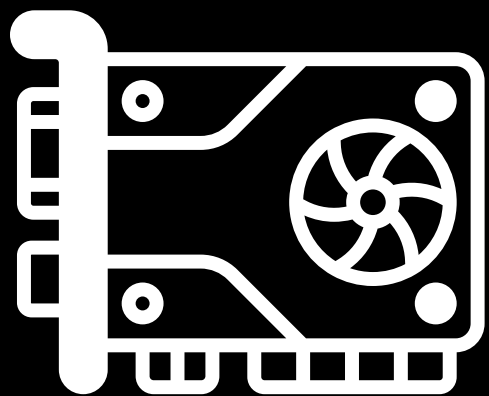


Motivação: Redes Neurais

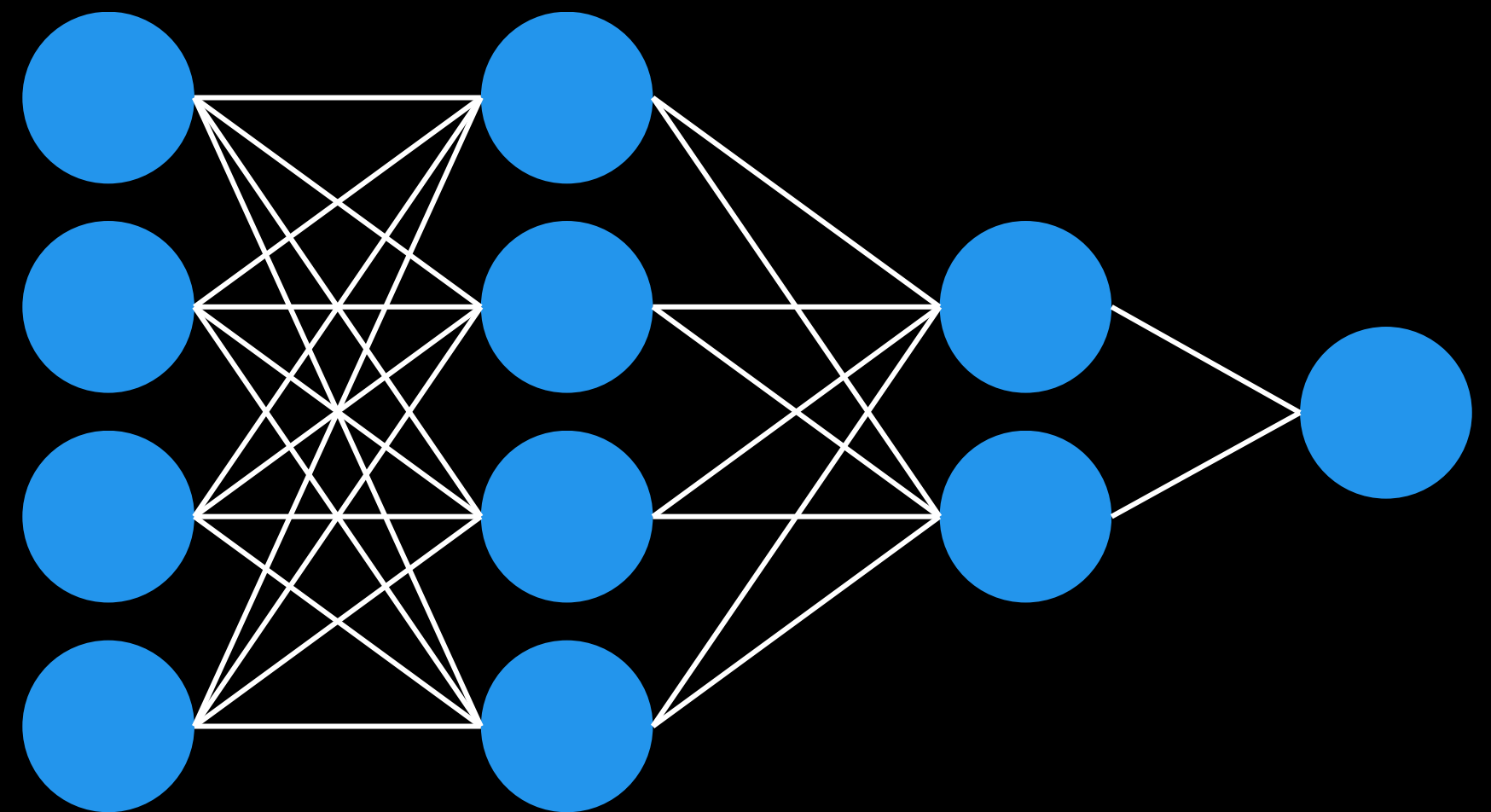


Motivação: Redes Neurais

- $4 \times 4 + 4 \times 2 + 2 \times 1 = 26$ parâmetros (MLP sem considerar pesos da camada de input)
- Para redes maiores, GPUs facilitam os cálculos matriciais para acelerar inferência em treinamento/produção

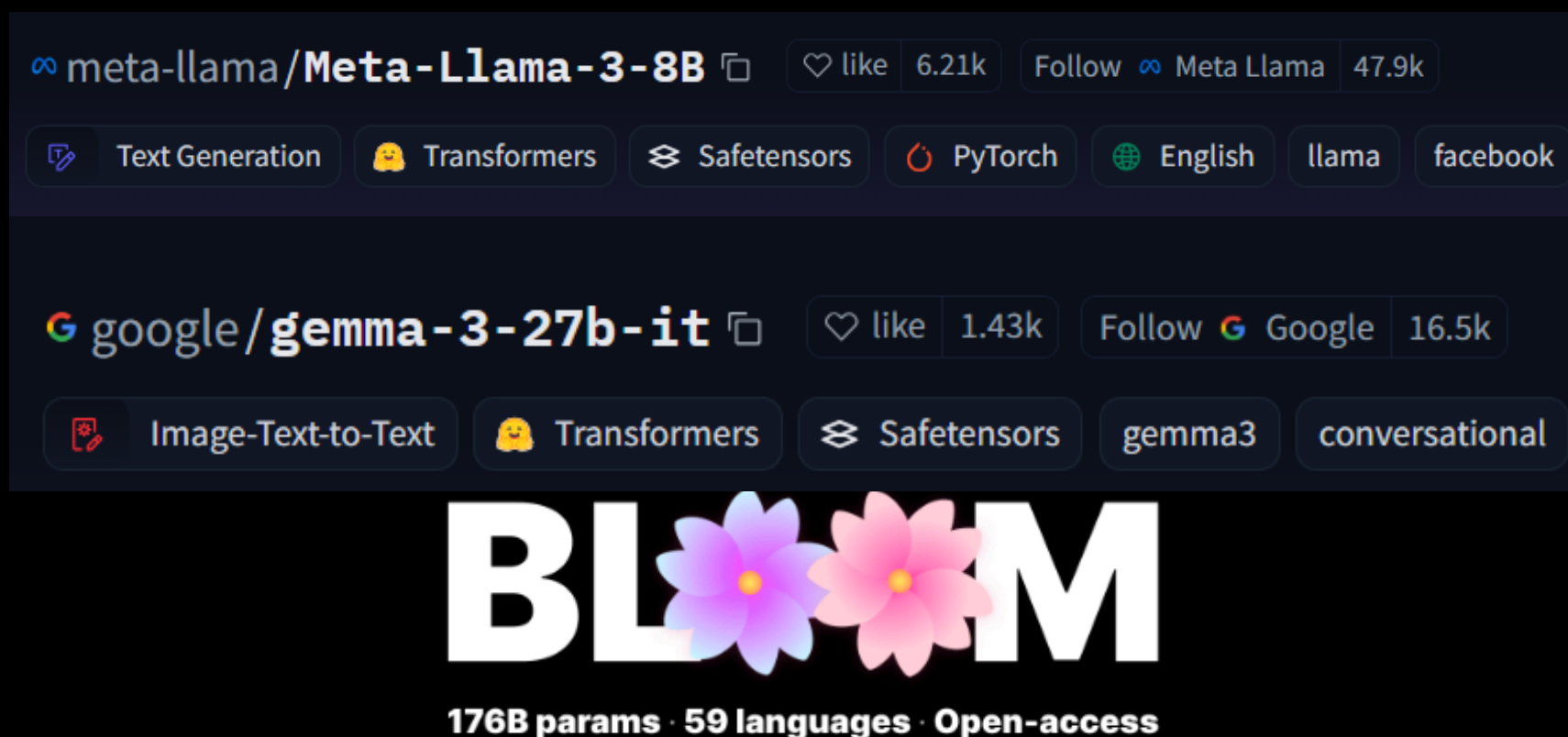


obs) #parâmetros != #neurônios



Neural Network

Motivação: Modelos Grandes



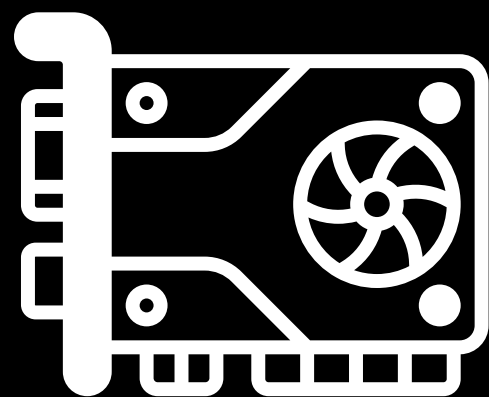
As these LLMs get bigger and more complex, their capabilities will improve. We know that ChatGPT-4 has in the region of 1 trillion parameters (although OpenAI won't confirm,) up from 175 billion in ChatGPT 3.5—a parameter being a mathematical relationship linking words through numbers and algorithms. That's a vast leap in terms of understanding relationships between words and knowing how to stitch them together to create a response.

What are Gemini 1.5 Pro and GPT-4o?


Gemini 1.5 Pro, developed by Celestial AI, is a state-of-the-art language model released in 2023. It boasts an impressive 1.5 trillion parameters, making it one of the largest models available.

Motivação: Preço

GPU	Preço Mensal Estimado GCP
NVIDIA T4	US\$ 262,00
NVIDIA P100	US\$ 1817,00



Entre em contato com nossa equipe

 Compute Engine ⓘ \$ 29.602,78 / month

Spot (Preemptible VM)

Machine type* ⓘ

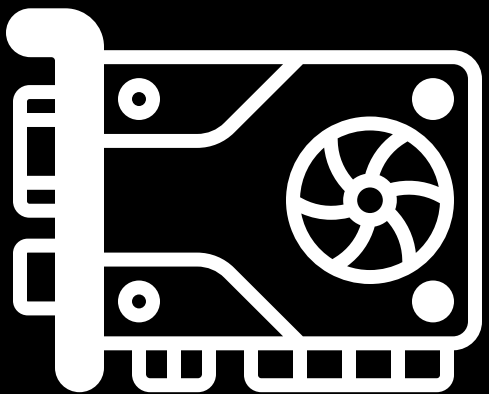
Machine Family*
Accelerator-optimized ▼

Series*
A2 ▼

Machine type*
a2-ultragpu-8g ▼

Motivação: Etapas de Geração

- Aumentar o tamanho ou a quantidade de GPUs resolve todos os problemas?
- Problemas com parte paralela e serial da geração
- Parte serial limitada pela necessidade de fornecer contexto dos tokens que o próprio modelo gerou



Motivação: Transformers

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

Motivação: Embeddings Posicionais

“Eu vou ao **banco** para sacar dinheiro”

“Eu sentei no **banco** para descansar”

- As palavras que vem antes de “banco” definem seu significado semântico e sua probabilidade de ser gerada, além das possibilidades de geração das próximas palavras

“O **lobo** caça o **homem**”

“O **homem** caça o **lobo**”

Motivação: QKV e Head Attentions

- Essa correlação é calculada durante a etapa de inferencia utilizando vetores de
 - Query (Q)
 - Key (K)
 - Value (V)
- Essa correlação compõe o que chamamos de cabeças de atenção (Head Attentions)

Eu	Q1,K1	Q1,K2	Q1,K3	Q1,K4
vou	Q2,K1	Q2,K2	Q2,K3	Q2,K4
ao	Q3,K1	Q3,K2	Q3,K3	Q3,K4
banco	Q4,K1	Q4,K2	Q4,K3	Q4,K4
	Eu	vou	ao	banco

Motivação: Etapas de Geração

Prompt de entrada

“Escreva uma sentença que possa ser usada como frase de um personagem de um livro que ama gatos de estimação”

Estapa passível de paralelização (todos os tokens de entrada são conhecidos para gerar embeddings posicionais/cálculo de matriz de atenção)

Motivação: Autoregressivo

Geração de Output: Etapa não paralelizável



Os tokens de toda a sentença resultante não são conhecidos porque estão sendo gerados

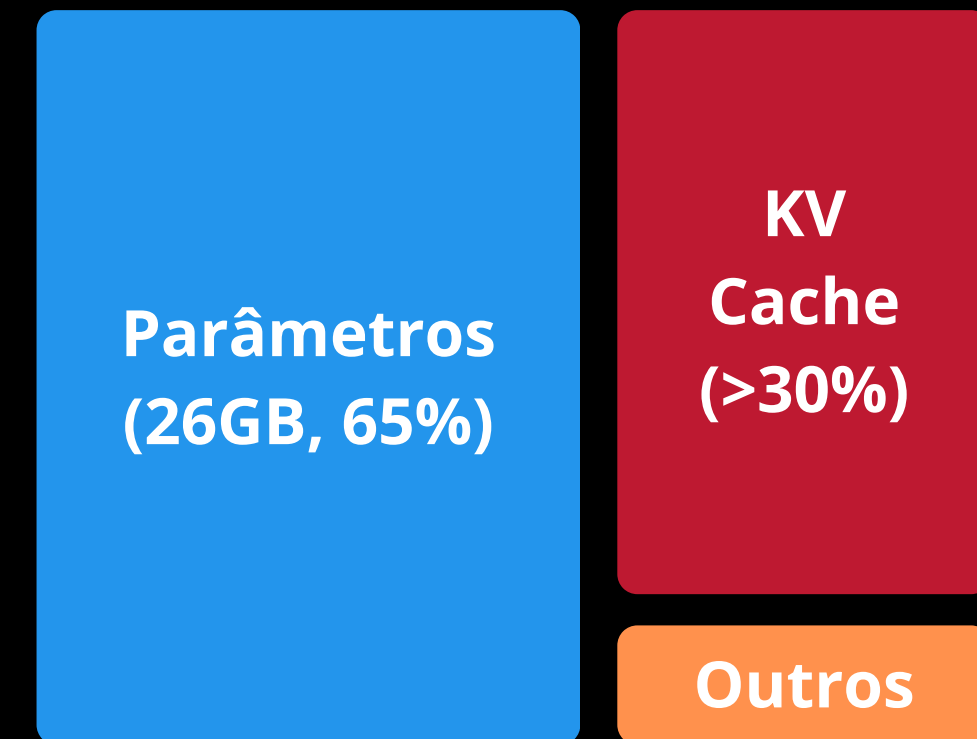
Motivação: KV Cache

- Não há necessidade de calcular toda a matriz novamente toda vez que um token é gerado
- Matriz é cacheada para ser utilizada nas próximas iterações
- Qs mudam, mas os K e Vs são fixos para as sentenças, por isso podem ser armazenados

Eu	Q1,K1	Q1,K2	Q1,K3	Q1,K4
vou	Q2,K1	Q2,K2	Q2,K3	Q2,K4
ao	Q3,K1	Q3,K2	Q3,K3	Q3,K4
banco	Q4,K1	Q4,K2	Q4,K3	Q4,K4
	Eu	vou	ao	banco

Motivação: KV Cache

- Mais da metade dos recursos da GPU alocadas apenas para os parâmetros (estáticos)
- Uma grande parte alocada para KV cache de inferencia
- Espaço do KV cache precisa ser otimizado ao máximo
- **Deep Learning tradicional não lida bem com um cache que tem tamanho alterado dinamicamente (acostumados com tamanhos fixos de entradas e saídas)**



NVIDIA A100 40GB
Modelo 13B



Efficient Memory Management for Large Language Model Serving with *PagedAttention*

Woosuk Kwon^{1,*} Zhuohan Li^{1,*} Siyuan Zhuang¹ Ying Sheng^{1,2} Lianmin Zheng¹ Cody Hao Yu³
Joseph E. Gonzalez¹ Hao Zhang⁴ Ion Stoica¹

¹UC Berkeley ²Stanford University ³Independent Researcher ⁴UC San Diego

Abstract

High throughput serving of large language models (LLMs) requires batching sufficiently many requests at a time. However, existing systems struggle because the key-value cache (KV cache) memory for each request is huge and grows and shrinks dynamically. When managed inefficiently, this memory can be significantly wasted by fragmentation and redundant duplication, limiting the batch size. To address this problem, we propose PagedAttention, an attention algorithm inspired by the classical virtual memory and paging techniques in operating systems. On top of it, we build vLLM, an LLM serving system that achieves (1) near-zero waste in KV cache memory and (2) flexible sharing of KV cache within and across requests to further reduce memory usage. Our evaluations show that vLLM improves the throughput of popular LLMs by 2-4× with the same level of latency compared to the state-of-the-art systems, such as FasterTransformer and Orca. The improvement is more pronounced with longer sequences, larger models, and more complex decoding algorithms. vLLM’s source code is publicly available at <https://github.com/vllm-project/vllm>.

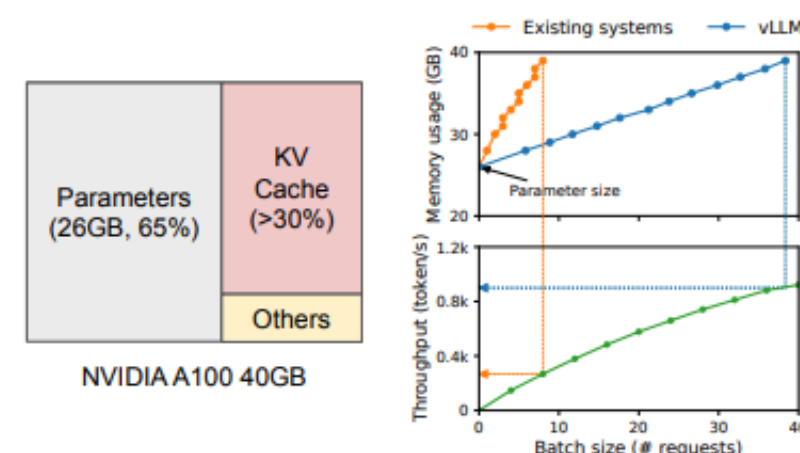


Figure 1. *Left:* Memory layout when serving an LLM with 13B parameters on NVIDIA A100. The parameters (gray) persist in GPU memory throughout serving. The memory for the KV cache (red) is (de)allocated per serving request. A small amount of memory (yellow) is used ephemerally for activation. *Right:* vLLM smooths out the rapid growth curve of KV cache memory seen in existing systems [31, 60], leading to a notable boost in serving throughput.

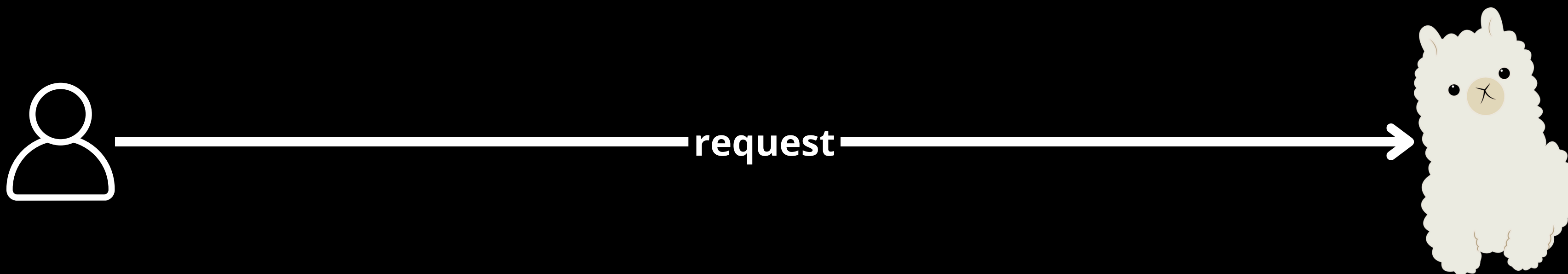
the cost per request—of *LLM serving* systems is becoming more important.

Motivação: Objetivo

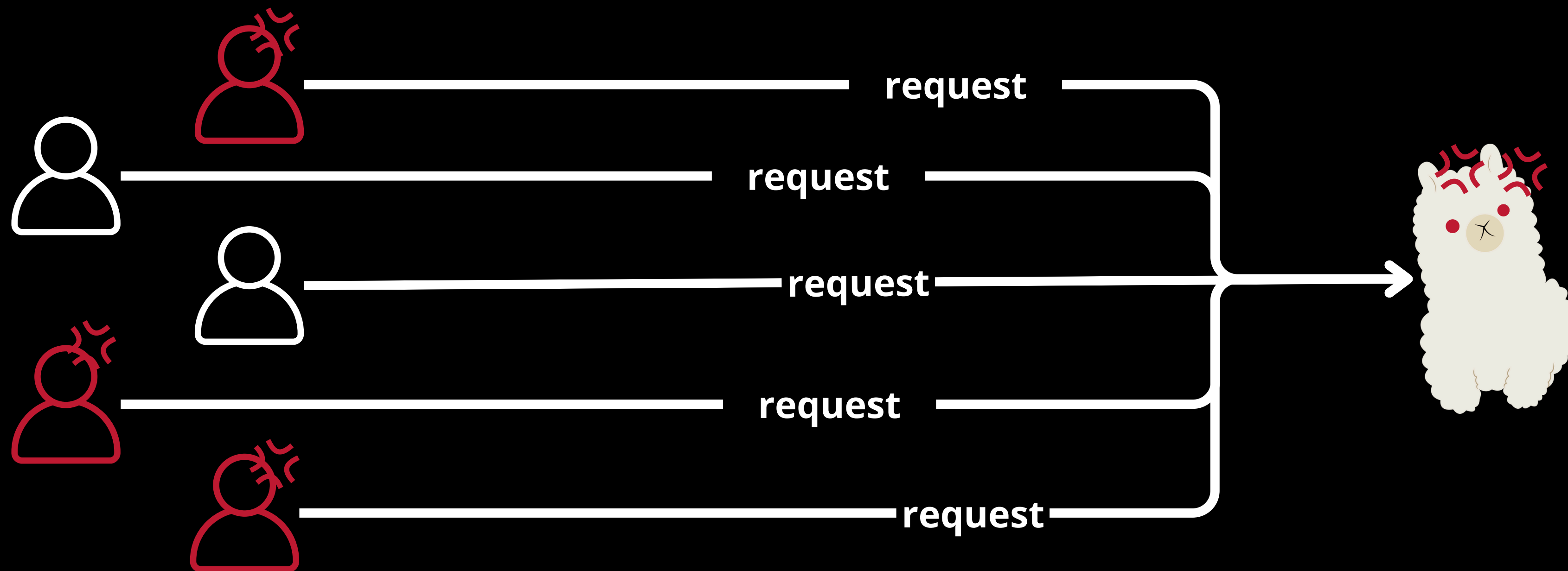
- Nosso objetivo é criar aplicações que não performam bem apenas para um usuário (o que o KV cache sozinho resolveria)
- Recursos computacionais precisam ser bem aproveitados para escalar soluções para um grande número de usuários



Motivação

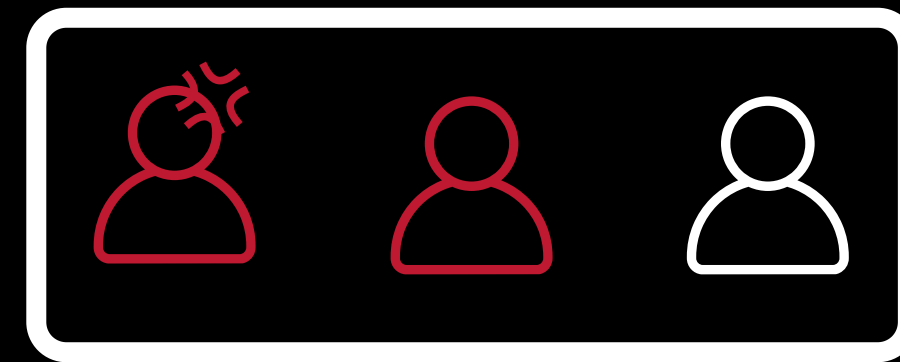


Motivação



Motivação: Processar em Fila

- Poderíamos evitar uma excedência de memória através do enfileiramento das requisições que chegam ao modelo
- Criaria um gargalo para usuários que esperam respostas em tempo real
- Característica que depende da aplicação, mas muitas delas requerem tempo real
- Exemplo do modelo DALL-E da OpenAI ou outros generativos

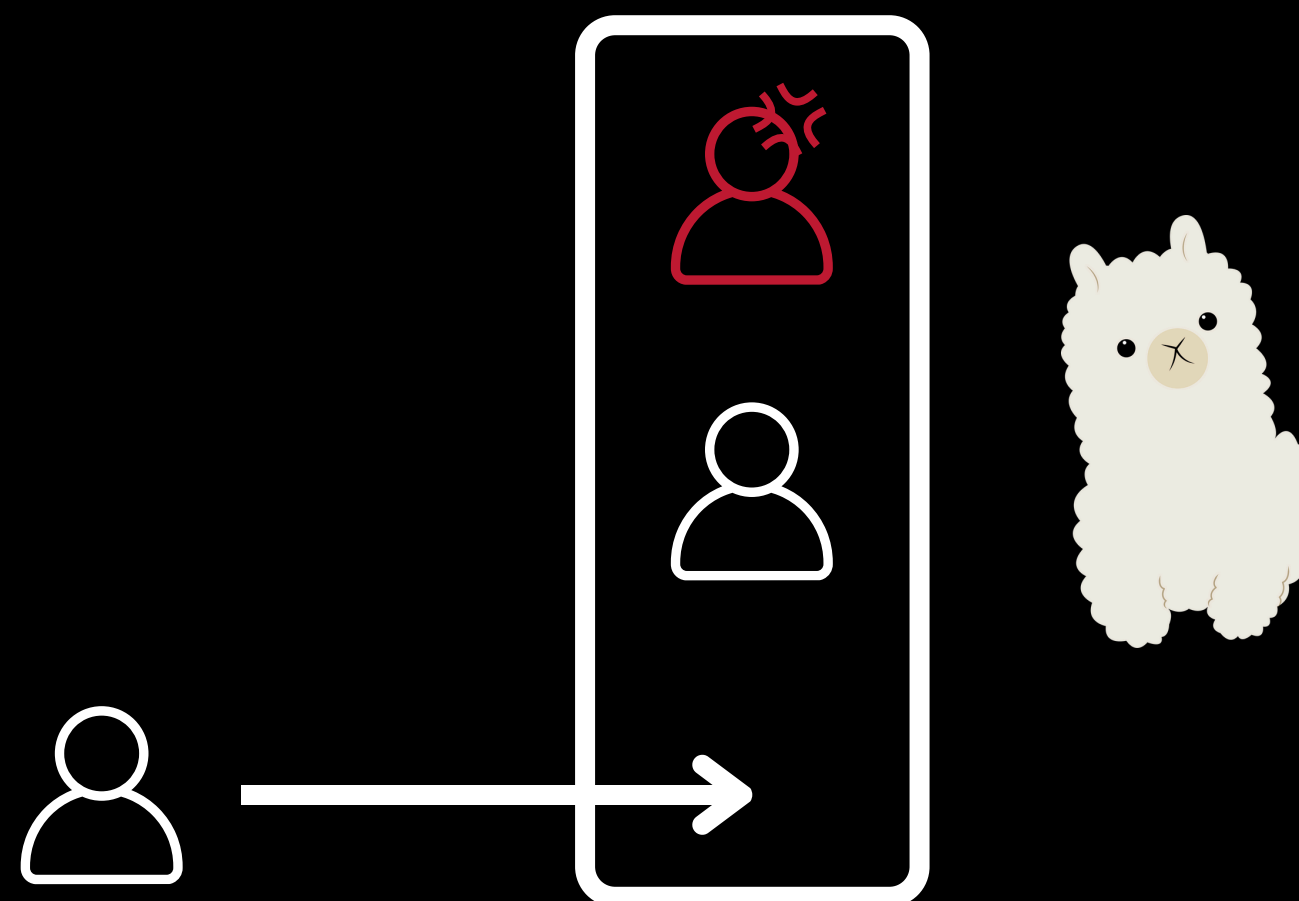


“Apache”



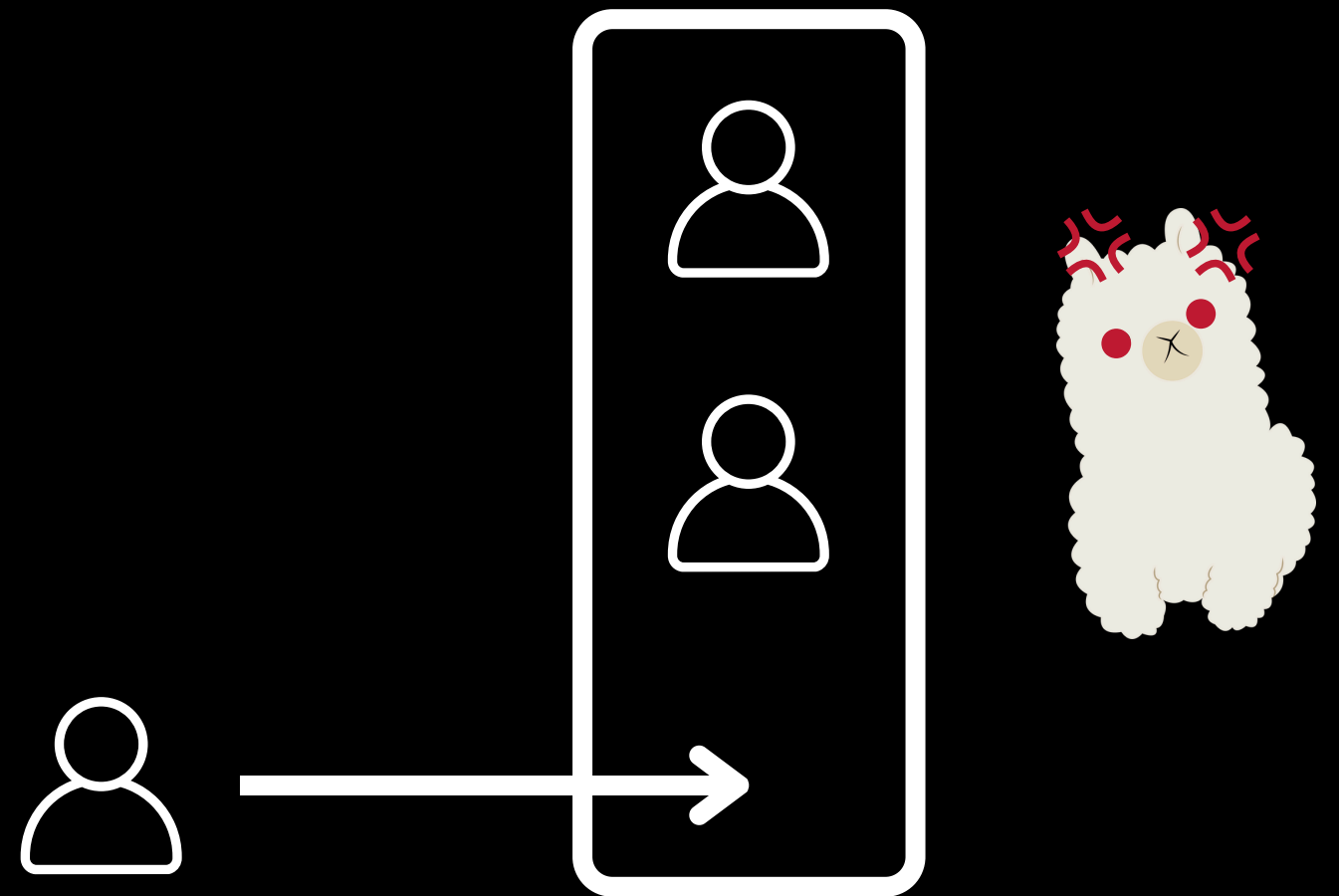
Motivação: Batching de Requisições

- **Processamento de várias requests devem ser feitos em batch**
- Requisições chegam em momentos diferentes
- Requisições possuem tamanhos diferentes para serem alocados



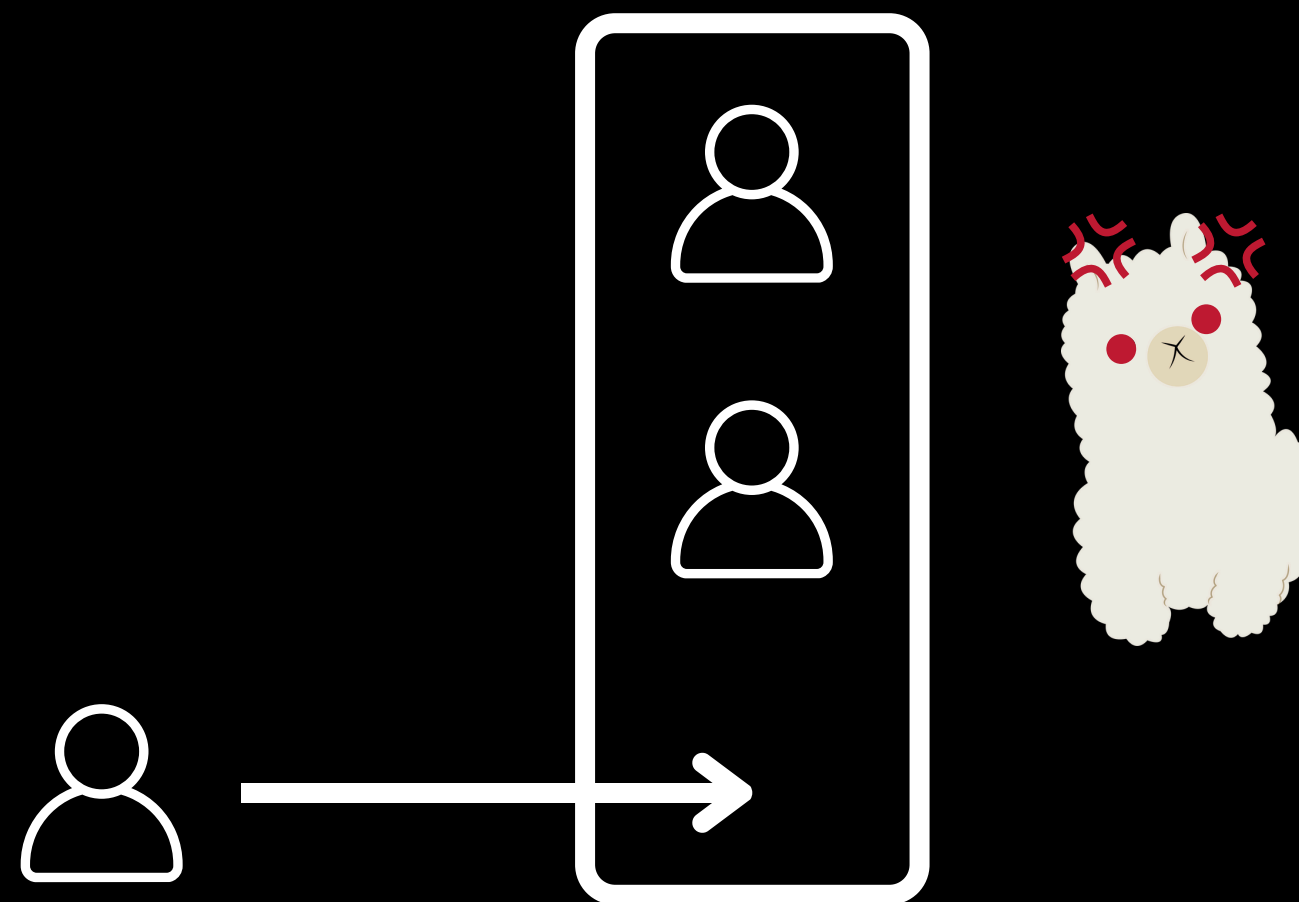
Motivação: Batching por Iteração

- Diminuir granularidade de inferência para iteração
- Não preciso esperar batching completo para iniciar
- Trocas de requisições no batch podem ocorrer quando uma requisição termina antes

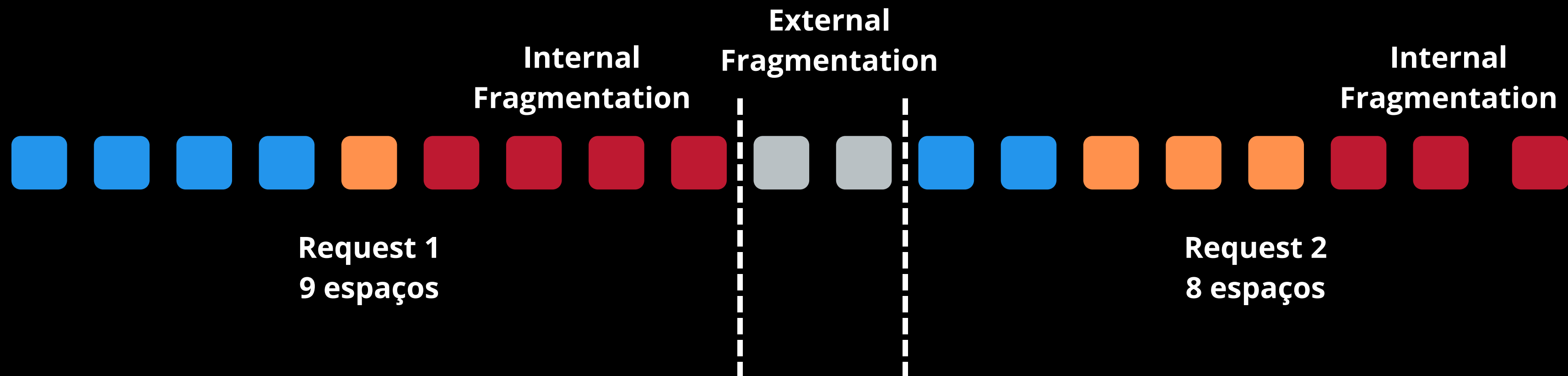


Motivação: Métodos “Tradicionais”

- O problema é que não sabemos qual o tamanho final da sentença antes de ela ser gerada
- Aloca-se todo o espaço do tamanho máximo que ela pode ter em um espaço contíguo na memória



Motivação: Métodos “Tradicionais”



■ Espaço não reservado e não usado

■ Input

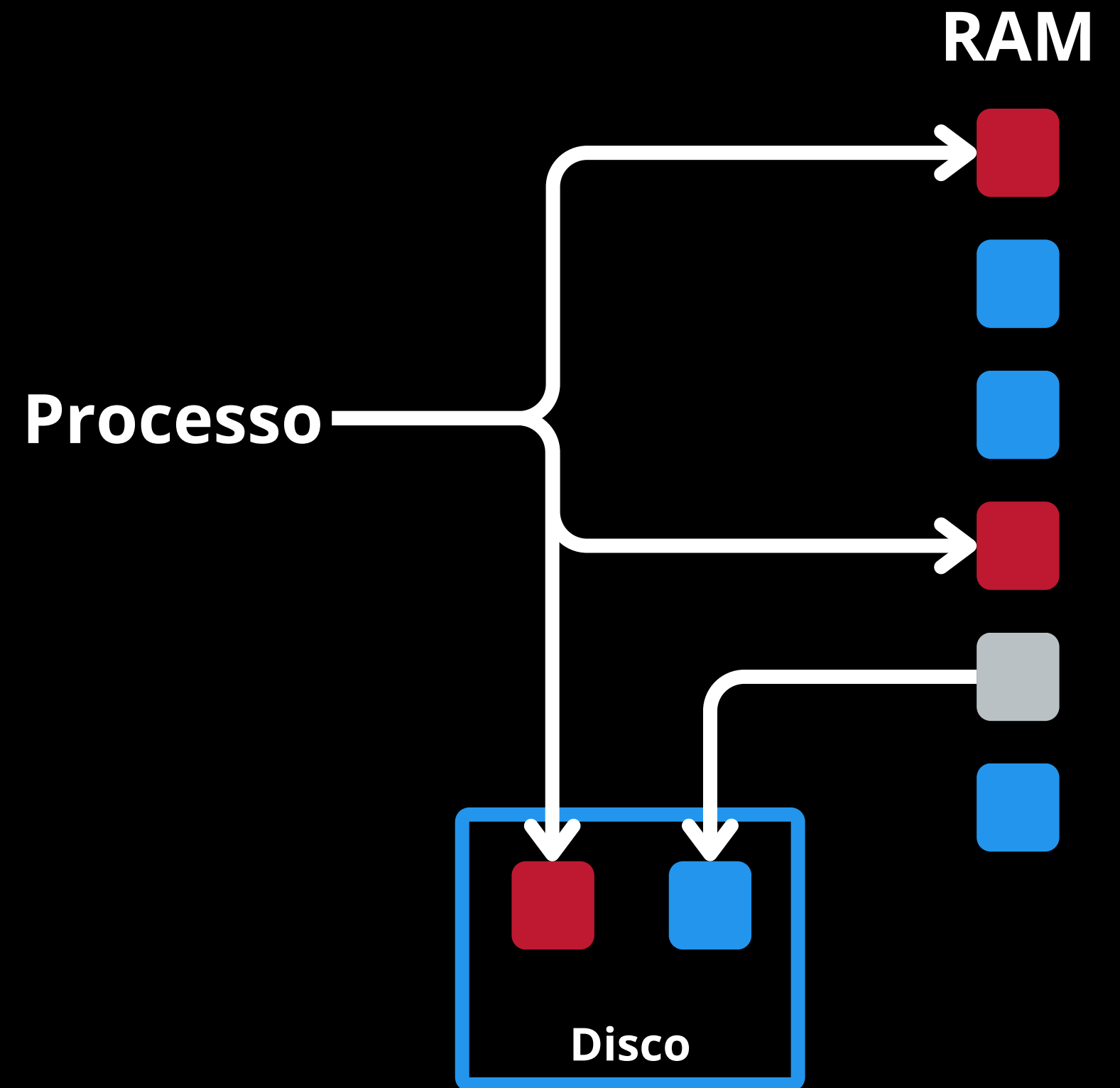
■ Output

■ Espaço reservado não usado

Ineficiência no uso de espaços contíguos

Paged Memory

- Inspiração na forma com que sistemas operacionais orquestram memória RAM
- Criação de blocos de memória não contíguos para um mesmo processo
- Possibilidade de criar uma RAM virtual presente no disco



PagedAttention

- Divide o KV cache da requisição em blocos, cada um podendo conter as K e V de um número fixo de tokens
- Blocos não armazenados em espaço contíguo
- “Blocos são páginas, tokens são bytes e requests são processos”

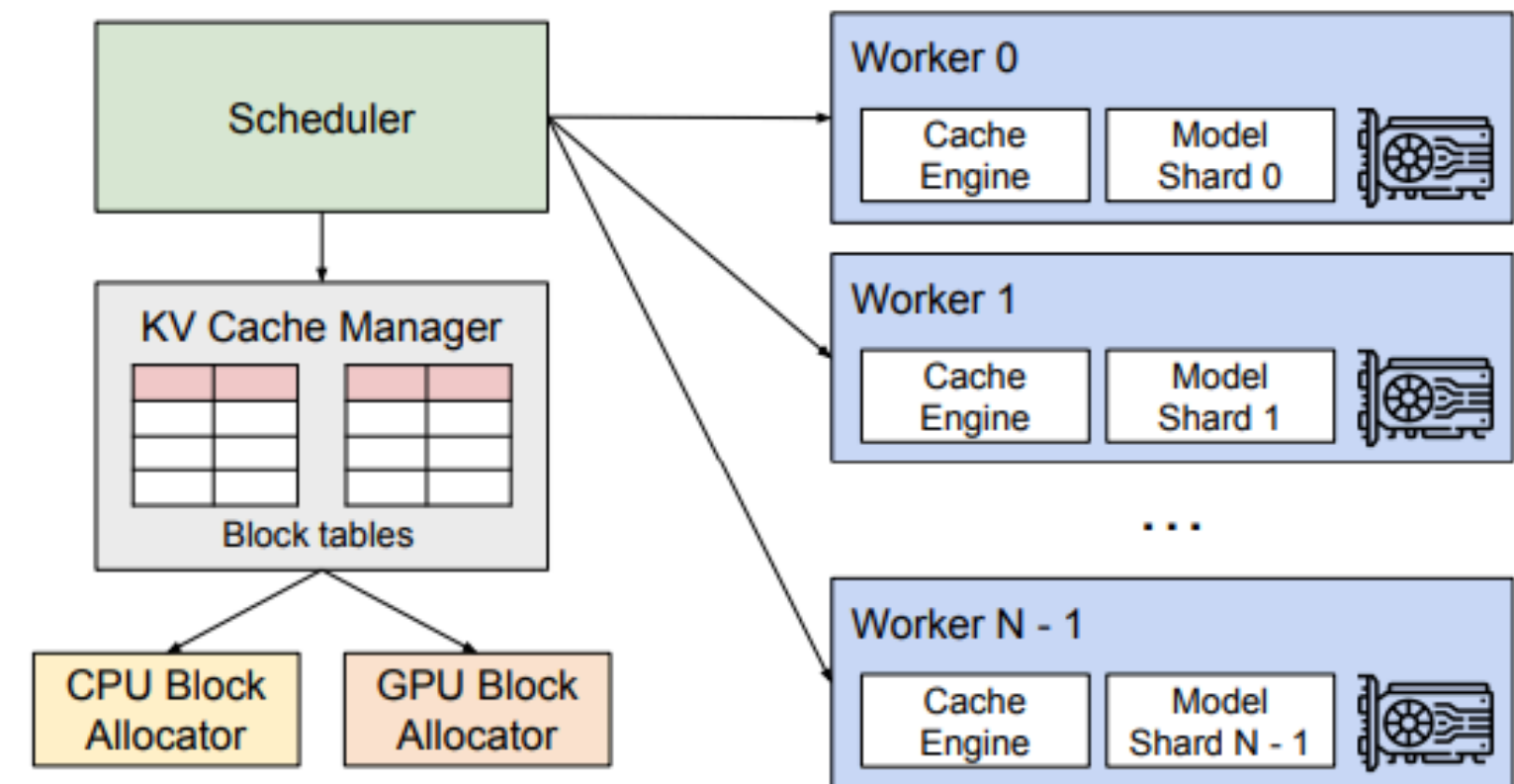


Figure 4. vLLM system overview.

PagedAttention

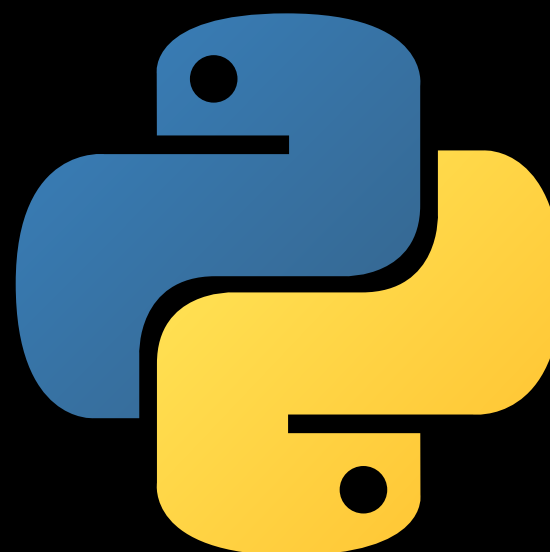
- Espaço passa a ser alocado dinamicamente, o que evita espaços ociosos da GPU
 - Permite compartilhamento de KV cache entre requests, principalmente em sistemas com system prompt
 - Implementações de códigos de mais baixo nível (kernel) otimizam a transferência de dados para acelerar inferência
-



- Aplicação end-to-end



**Kernels e
PagedAttention**



**Schedulers e
Blocos**



**API para
Inferência**

vLLM: Package

```
1  import streamlit as st
2  import os
3  from PyPDF2 import PdfReader
4  import chromadb
5  from vllm import LLM
6  import numpy as np
7
8  llm = LLM(model="meta-llama/Llama-2-7b-chat-hf", gpu_memory_utilization=0.9, max_model_len=752)
9
10 response = llm.generate("Oi, pode me ajudar?")
```

vLLM: CLI



```
vllm serve meta-llama/Llama-2-7b-chat-hf \
--multi-step-stream-outputs \
--dtype 'float16' \
--device 'cpu' \
--max-num-batched-tokens 2048
```

<http://localhost:8000>

```
lucas@WIN-D6K6FFNT79B MINGW64 ~
$ curl https://bc5e-34-16-203-104.ngrok-free.app/v1/completions \
  -H "Content-Type: application/json" \
  -d '{
    "model": "meta-llama/Llama-2-7b-chat-hf",
    "prompt": "San Francisco is a",
    "max_tokens": 7,
    "temperature": 0
  }'
{"id": "cmpl-2596b3d406a64556a2257e62b26ab0dc", "object": "text_completion", "created": 1746500528, "model": "meta-llama/Llama-2-7b-chat-hf", "choices": [{"index": 0, "text": " city in Northern California that is known", "logprobs": null, "finish_reason": "length", "stop_reason": null, "prompt_logprobs": null}], "usage": {"prompt_tokens": 5, "total_tokens": 12, "completion_tokens": 7}, "prompt_tokens_details": null}}
lucas@WIN-D6K6FFNT79B MINGW64 ~
$ |
```

Referências

- <https://arxiv.org/pdf/2309.06180>
 - https://docs.vllm.ai/en/v0.7.3/getting_started/installation/cpu/index.html
 - <https://github.com/vllm-project/vllm>
 - <https://wiki.inf.ufpr.br/maziero/lib/exe/fetch.php?media=socm%3Asocm-slides-17.pdf>
 - <https://medium.com/%40javaid.nabi/efficient-generative-large-language-model-serving-1c22b58f3c92>
 - <https://medium.com/%40neltac33/gemini-1-5-pro-vs-gpt-4o-a-head-to-head-showdown-29c4cc837e7b>
 - https://www.wired.com/story/how-chatgpt-works-large-language-model/?utm_source=chatgpt.com
-