Large Language Models In 5 Formulas

Sasha Rush

@srush_nlp

You've probably heard a lot about langauge models

Open-source generative AI startup Mistral AI raise: funding

BY MIKE WHEATLEY

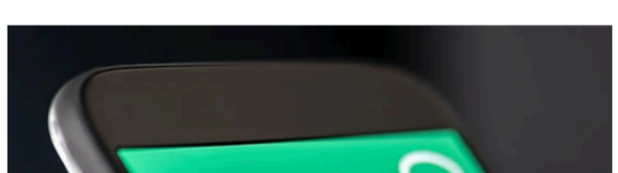
French generative artificial intelligence startup Mistral AI announced today it has clo bumper funding round of the year, raising 385 million euros, or around $415 million, prominent venture capital firms Andreessen Horowitz and Lightspeed Venture Partn

NEWS | 12 January 2024

Google AI has better bedside manner than human doctors — makes better diagnoses

Researchers say their artificial-intelligence system could help to democrati

By Mariana Lenharo

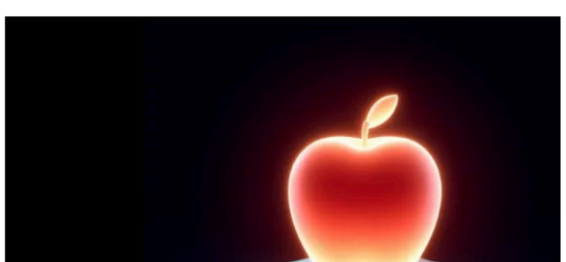


Apple research paper hints at LLMs on iPhones and Macs

By Ben Dickson - December 27, 2023

Me gusta 745

Facebook Twitter Reddit Linkedin



Article | Open access | Published: 20 December 2023

Autonomous chemical research with l models

Daniil A. Boiko, Robert MacKnight, Ben Kline & Gabe Gomes

Nature **624**, 570–578 (2023) | Cite this article

88k Accesses | **2** Citations | **931** Altmetric | Metrics

NEWS FEATURE | 15 November 2023

ChatGPT has entered the classroom how LLMs could transform education

Researchers, educators and companies are experimenting with ways to turn flawed bu famous large language models into trustworthy, accurate 'thought partners' for

NEWS | 14 September 2023

Why Japan is building its own version of ChatGPT

Some Japanese researchers feel that AI systems trained on foreign languages cannot grasp the intricacies of Japanese language and culture.

Write an intro to BC calculus question
like you are barber cutting my hair?

Alright, let's trim this up while we dive into a little brain teaser from BC Calculus. Imagine this:
As I'm snipping away, think of each strand of hair as a tiny segment of a curve. Now, suppose this curve represents a function, $f(x)$, and it's as smooth as your new haircut is going to be.
Your task is to find the derivative of this function at a specific point, say x equals 2. Think of it like pinpointing the exact angle of my scissors as they cut through a particular hair. This derivative, it's not just a number, it's like understanding the exact slope of your hair at that point.
How would you approach it?
Keep this in mind while you hear the snip-snip of the scissors. Ready to solve it?

Our Hope:

Reasoning about LLMs

Our Issue:

We do not understand even small language models!

Tutorial Structure

LLMs in 5 Formulas

The Five

Perpexity

Attention

GEMM

Chinchilla

RASP

Generation

Memory

Efficiency

Scaling

Reasoning

Big Caveat

Focus on simplified,
base models

(Think like "frictionless"
environment)

Part 1:
Generation

(Perplexity)

Simplified Language

Documents are each made of $T=1000$ word tokens.

A language has $V=10{,}000$ word types.

Language Model

A language model is a probabilistic model of a document

$$p(x_1, x_2, \ldots, x_T; \theta)$$

Chain Rule

Mathematically, we can
do this in any order

$$p(x_1, \ldots, x_T) = \prod_{t=1}^{T} p(x_t \mid x_1, \ldots, x_{t-1})$$

Auto-Regressive Language Model

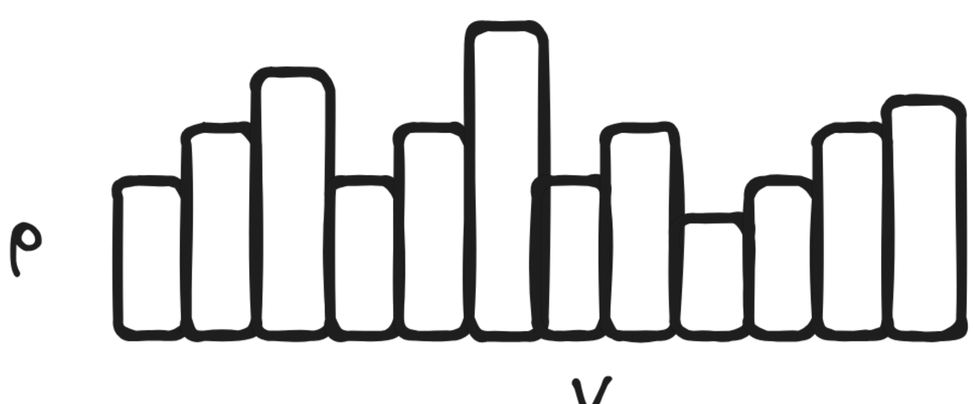
Predict the next token,
conditioned on the previous

$$p(x_t \mid x_1, \ldots, x_{t-1}; \theta)$$

Word Classification

Give a distribution over
V=10,000 choices

$$p(x_t \mid x_1, \ldots, x_{t-1}; \theta)$$



Sampling

Next word generation

$$x_1 \sim p(x_1)$$

$$x_2 \sim p(x_2 \mid x_1)$$

$$\vdots$$

$$x_T \sim p(x_T \mid x_1, \ldots, x_{T-1})$$

Isn't this an old idea?

Sure, Markov and Shannon, but...

Assumption #1: Markovian

Each word only depends on the last word

$$p(x_t \mid x_1, \ldots, x_{t-1}) \approx p(x_t \mid x_{t-1}; \theta)$$

Assumption #2: Categorical

A memorized distribution
for each last word,

i.e. count up next words and divide

Shannon (1948)

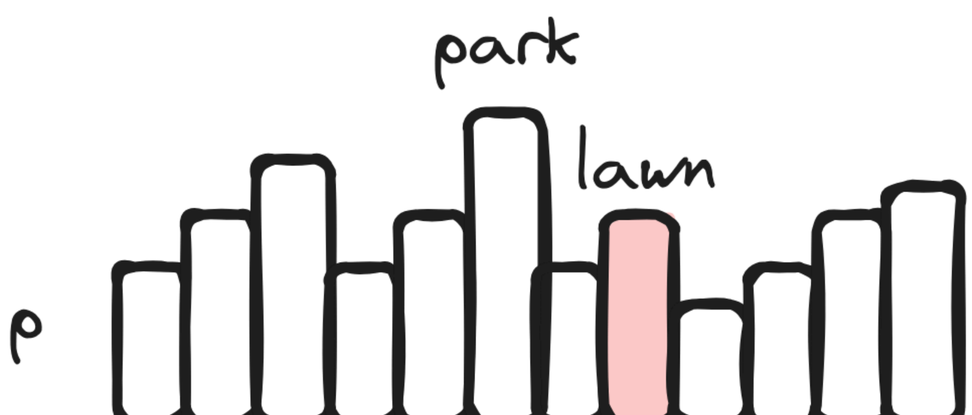
THE HEAD AND IN FRONTAL
ATTACK ON AN ENGLISH WRITER
THAT THE CHARACTER OF THIS POINT
IS THEREFORE ANOTHER METHOD
FOR THE LETTERS THAT THE TIME
OF WHO EVER TOLD THE
PROBLEM FOR AN UNEXPECTED

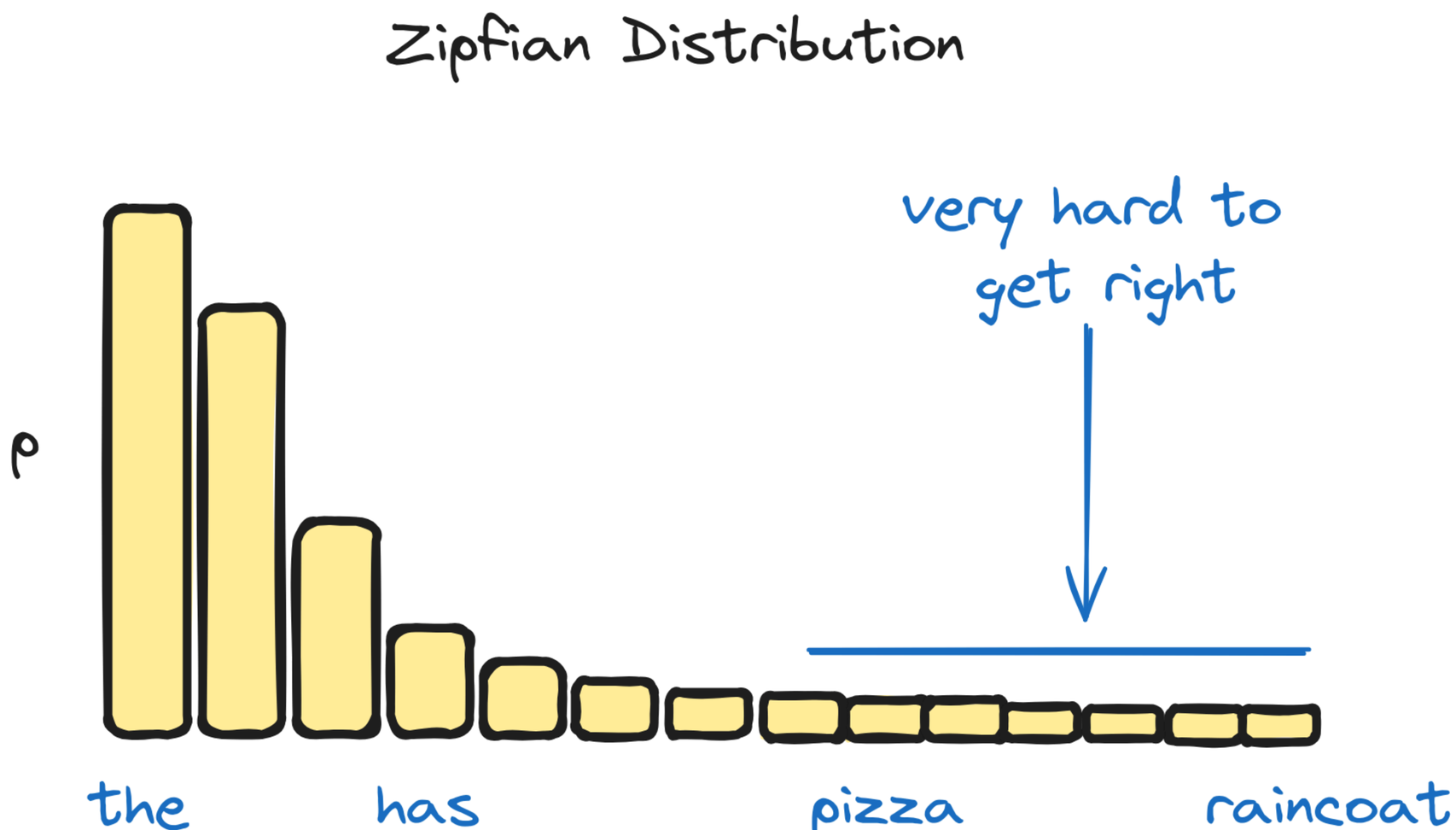
But how do we know it is bad?

1. Give it unseen text
2. Check it's next word prediction
3. Compute a metric

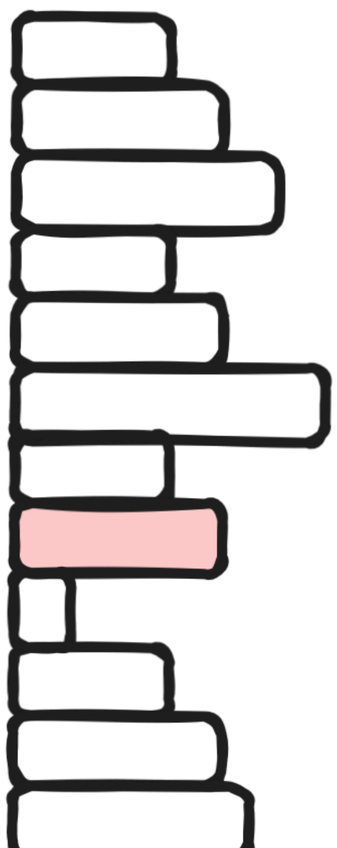
Metric: Accuracy

The dog walked to the ____




Zipfian Distribution
very hard to get right
p
the
has
pizza
raincoat

The dog walked to the ____



11001110

...

10001010

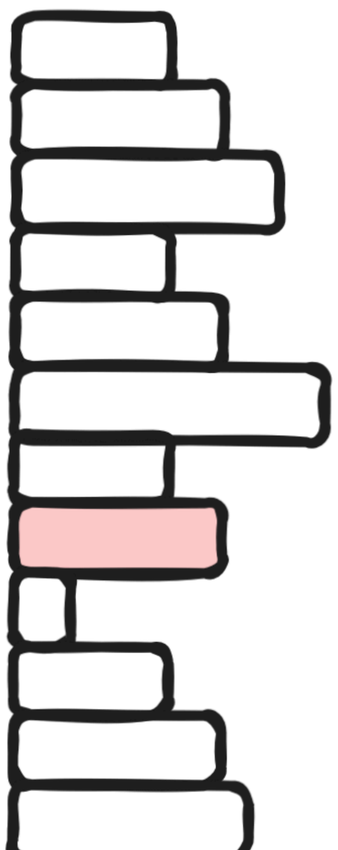
101

100101

01011001010010101010

...

Assign a binary number to each possible word



11001110

...

10001010

101

100101

0101100101001010101010

...

#bits bounded by

$$-\log_2 p(x_t|x_1,\ldots x_{t-1})$$

Perplexity

Think 2^(bits per word)
for the test set

$$2^{-\sum_t \log_2 p(x_t | x_1, \ldots x_{t-1})/T}$$

Examples: Perplexity

Perplexity 1

Deterministic! You've figured it out (no code needed)

Perplexity 10,000

Your model is uniform (same code per choice)

BUG!

Perplexity >10,000

You've assigned very long codes to common words!

Perplexity tells the story of LMs

Wall Street Journal

	Uniform Distribution	10,000
$p(x_t \mid x_{t-1})$	Bigram Distribution	600
$p(x_t \mid x_{t-2}, x_{t-1})$	Trigram Distribution	200
	5-gram + Tricks	140
	Early Markov NNs	~140
	Early non-Markov NNs	100

Why do we care?

Early LM results:
perplexity is highly correlated
with translation quality.

perplexity

PPL (dev)	BLEU (dev)	params $\times 10^6$
4.92	25.8	65
5.29	24.9	
5.00	25.5	
4.91	25.8	
5.01	25.4	
5.16	25.1	58
5.01	25.4	60
6.11	23.7	36
5.19	25.3	50

Actual Translation Quality (higher is better)

Perplexity drives LLMs

Later results: perplexity
is highly correlated with everything

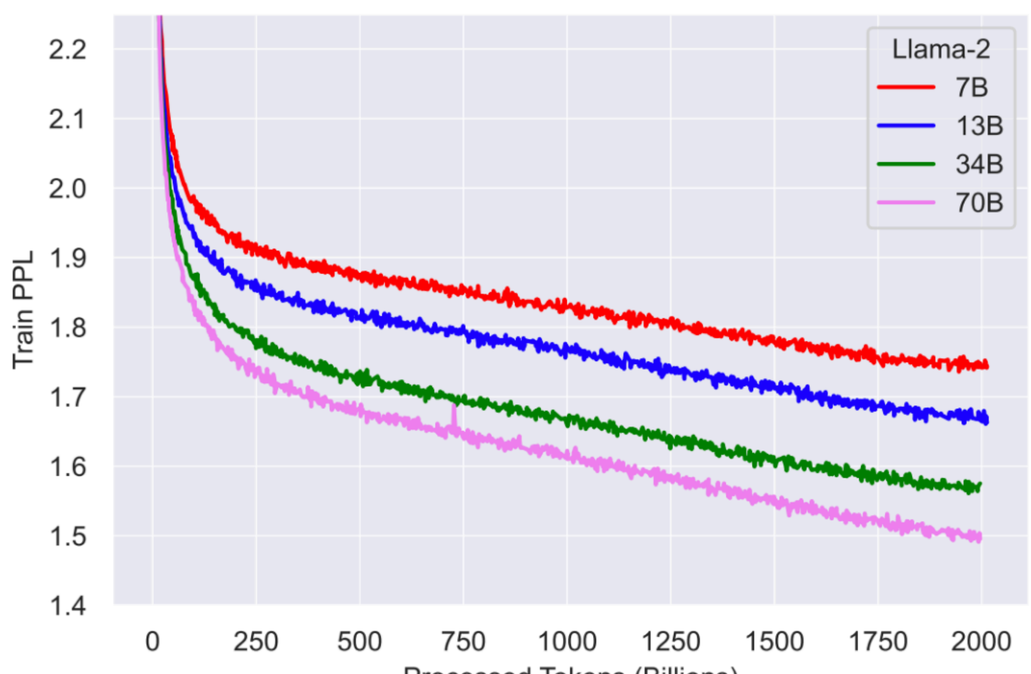
Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

perplexity

random NLP benchmarks

Modern Papers

Can you guess which model works the best?



Perplexity tells the story of LMs

Wall Street Journal

	Uniform Distribution	10,000
$p(x_t \mid x_{t-1})$	Bigram Distribution	600
$p(x_t \mid x_{t-2}, x_{t-1})$	Trigram Distribution	200
	5-gram + Tricks	140
	Early Markov NNs	~140
	Early non-Markov NNs	100

GPT-3

20.5!

(exact word 1/20)

But how do we get from Shannon to GPT-3?

Need to remove our two assumptions.

1. Move to neural networks
2. Consider all previous words

Part 2:
Memory

(Attention)

Language Model Parameterization

Even with Markov assumption,
we can use any model

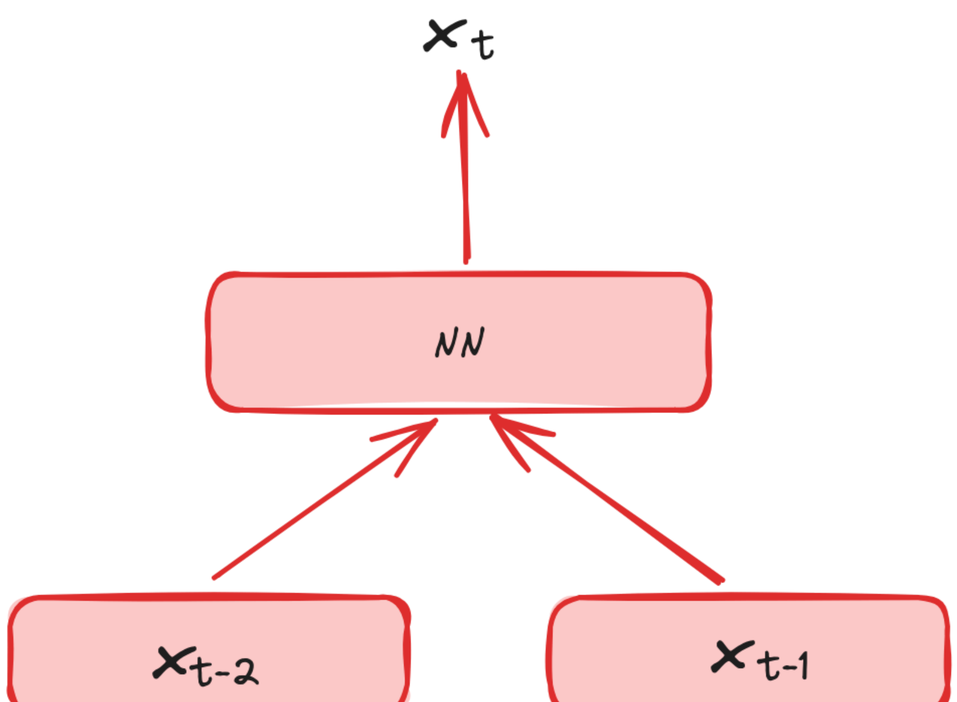
$$p(x_t | x_{t-2}, x_{t-1}; \theta)$$

Neural Net Language Model

Have a neural net pick the distribution

$$\mathrm{softmax}(\mathrm{NN}(x_{t-2}, x_{t-1}))$$

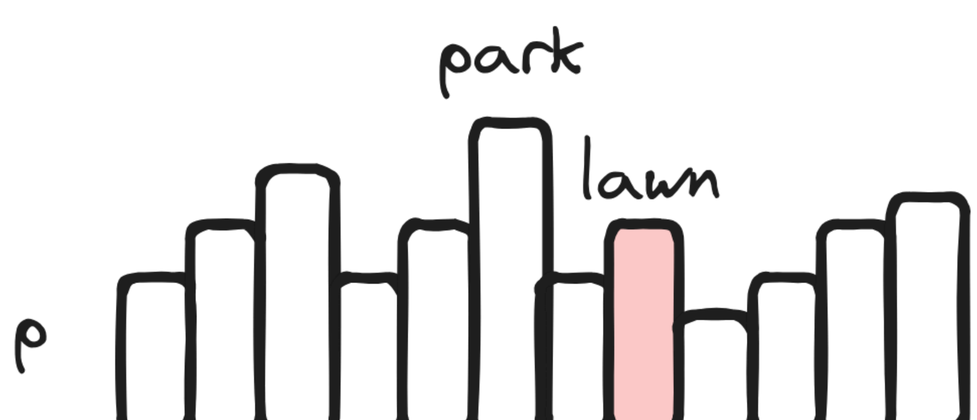
Neural Net Language Model



Softmax

Ensures it is positive and sums to one

$$\mathrm{softmax}(z)_v = \frac{\exp(z_v)}{\sum_{v'} \exp(z_{v'})}$$

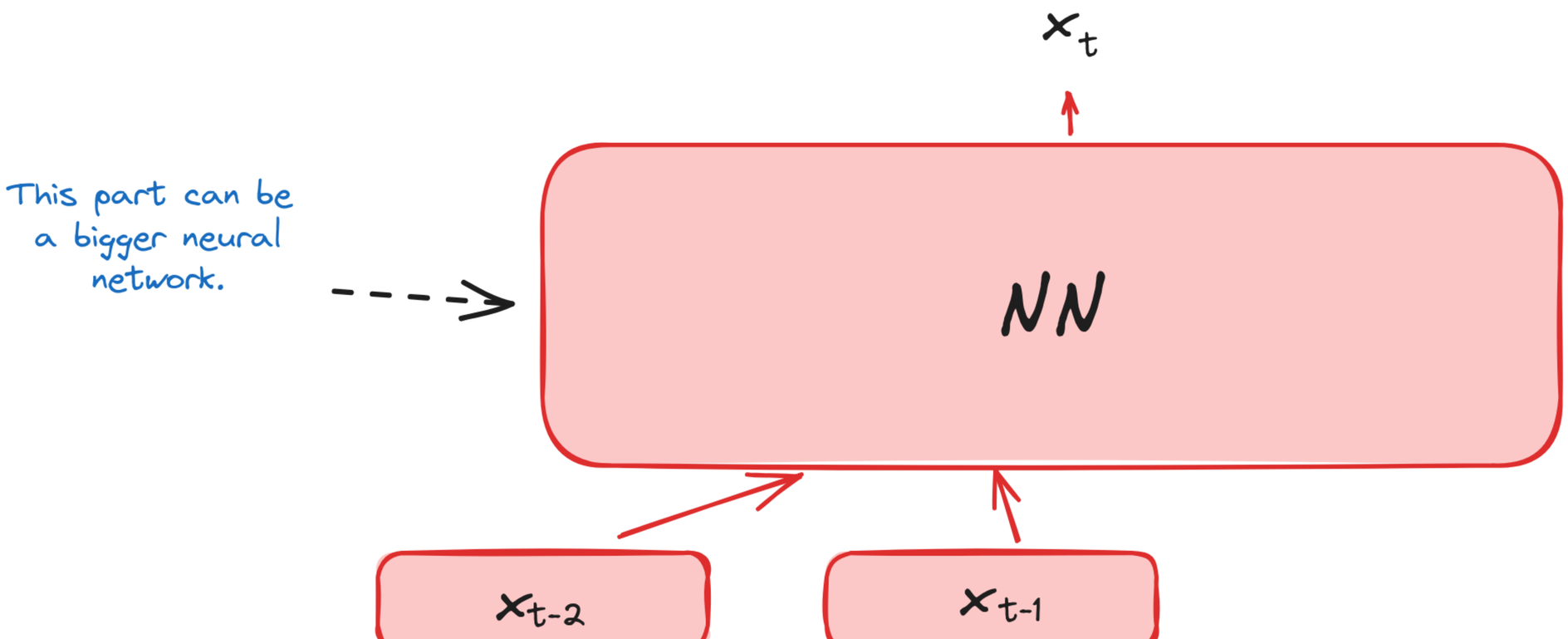


Early Breakthrough: Word2Vec

Demonstrated a Markov NN
language model at scale

Foundational to later methods.

Deep Language Models



Why wasn't this enough?

$$\mathrm{softmax}(\mathrm{NN}(x_{t-2}, x_{t-1}))$$

Most "bits" can be found in nearby tokens.

However, there are useful bits far away.

Running Example

The dog walked to the park

dog walked ___

Pretty easy

walked to ___

to the ___

A location but where? who?

Memory

We spoke with David Parkes in CS

...

...

We looked into that," said Professor ____

Easy bits -> it's a name

Hard bits -> who?

Fully Auto-Regressive Models

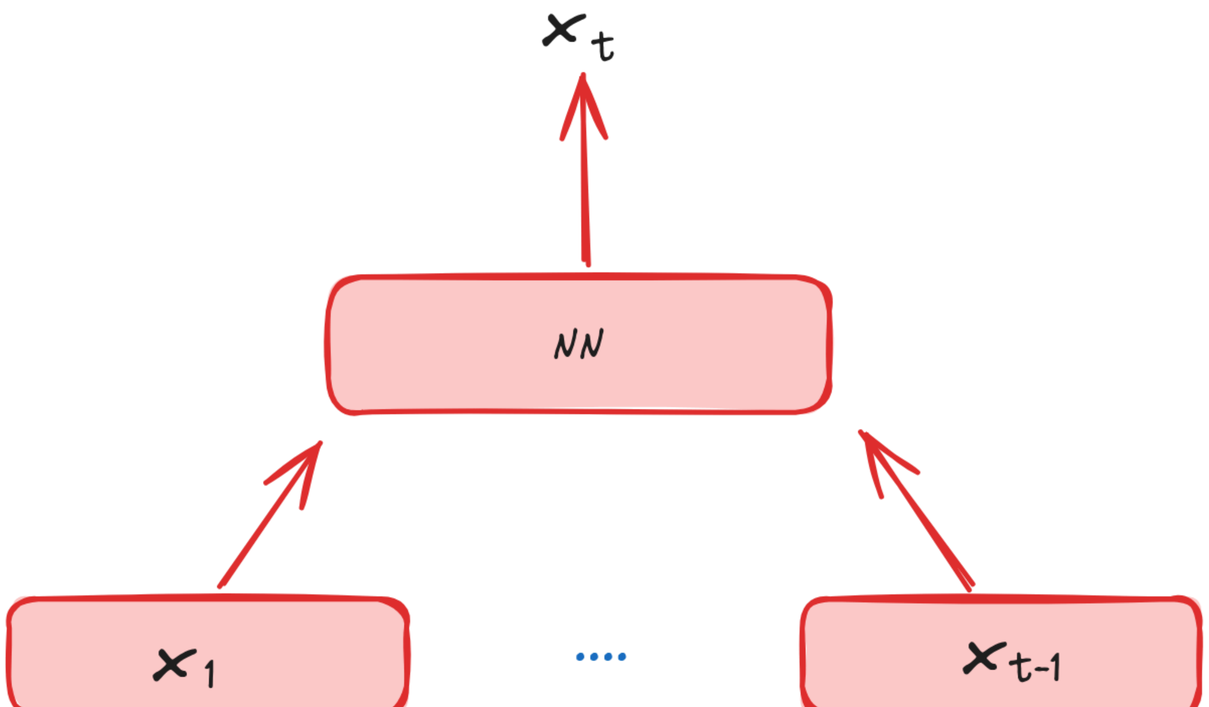
Ideally use all the previous tokens.

(Many different models tried this)

$$\cancel{p(x_t \mid x_1, \ldots, x_{t-1}) \approx p(x_t \mid x_{t-1}; \theta)}$$

$$p(x_t \mid x_1, \ldots, x_{t-1}; \theta)$$

Can't simple neural networks do this?



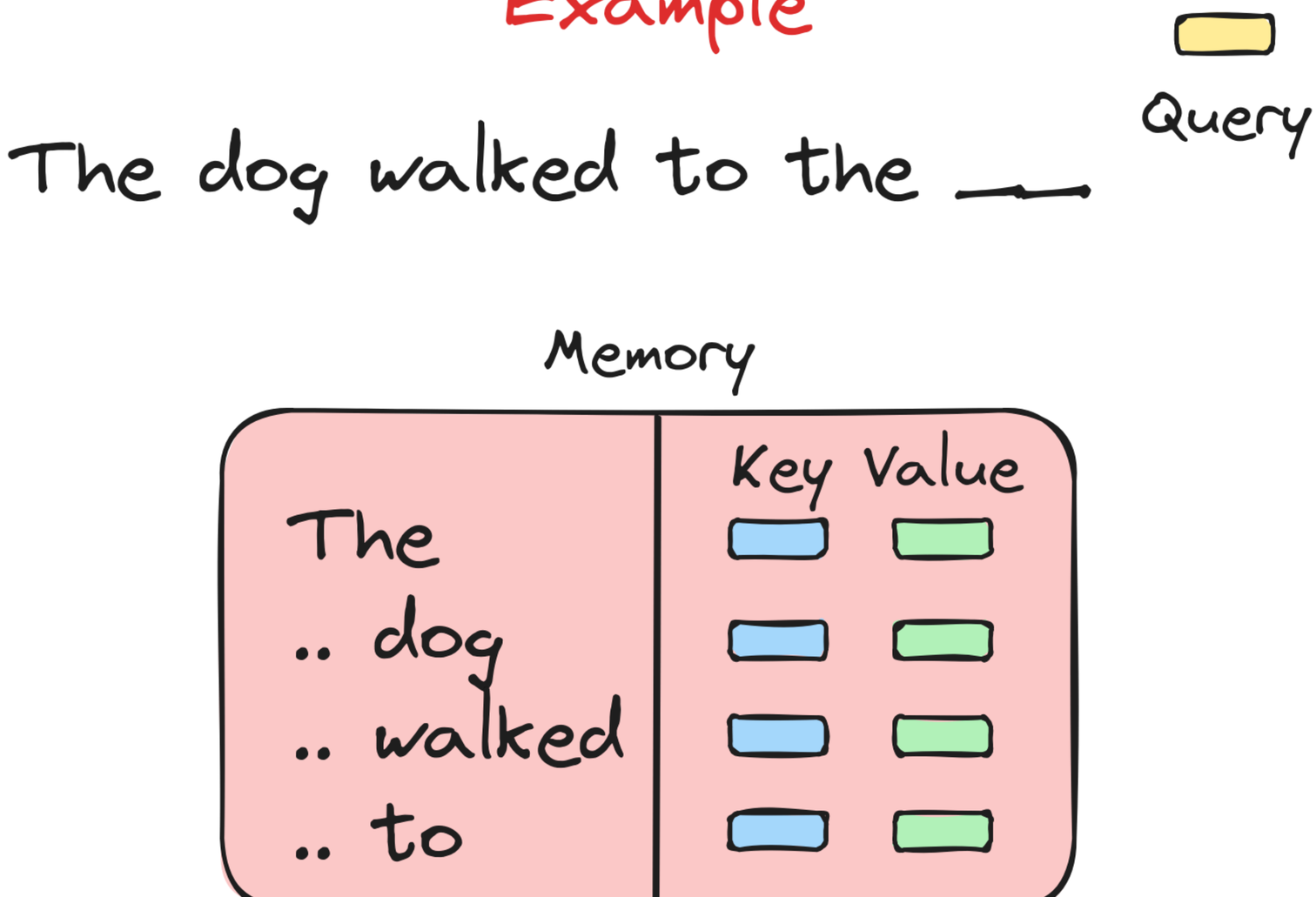
Issue: Learning about Position t=7 doesn't generalize between documents

Solution: Attention

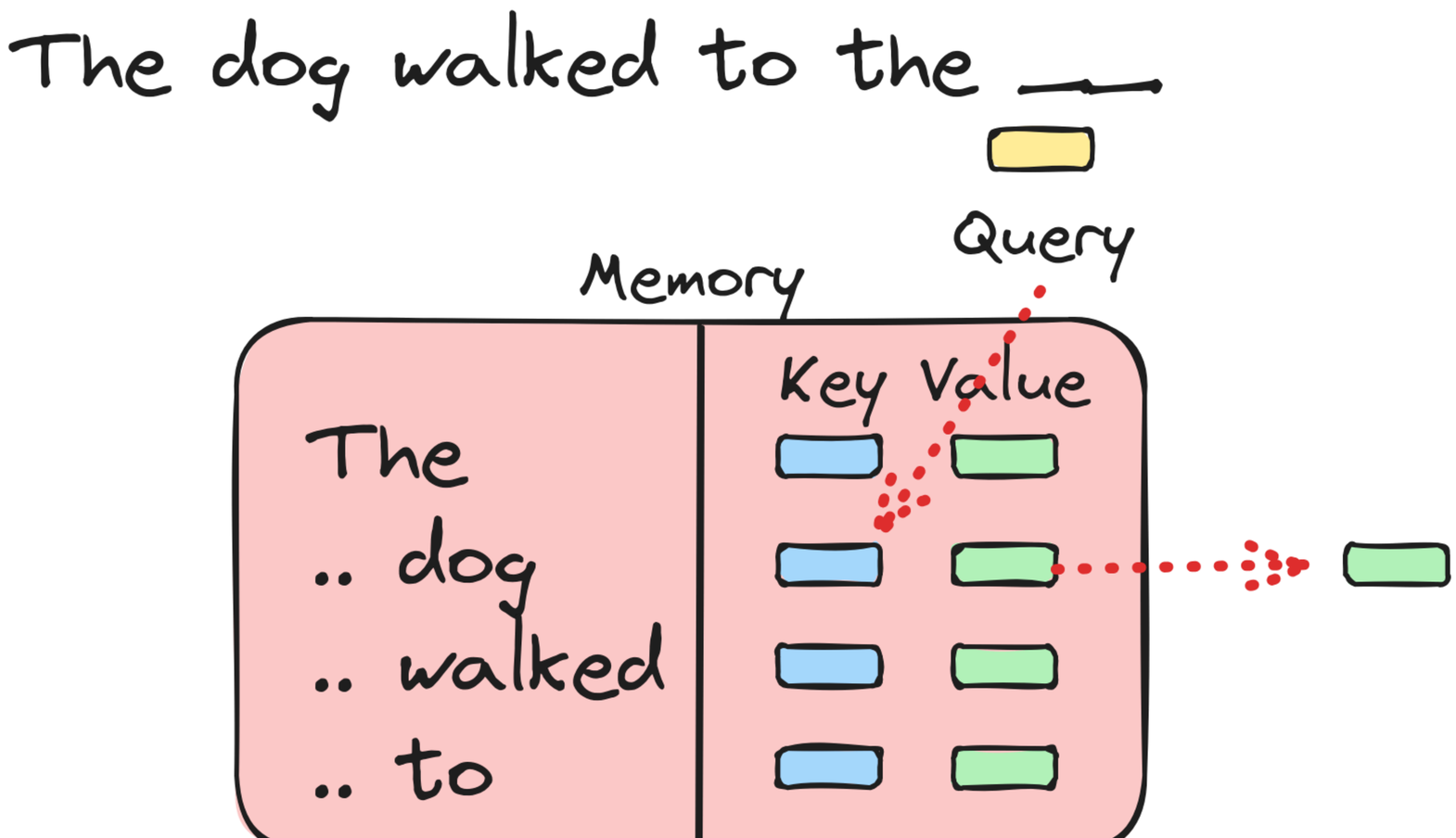
Think "random access memory"
or "lookup table"

Save previous information
in case you need it later.

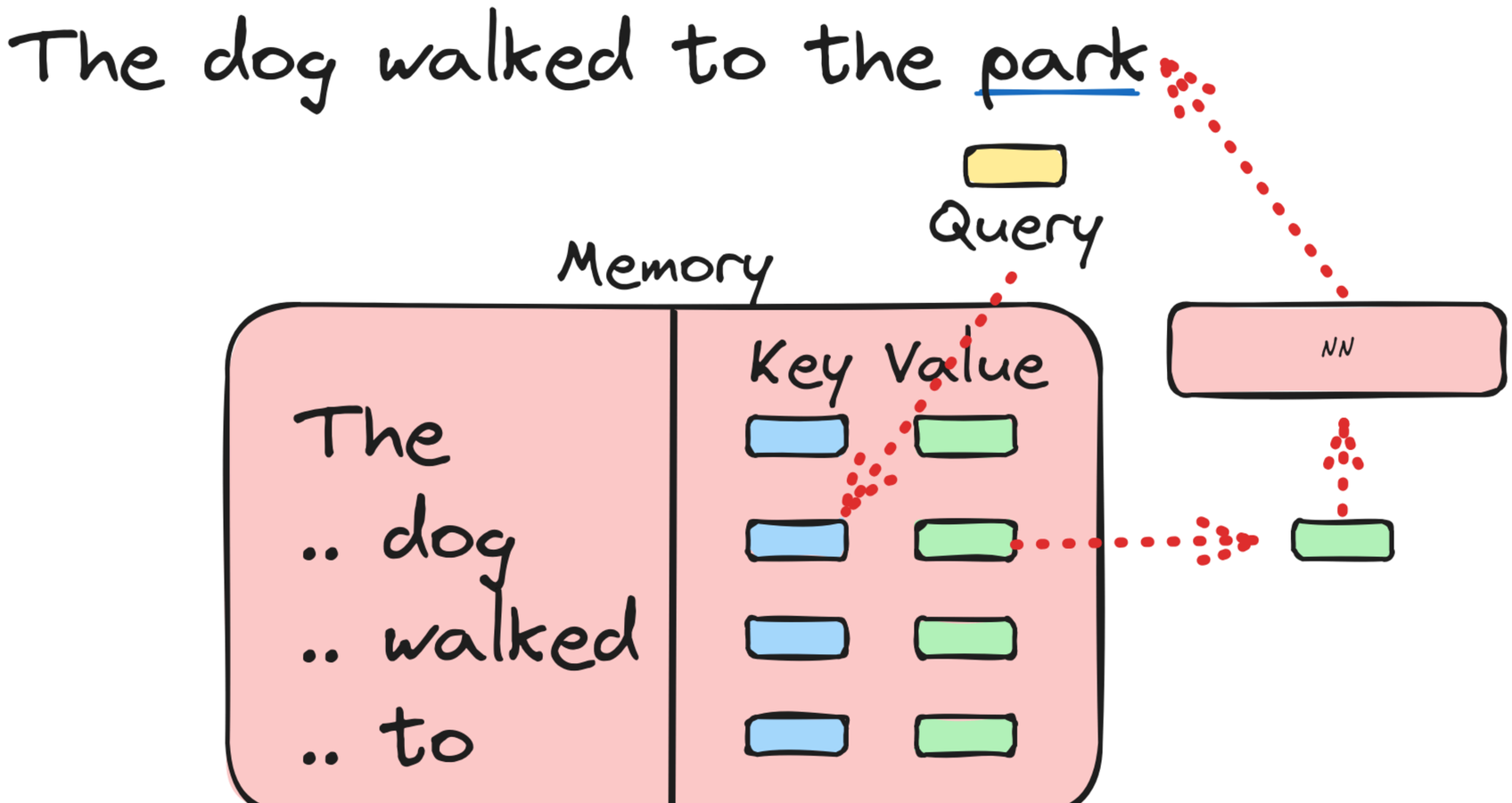
Example


Query
The dog walked to the ___
Memory
Key
Value
The
.. dog
.. walked
.. to

Example


The dog walked to the _
Query
Memory
Key
Value
The
.. dog
.. walked
.. to

Example


The dog walked to the park
Query
Memory
Key Value
The
.. dog
.. walked
.. to
NN

Process

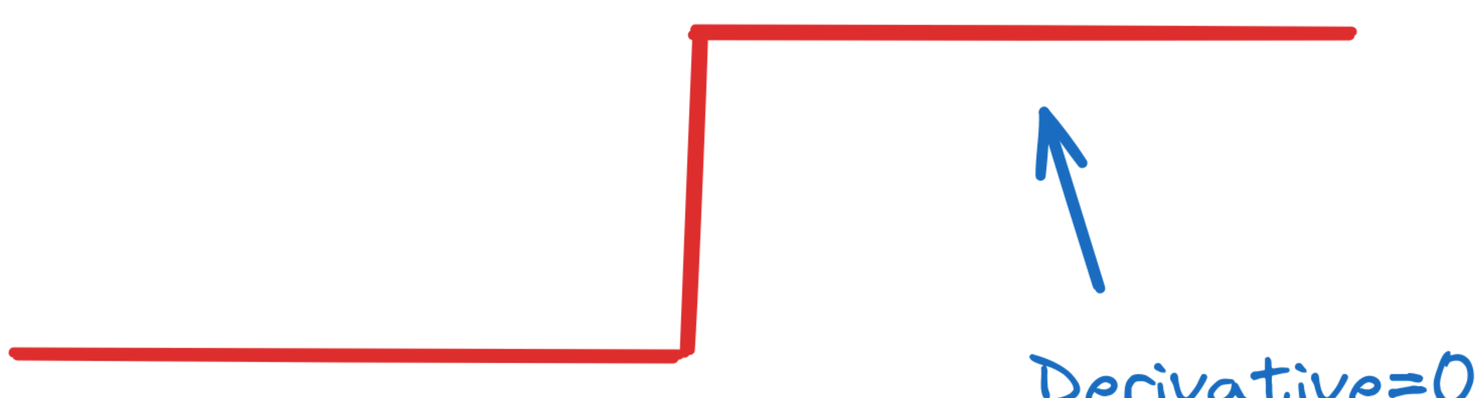
Step 1. Query matches Key

Step 2. Argmax match is selected

Step 3. Return Value of that Key

Issue

Neural Networks
cannot learn with argmax.

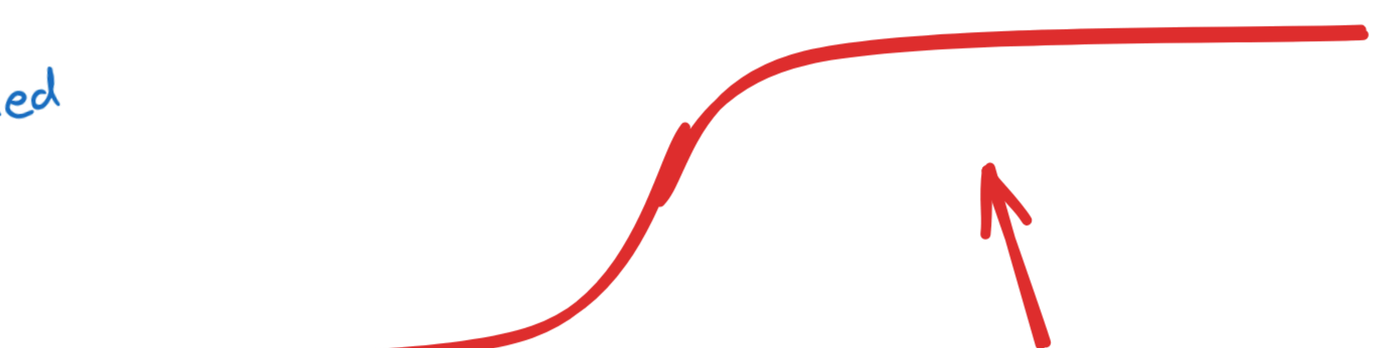


Argmax => Softmax

Replace argmax with softmax

$$\mathrm{softmax}(z)_v = \frac{\exp(z_v)}{\sum_{v'} \exp(z_{v'})}$$

softmax is called sigmoid in 2D



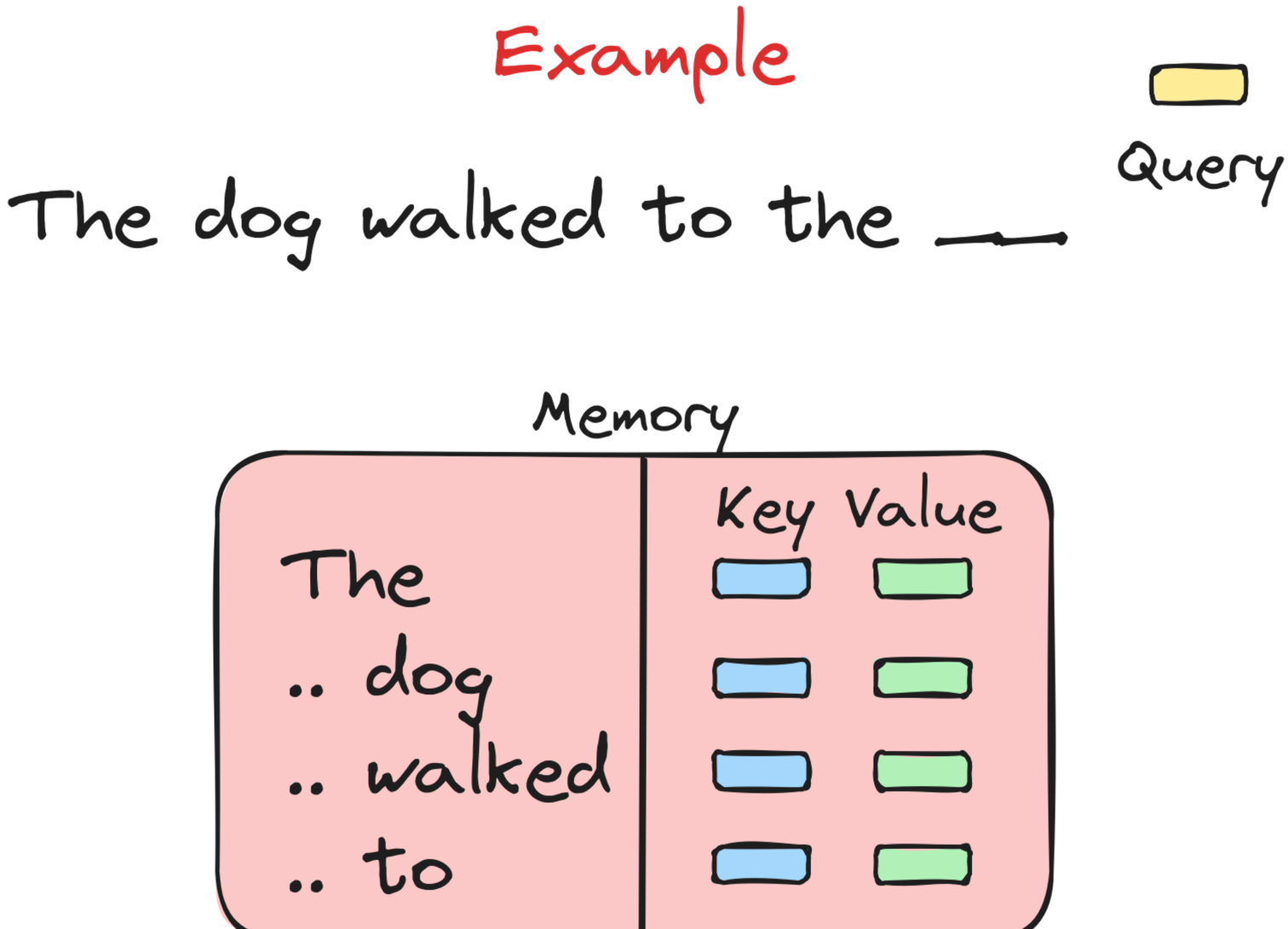
New Process

Step 1. Query scores Key

Step 2. Softmax normalizes scores

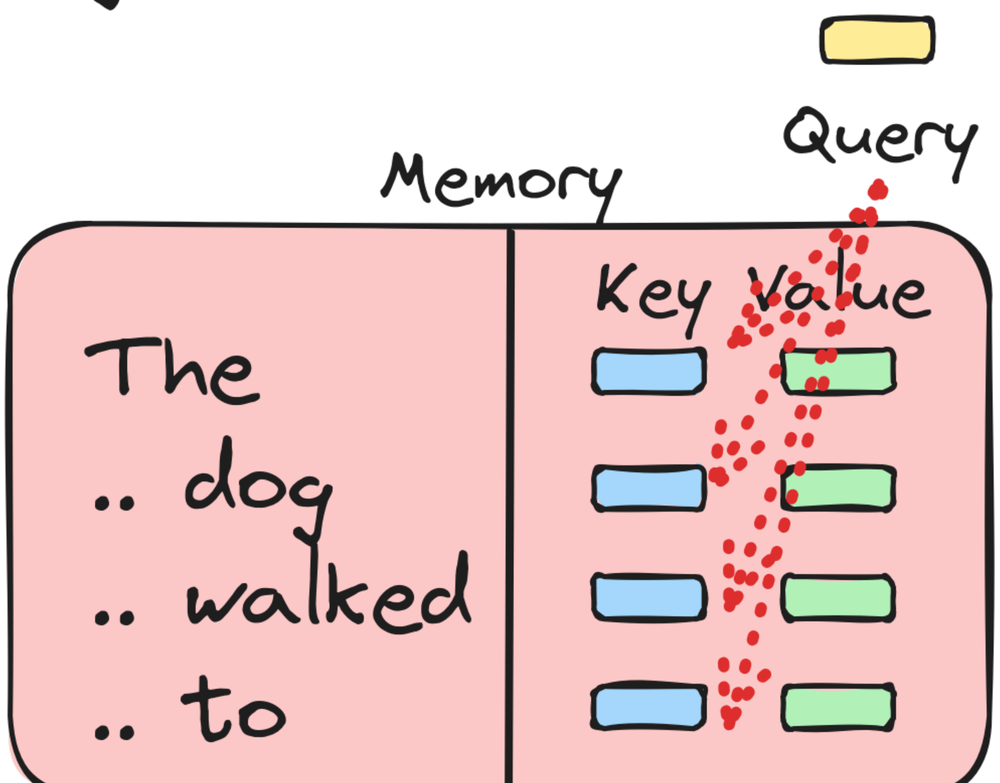
Step 3. Average Value by score

Example


Query
The dog walked to the ___
Memory
Key
Value
The
.. dog
.. walked
.. to

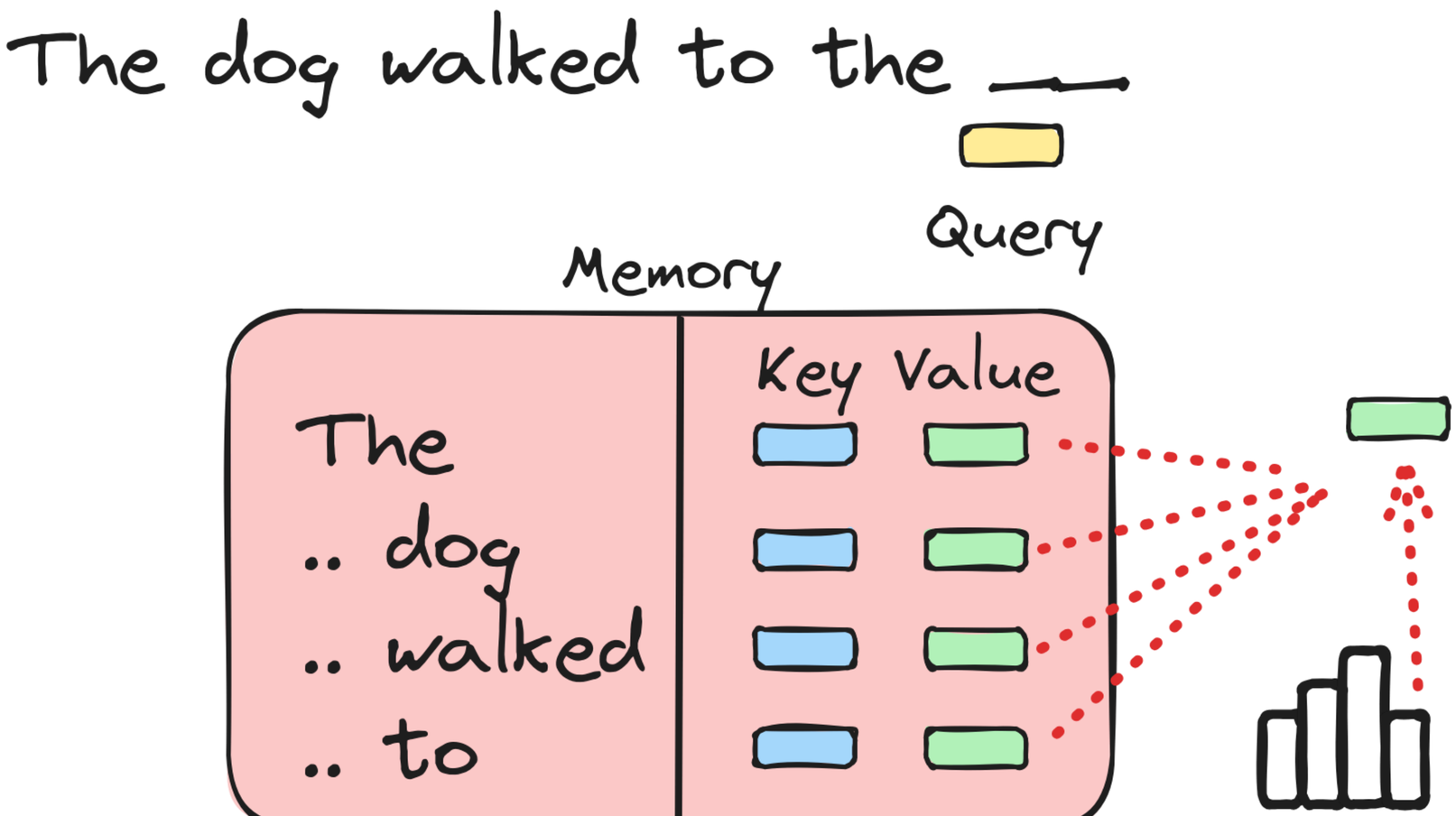
Example

The dog walked to the ___

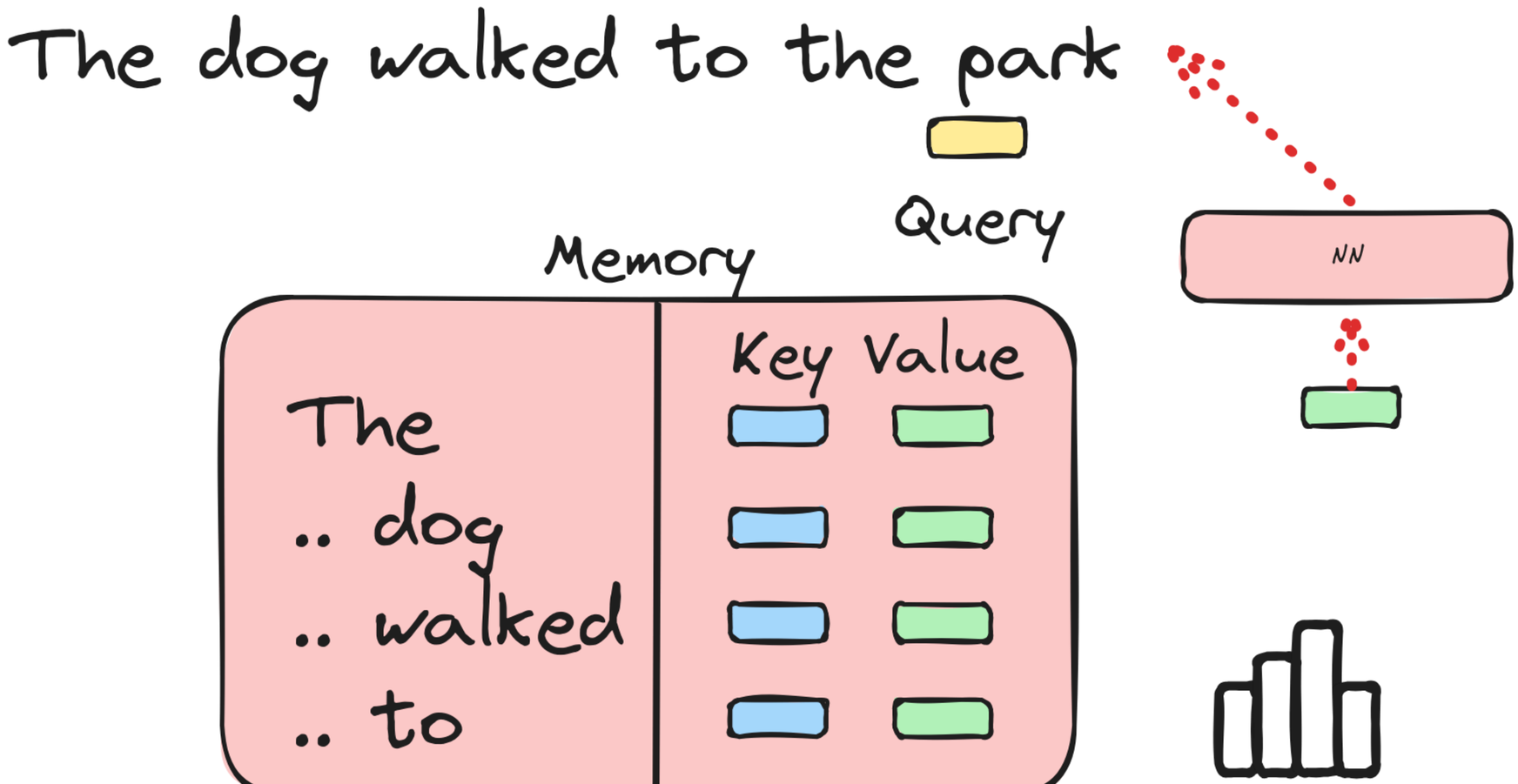


Softmax of match scores

Example


The dog walked to the
Query
Memory
Key
Value
The
.. dog
.. walked
.. to

Example


The dog walked to the park
Query
NN
Memory
Key
Value
The
.. dog
.. walked
.. to

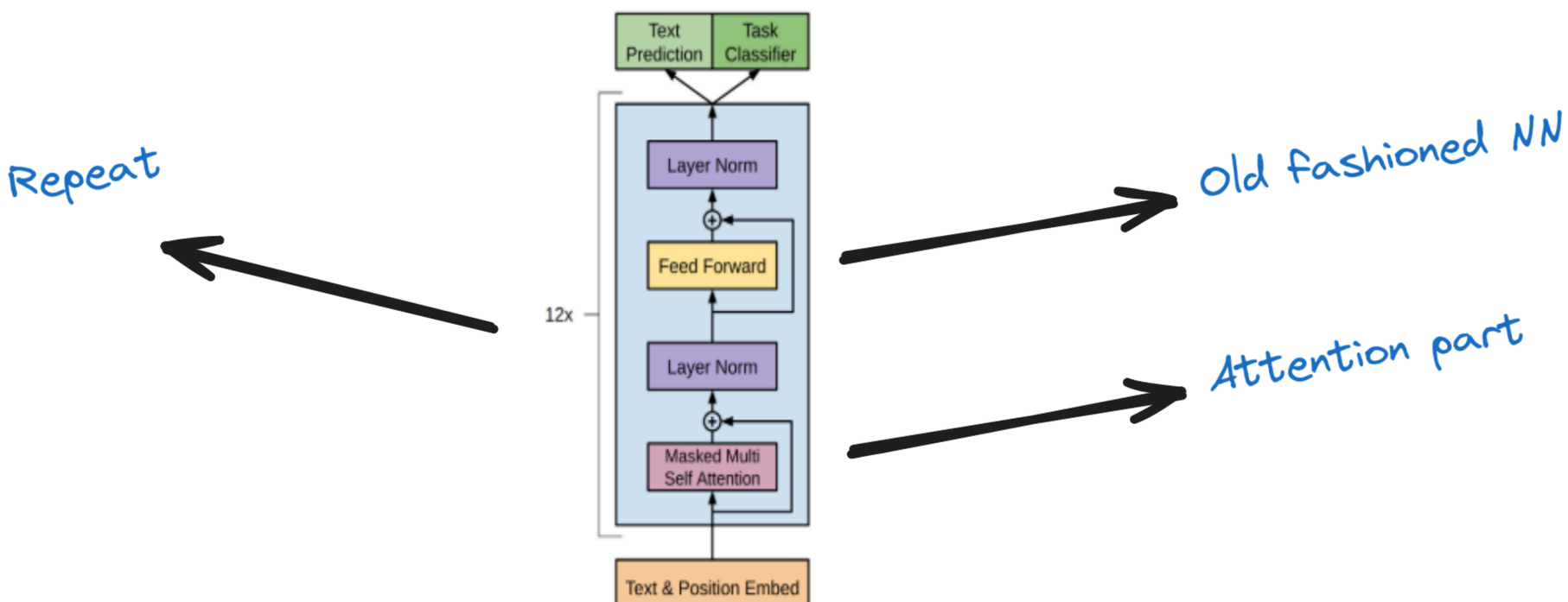
Attention is all you need (2017)

Central paper in language modeling.

Showed that NN + Attention = Transformer

$$p(x_t \mid x_1, \ldots, x_{t-1}; \theta)$$

Transformer



But why Attention?

High-level answer:
Attention is like memory

Real answer:
Attention is very efficient and parallizable

Example

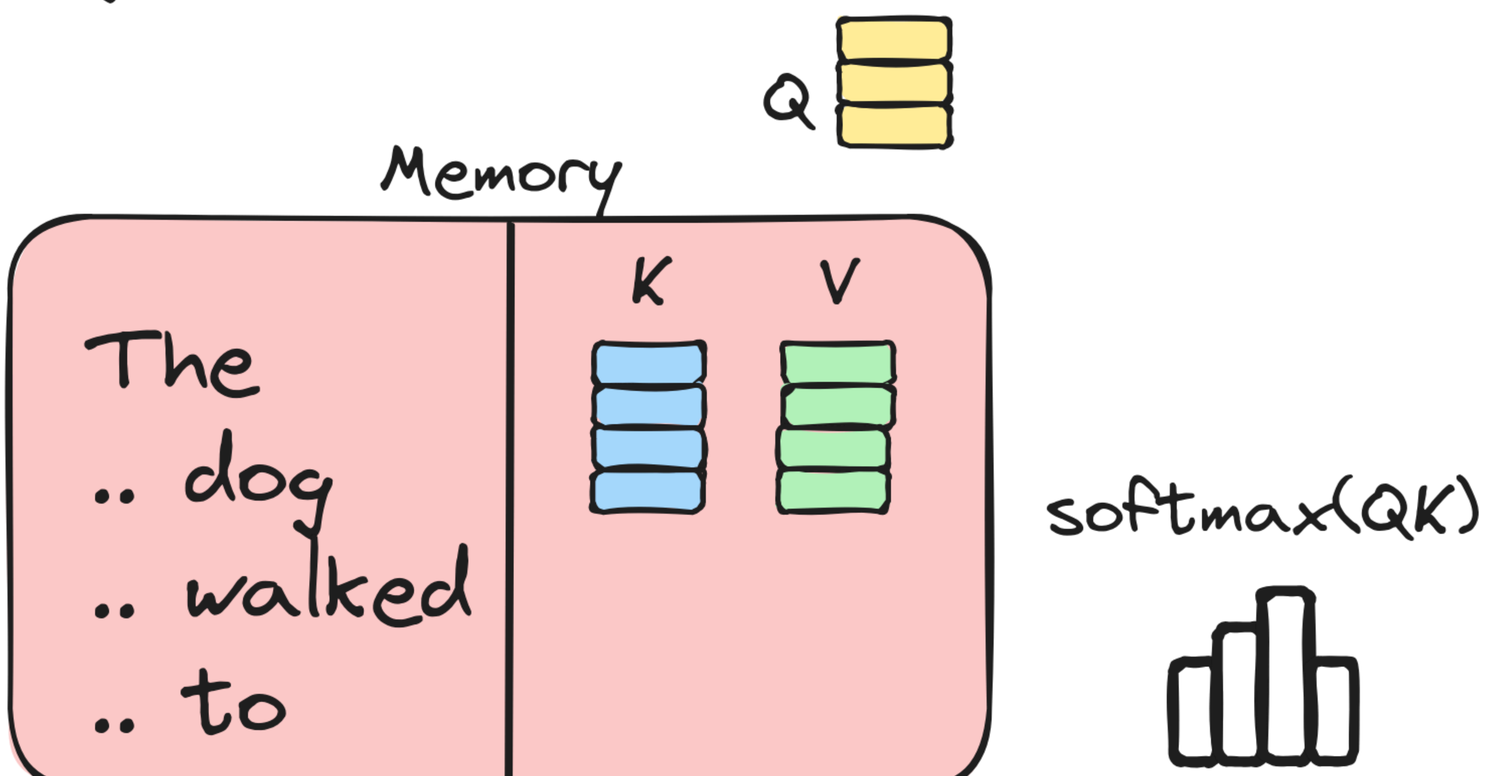
The dog walked to the _____

Memory

The
.. dog
.. walked
.. to

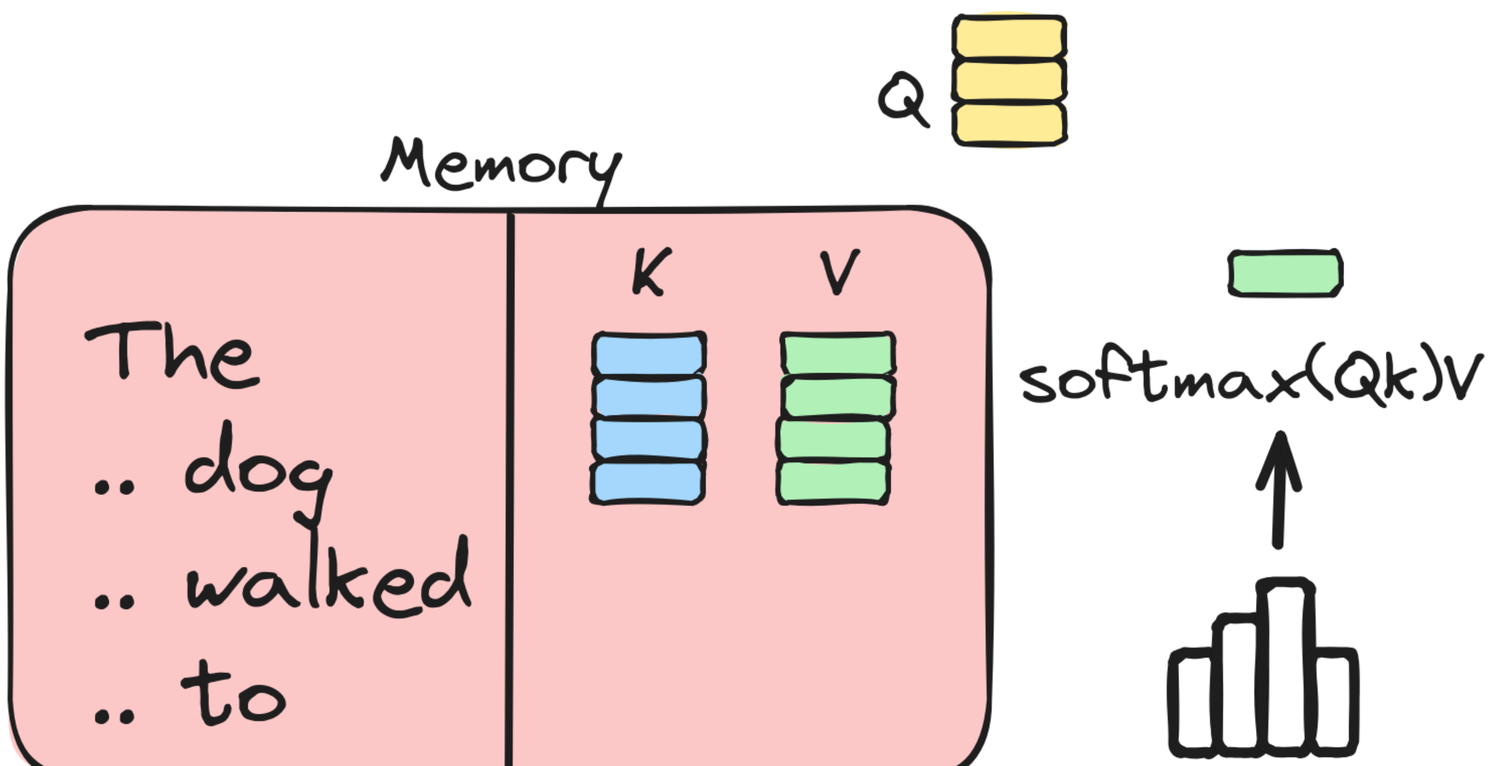
Example

The dog walked to the ___



Example

The dog walked to the ___



Final Process

$$\mathrm{softmax}(\mathbf{Q}\mathbf{K}^\top)\mathbf{V}$$

1. Q scores K
2. Softmax normalizes scores
3. Weight V by score

Efficient because of matrix mult

But why is this actually efficient?

We now have a "Generative Transformer"

Need to understand how this runs fast.

$$\mathrm{softmax}(\boxed{\mathbf{Q}\mathbf{K}^\top})\mathbf{V}$$

Part 3:
Efficiency

(GEMM)

Why did language models suddenlty get so much better?

GPUs

NVidia

1D
5D
1M
6M
YTD
1Y
5Y
MAX
Key events
700
600
500
400
300
200
100
0
2020
2021
2022
2023
2024

General Purpose GPUs

The move from CPUs to GPUs fundamentally altered what models were possible to build

One example:
Softmax for Vocab

$$\text{softmax}(z)_v = \frac{\exp(z_v)}{\sum_{v'} \exp(z_{v'})}$$

This sum is
really big! V = 10,000

~2010

$$\mathrm{softmax}(z)_v = \frac{\exp(z_v)}{\sum_{v'} \exp(z_{v'})}$$

Lots of research into approximating the sum

i.e. importance sampling

GPU

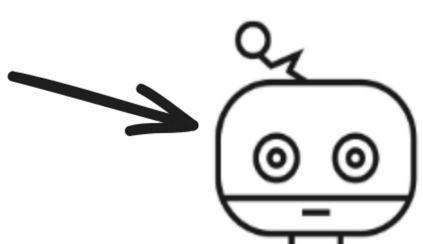
Our focus:
Matrix Multiplication

$$C = AB$$

What is a GPU?

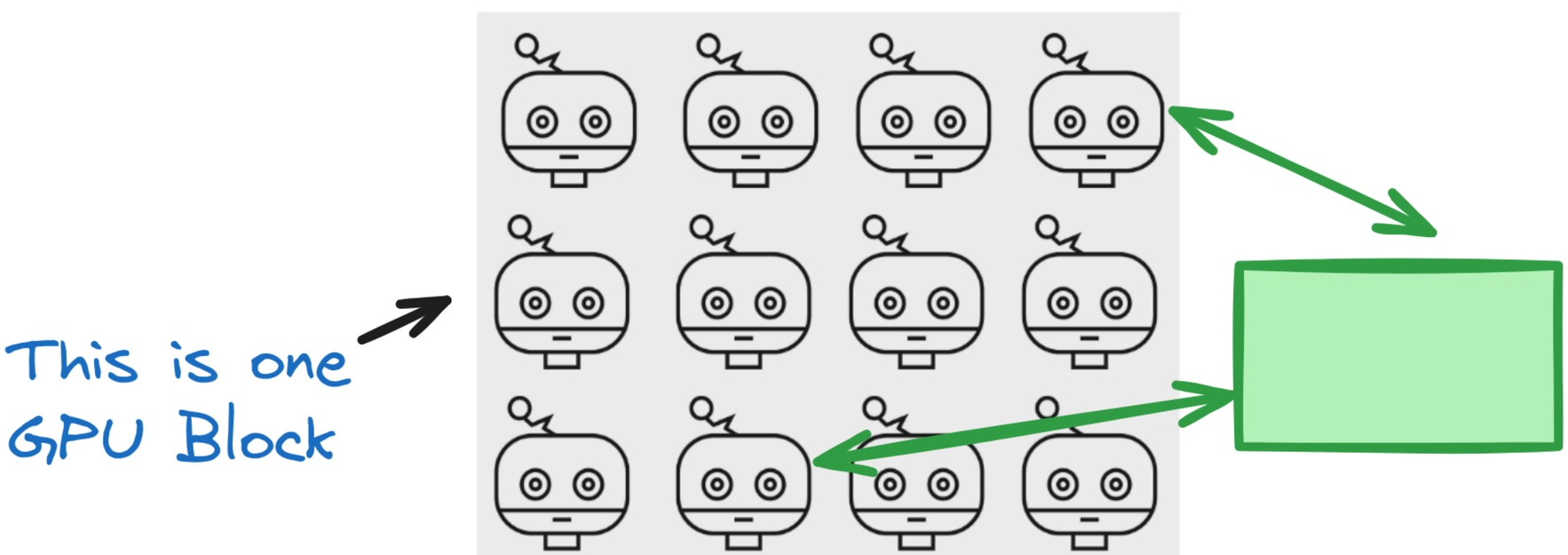
Think "parallel computer".
Many threads running simultaneously.

This is one
GPU thread



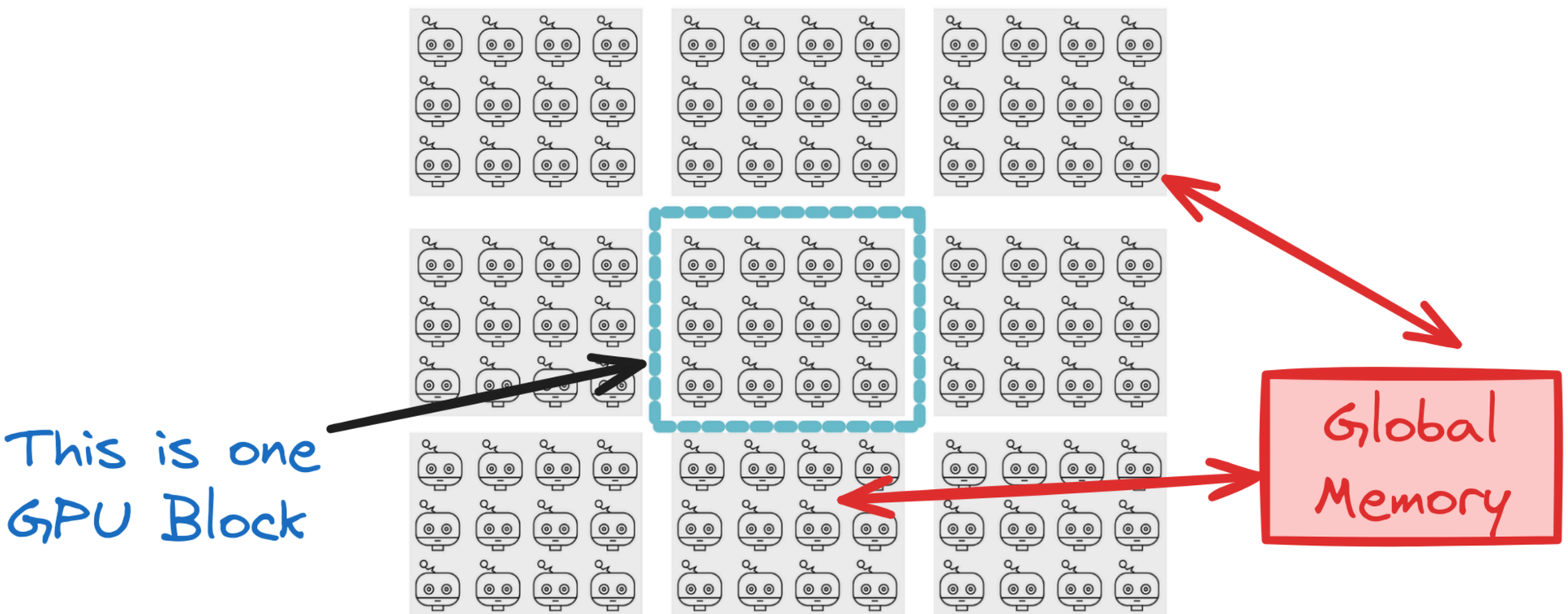
Blocks

Threads are grouped into Blocks



The Grid

Blocks are grouped into a Grid



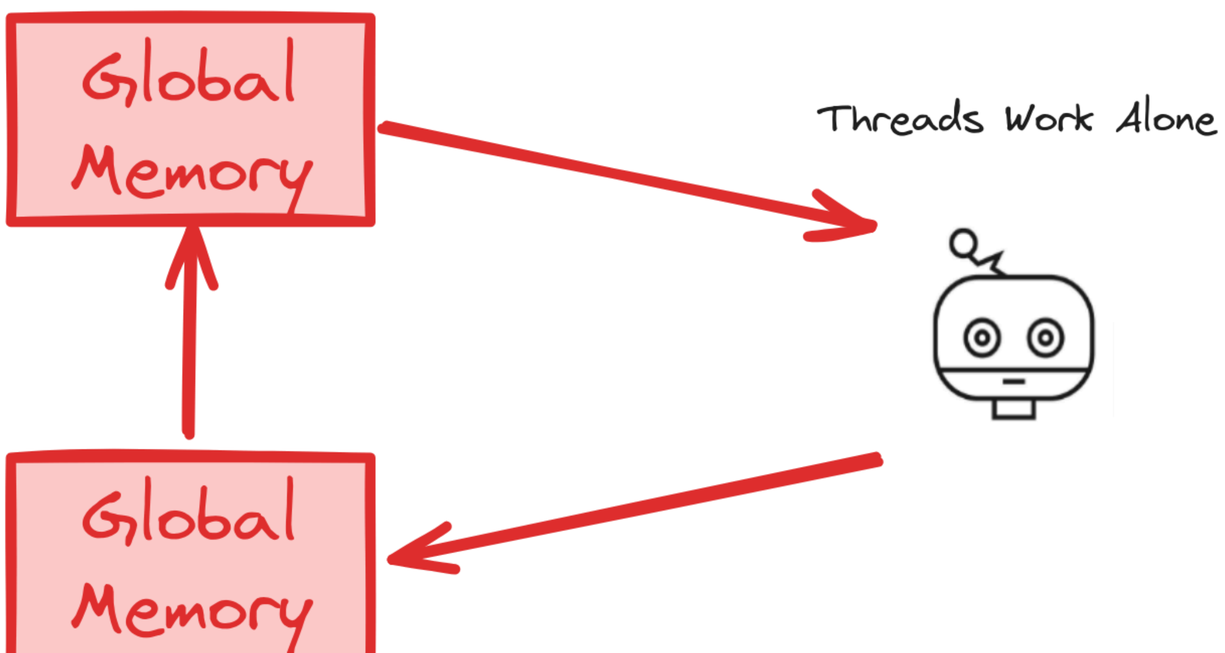
The Rules

Limited number of threads per block

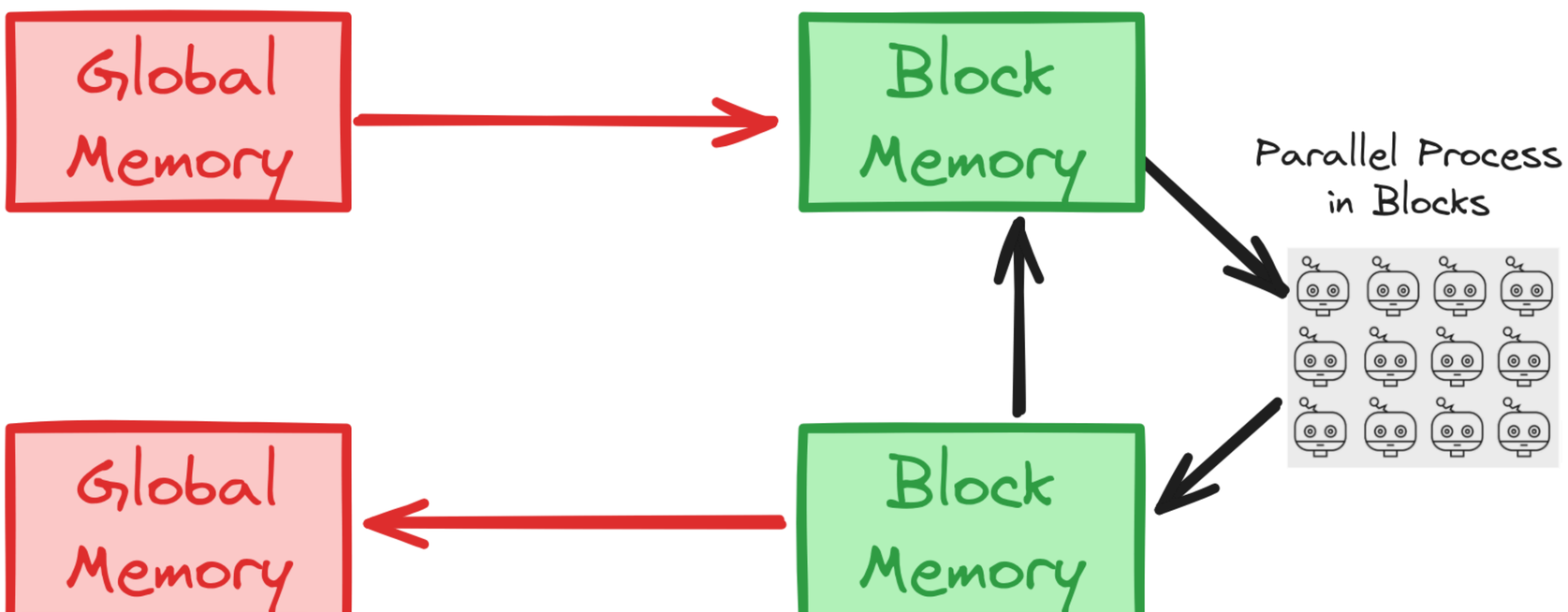
Global Memory is much, much slower than Block memory

This is the important one!

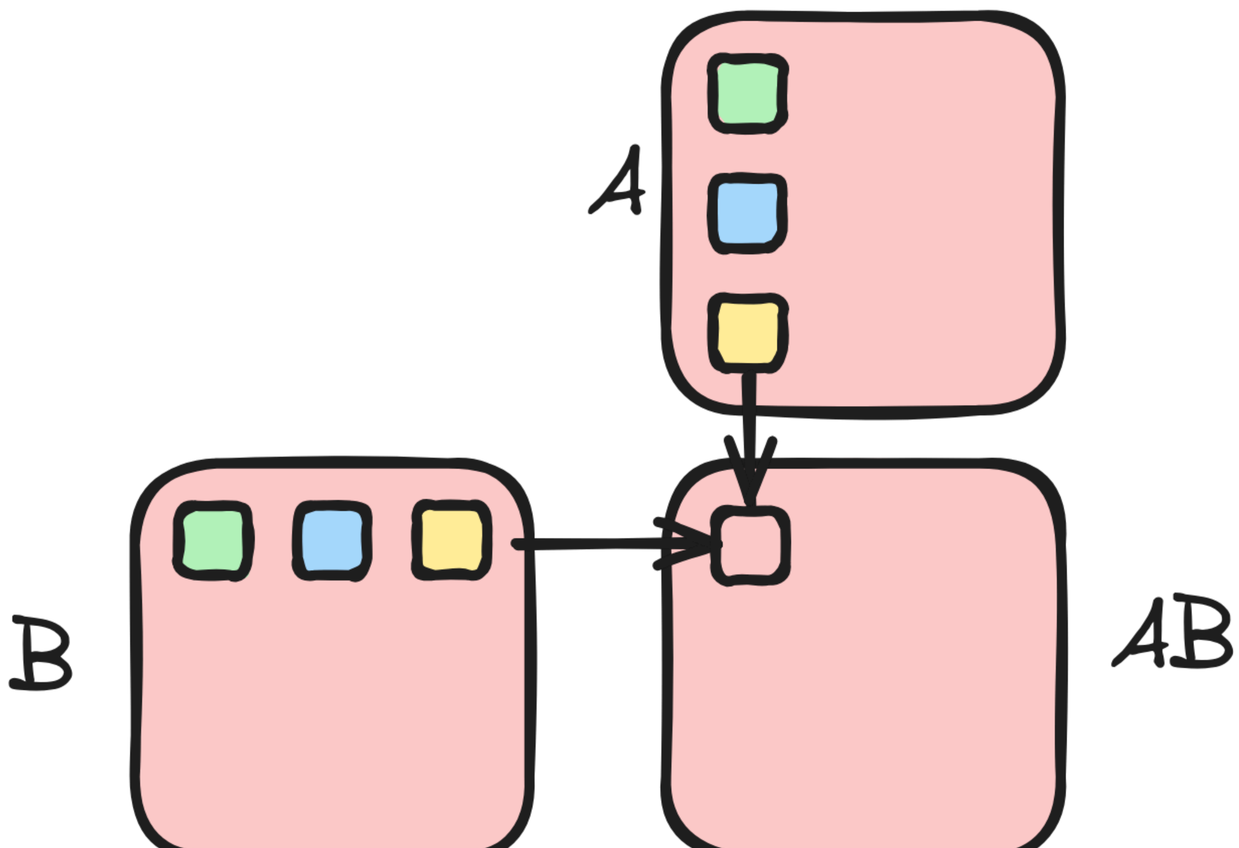
Bad: Threads Work alone



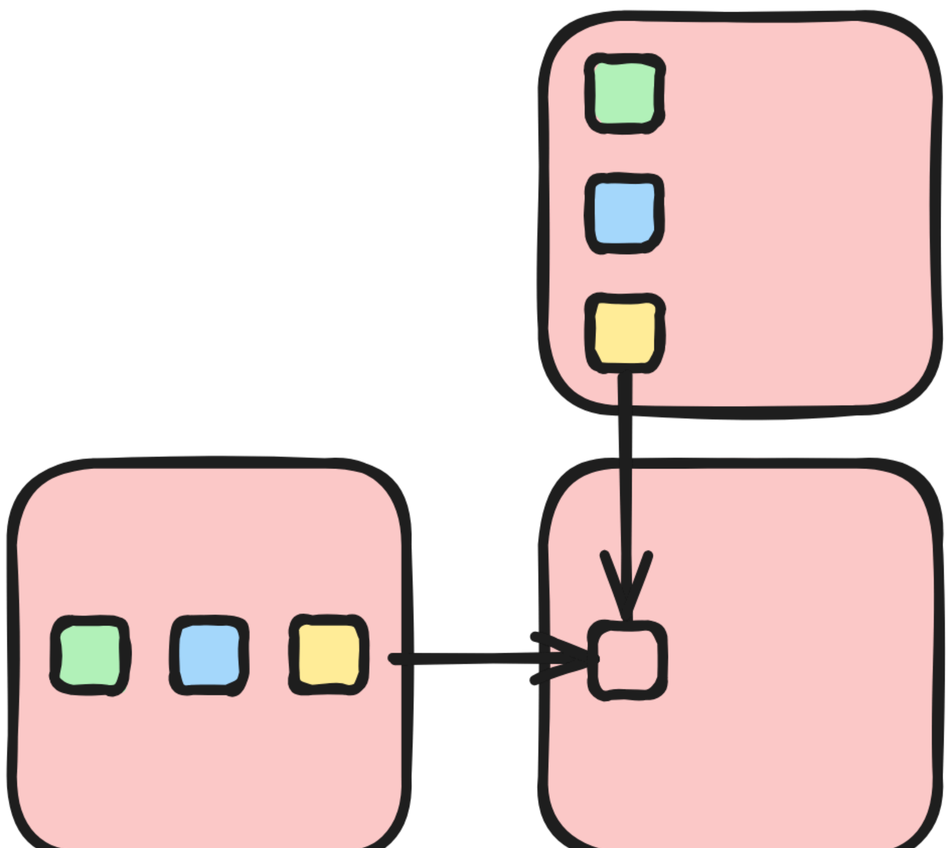
Good: Memory Hierarchy



Example: 3x3 Matrix Multiplication

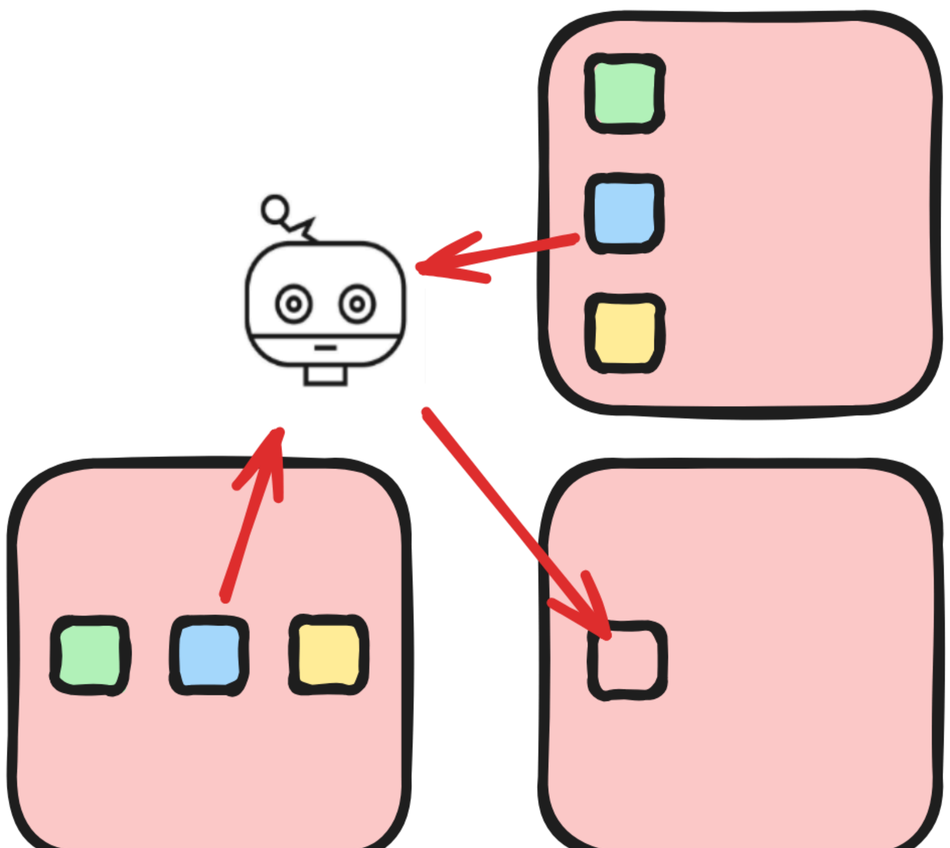


First element



Second element

Bad Method: Each thread works alone



6 reads
from Global

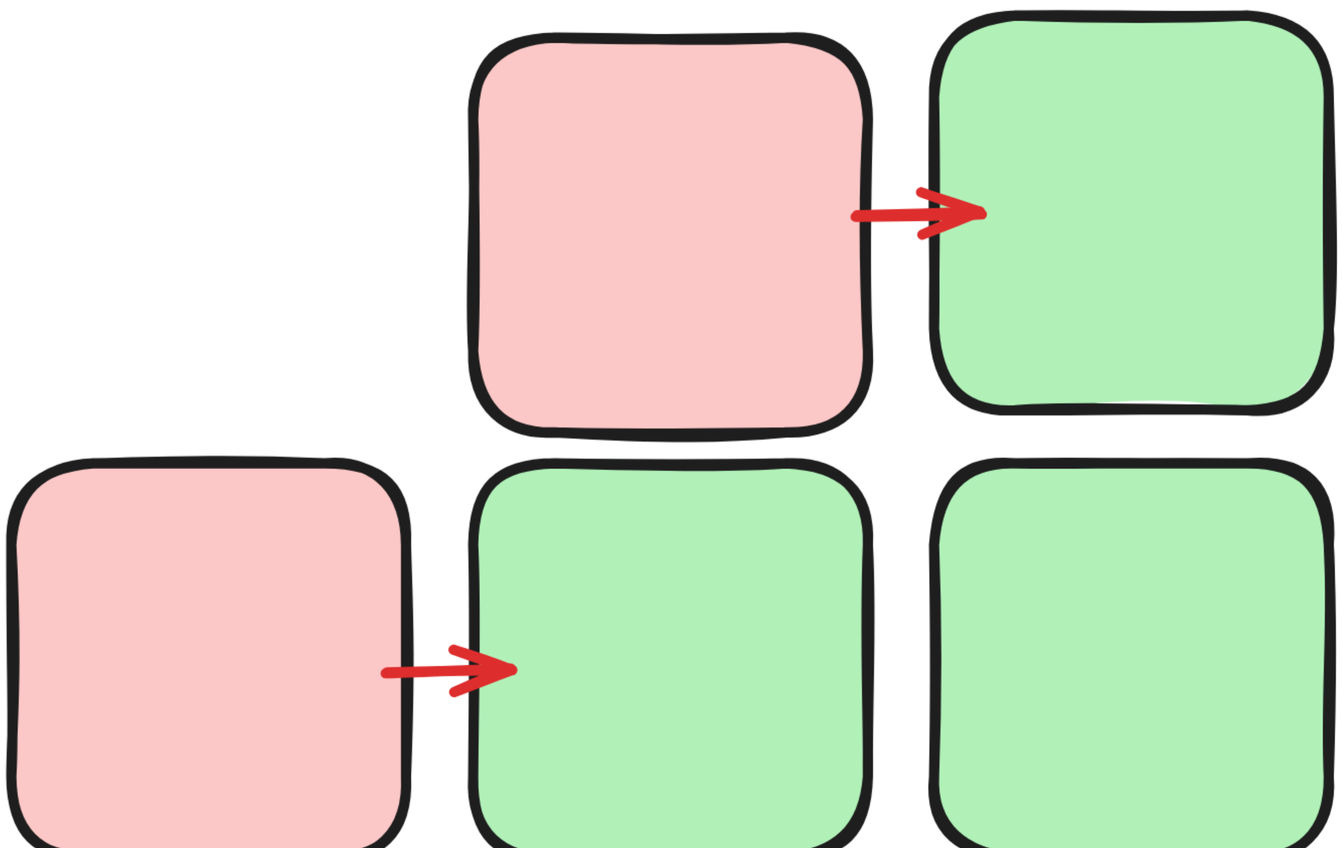
Good Method

1. Read first to Block memory

2. Calculate intermediate results in Block

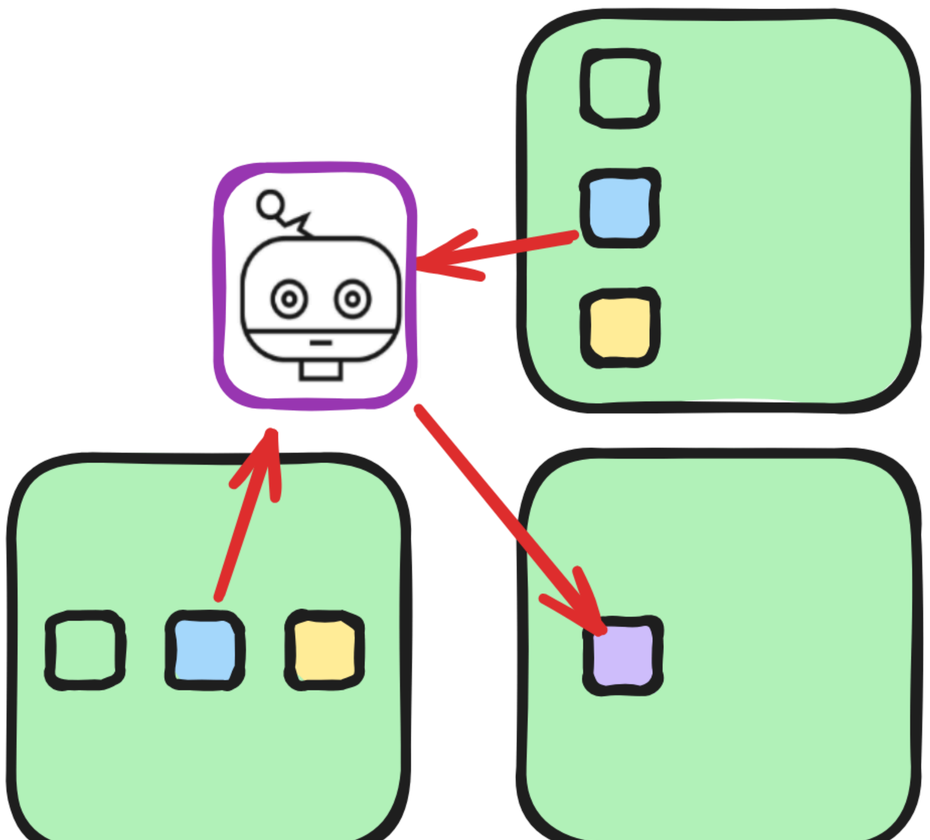
Repeat until done.

Step 1: Read to Block

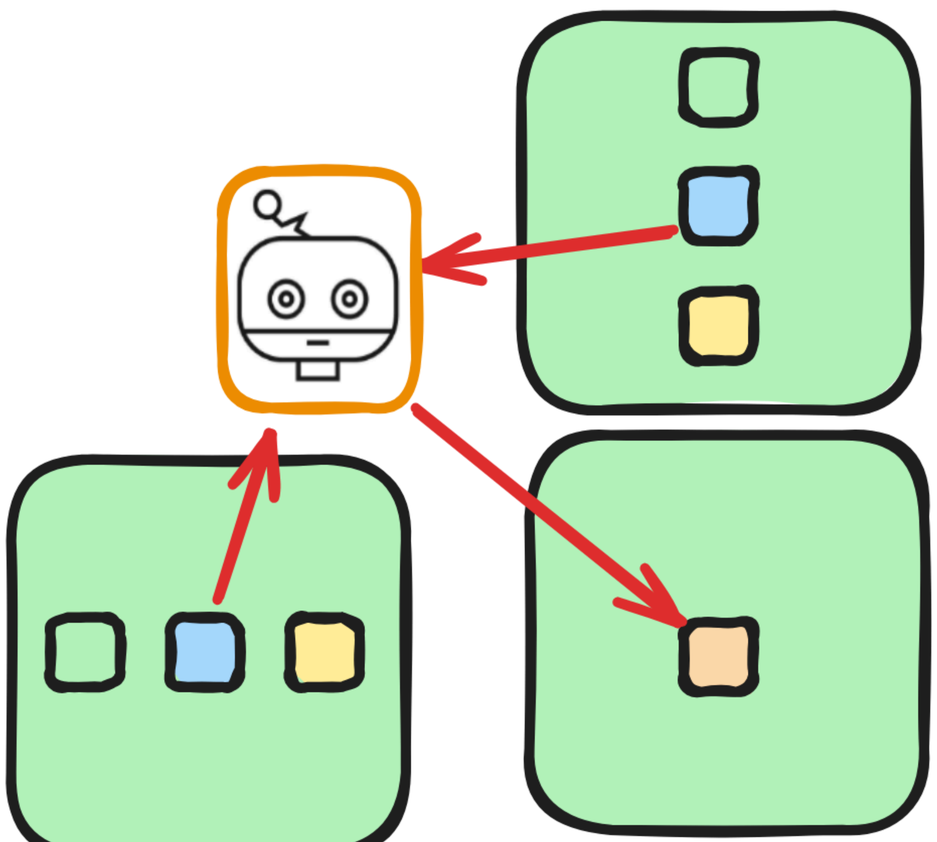


2x9 reads

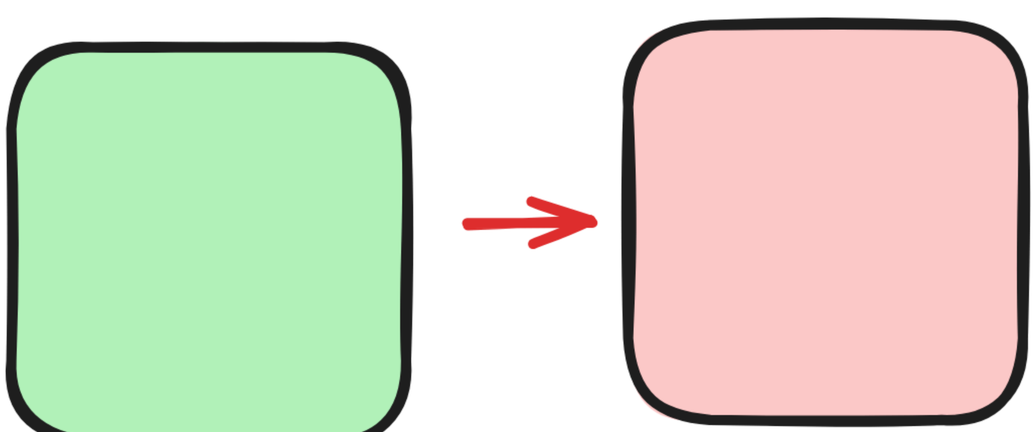
Step 2: Calulate in Block Memory



Step 2: Calulate in Block Memory



Write back out



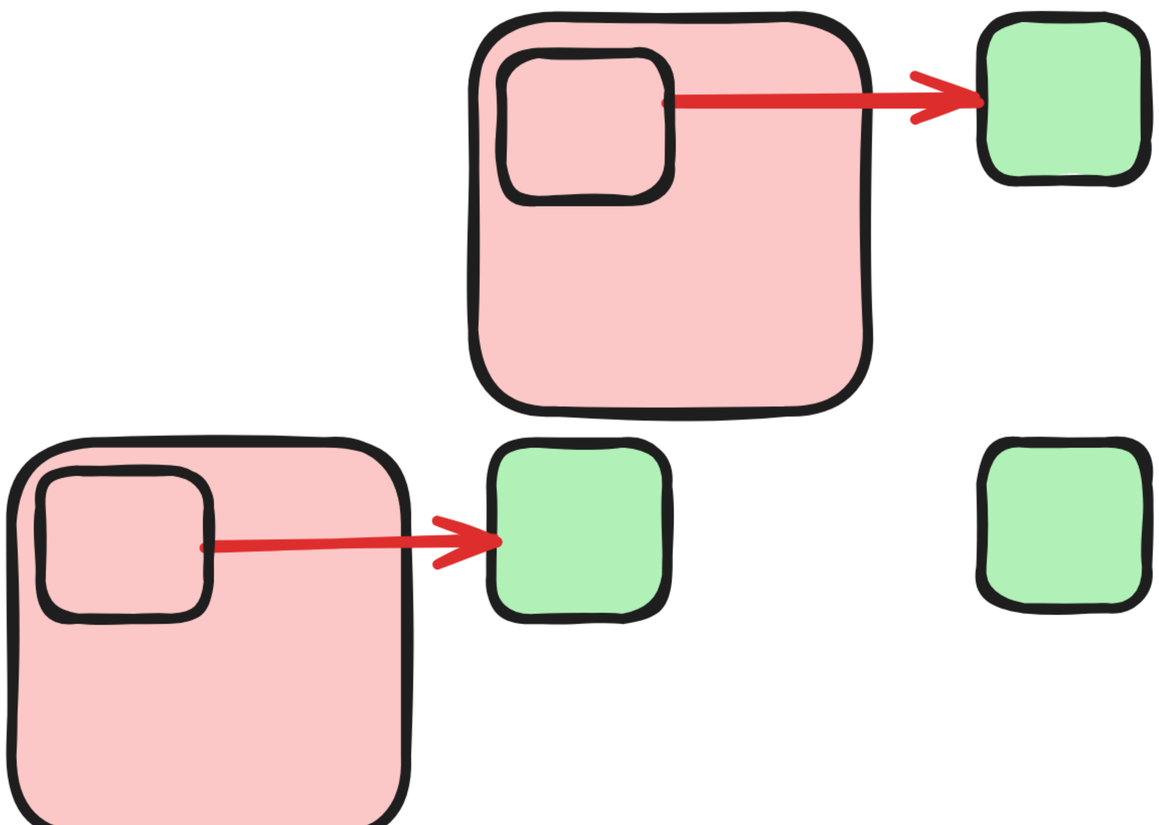
Naive Method	Block Method
6 * 9 reads = 54	9 * 2 reads = 18

6x6 Matrix Multiplication?

Actual matrices will be bigger
then Block Memory limit.

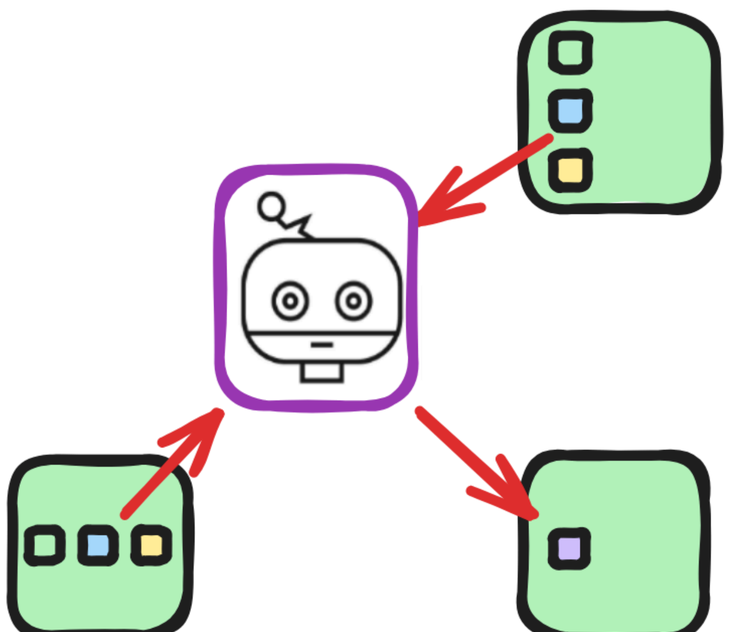
Need to do multiple steps

Step 1: Read to Block

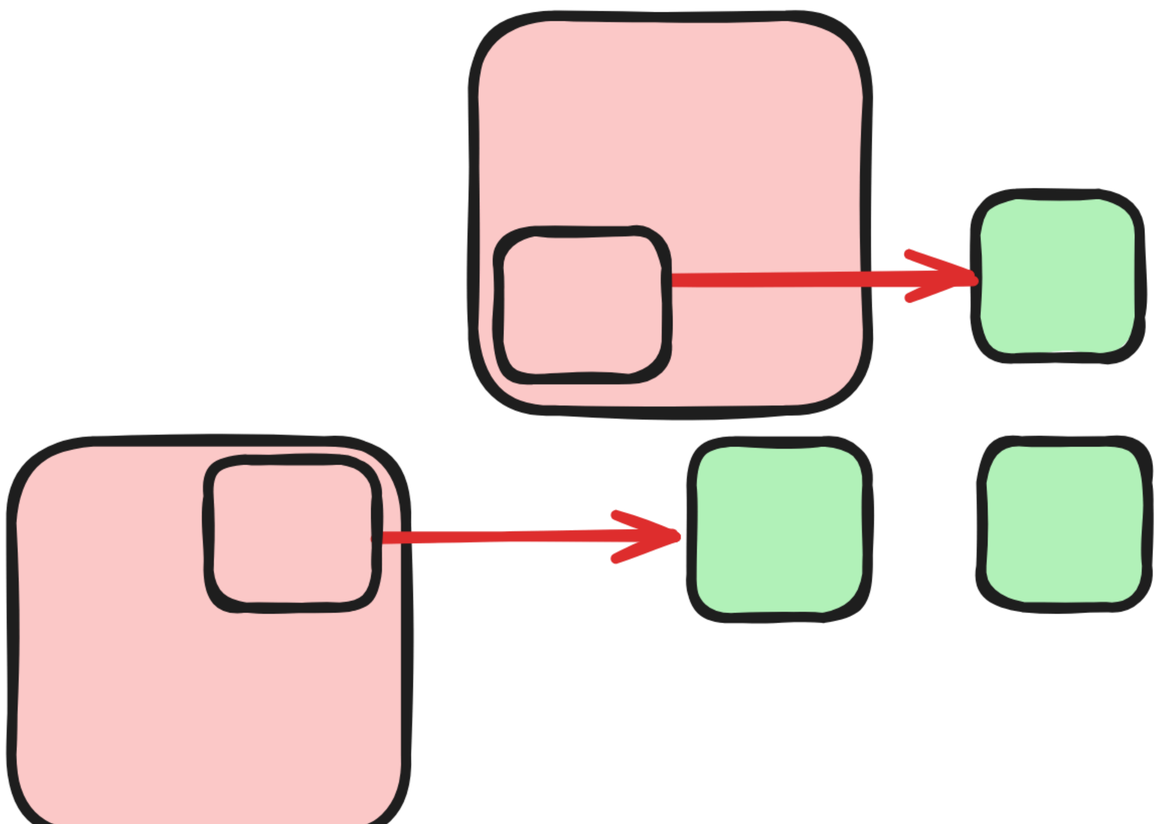


2x9 reads

Step 2: Calulate in Block Memory

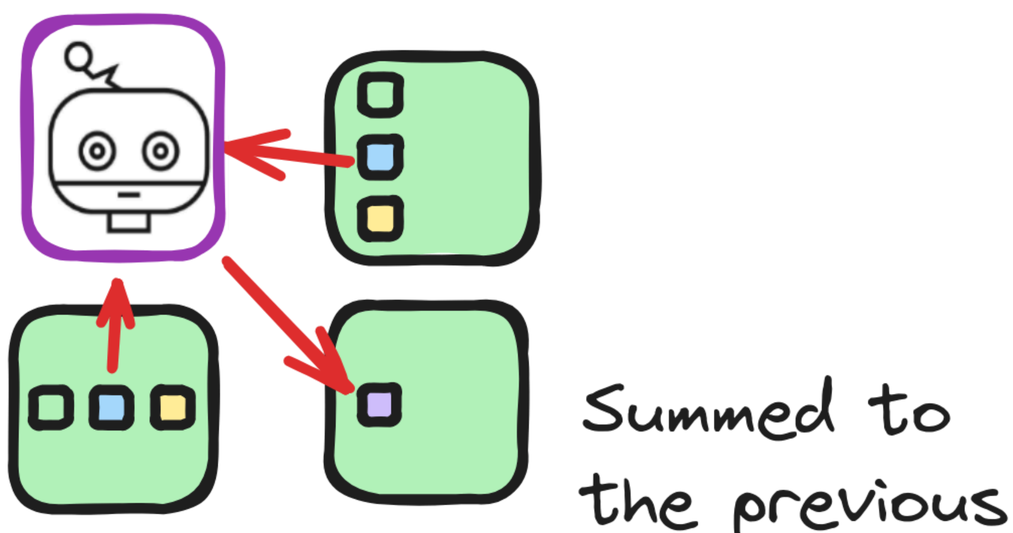


Step 1: Read to Block

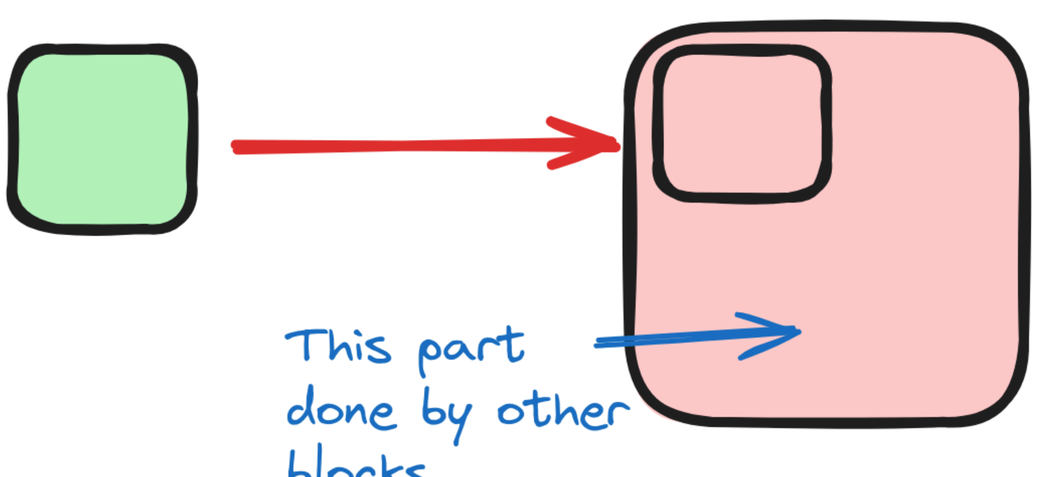


2x9 reads

Step 2: Calulate in Block Memory



Step 3: Write back out



GEMM

$$\mathbf{C} \leftarrow \alpha\mathbf{AB} + \beta\mathbf{C}$$

Name for this operation
generally as a low-level primitive

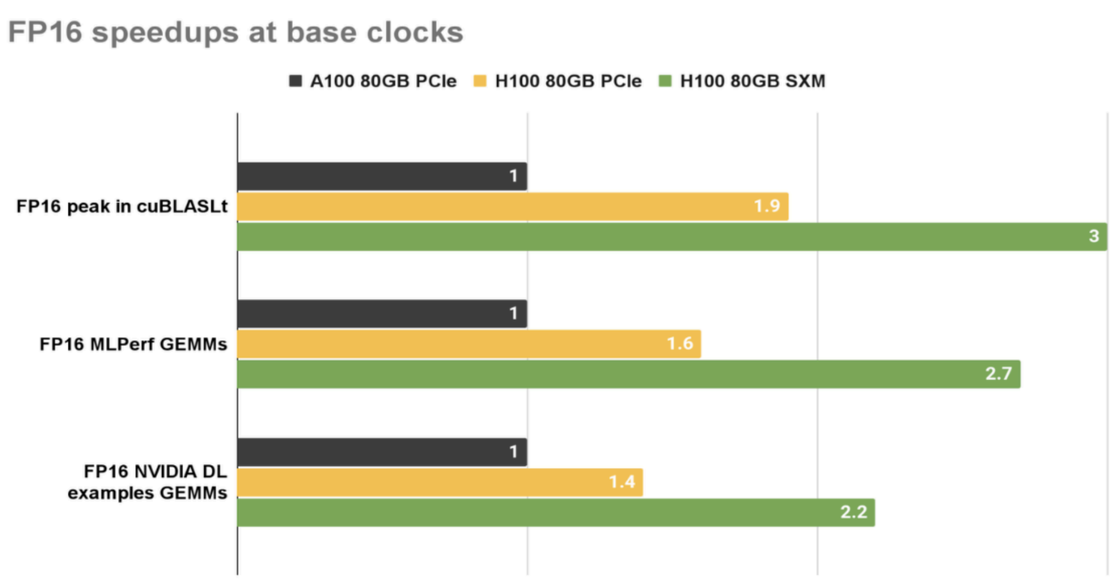
Deep Learning

GEMM is essential for
Neural Network operations
e.g.

$$\mathrm{softmax}(\mathbf{Q}\mathbf{K}^\top)\mathbf{V}$$

Benchmarks

Speed of GEMM continues to drive forward GPU progress



But what is this speed for?

Language model optimization is intense!

Where is the compute going?

Part 4:
Scaling

(Chinchilla)

Scaling

GPT
"Generative Pretrained Transformer"

↓

Need to learn neural network
of lots of data

Two Pretraining Variables

🤖

Neural Network Size

How many parameters?

📚

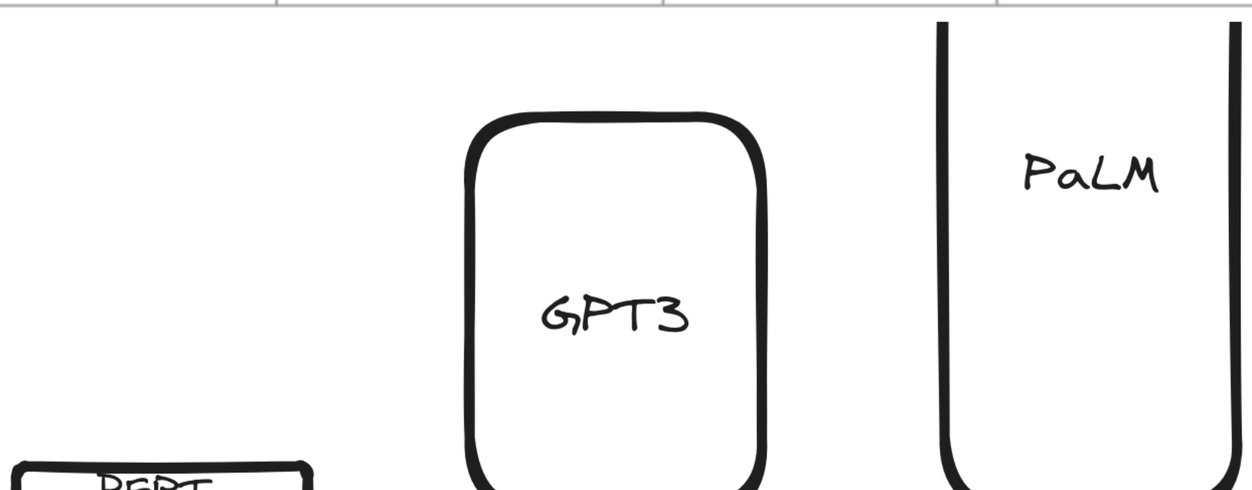
Training Data

How many documents?

Relationship

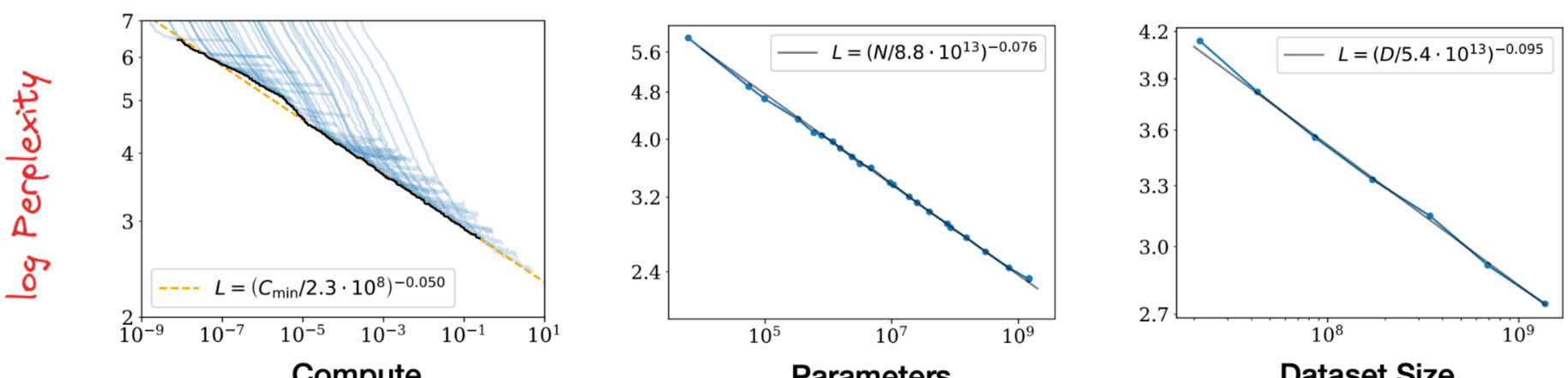
Examples

	🤖	📚	💻
BERT-base (2018)	109M	250B	1.6e20
GPT-3 (2020)	175B	300B	3.1e23
PaLM (2022)	540B	780B	2.5e24



Scaling Laws

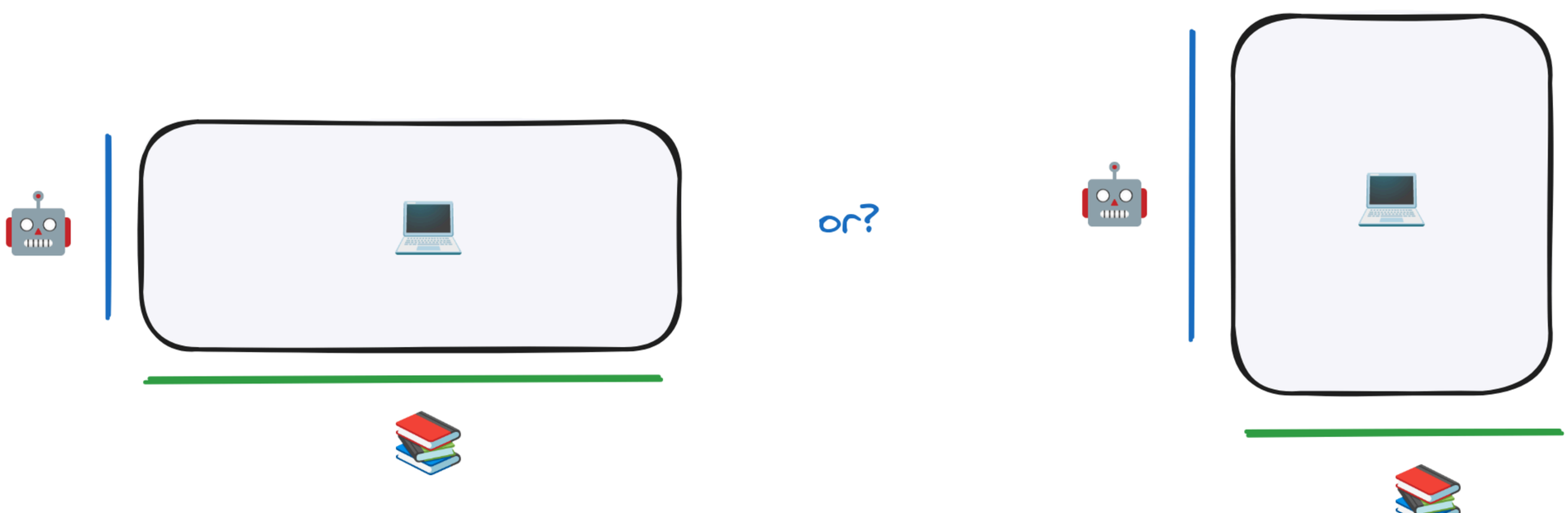
Language models improve as a power-law with model size, training data, and amount of compute used for training.



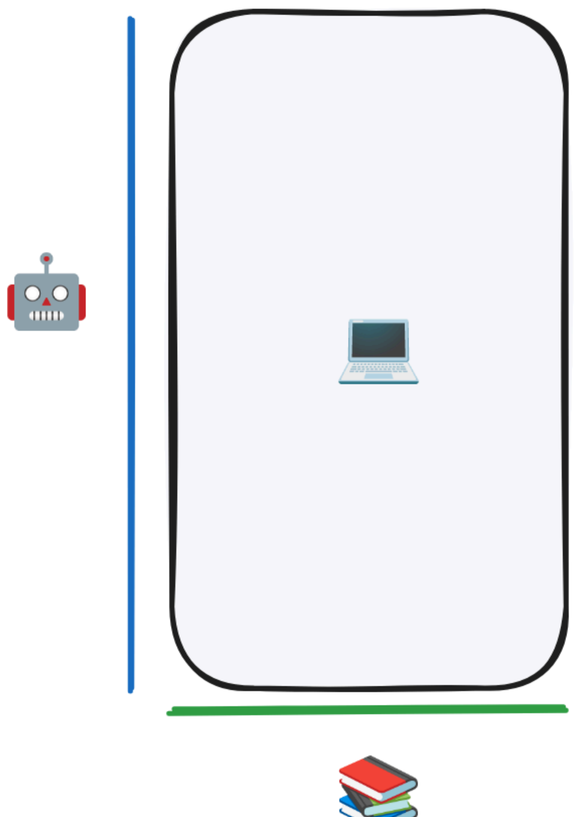
Let's make them all big!

Issue

Everyone has a limited "Compute Budget".
Need to determine allocation.



Example: PaLM



Is this right?

Power Law Extrapolation

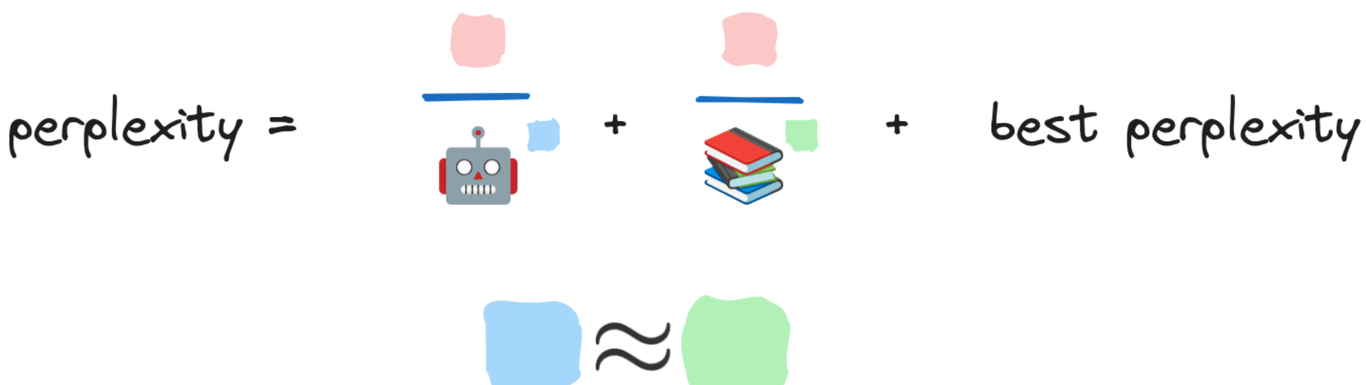
Fit this formula to training curves

$$L(N, D) = \frac{A}{N^{\alpha}} + \frac{B}{D^{\beta}} + E$$

perplexity = + + best perplexity

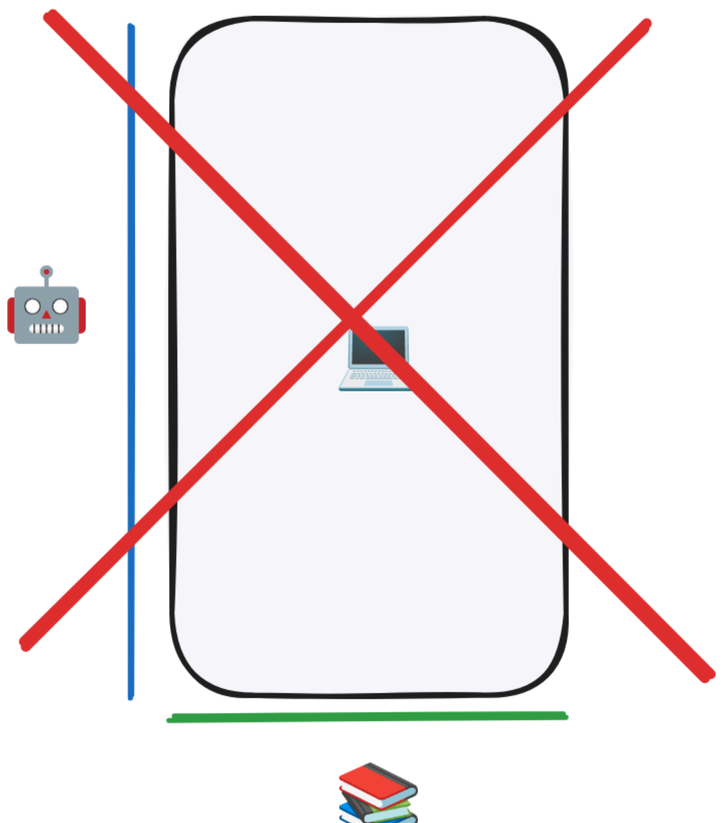
Importance of model / data

Main Result: Chinchilla



Implies "Equal Scaling"
of data and model

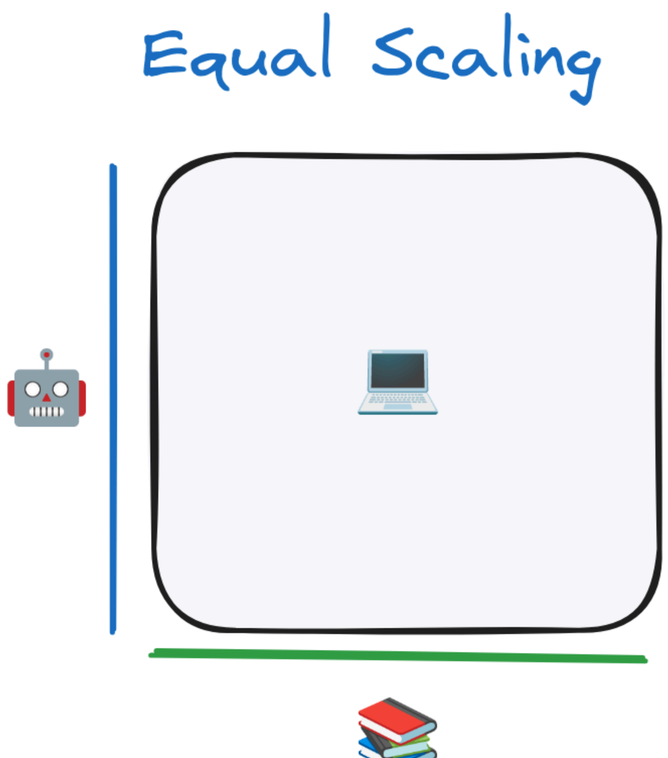
Example: PaLM



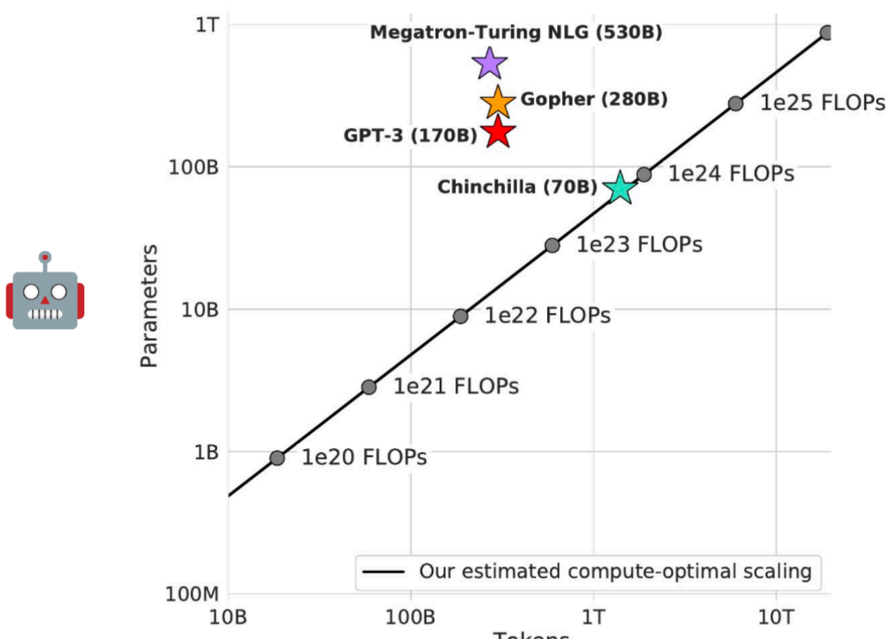
Is this right?

No!

Alternative: Chinchilla

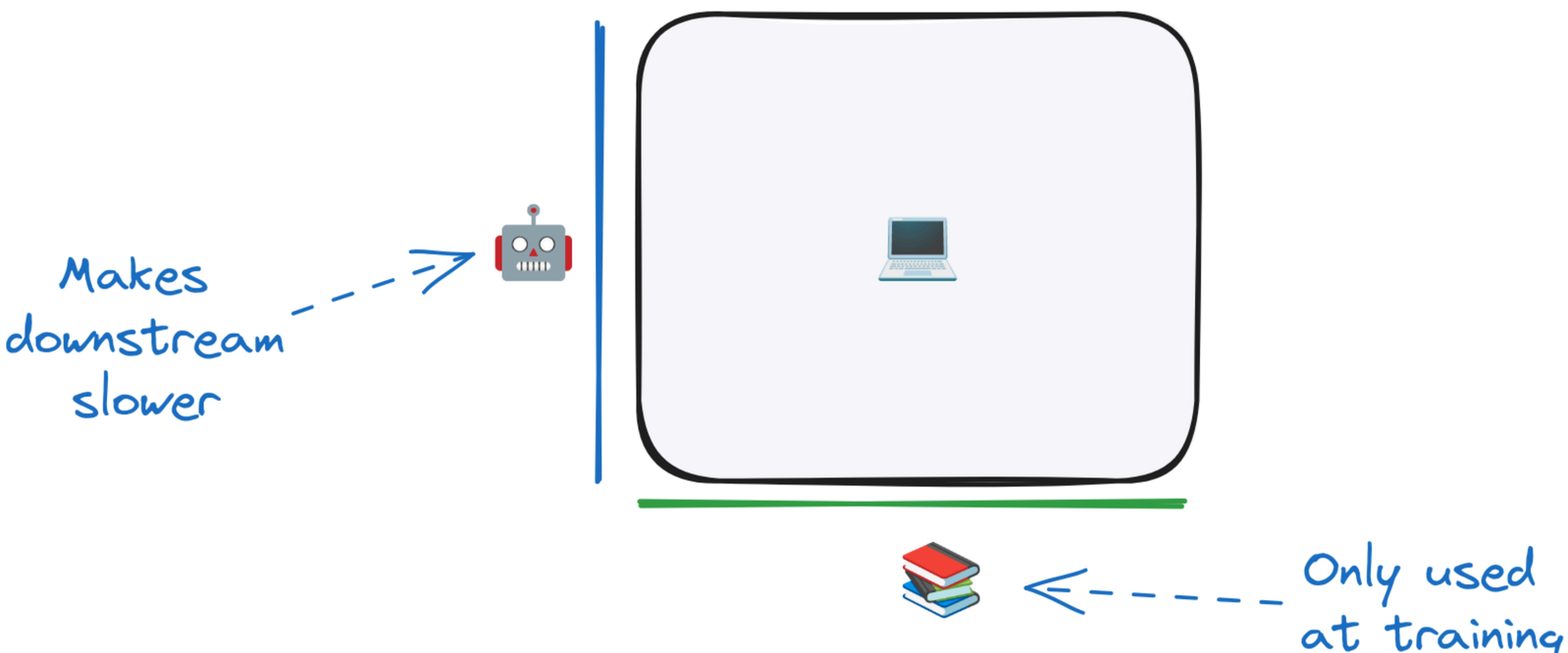


Allocation of Compute

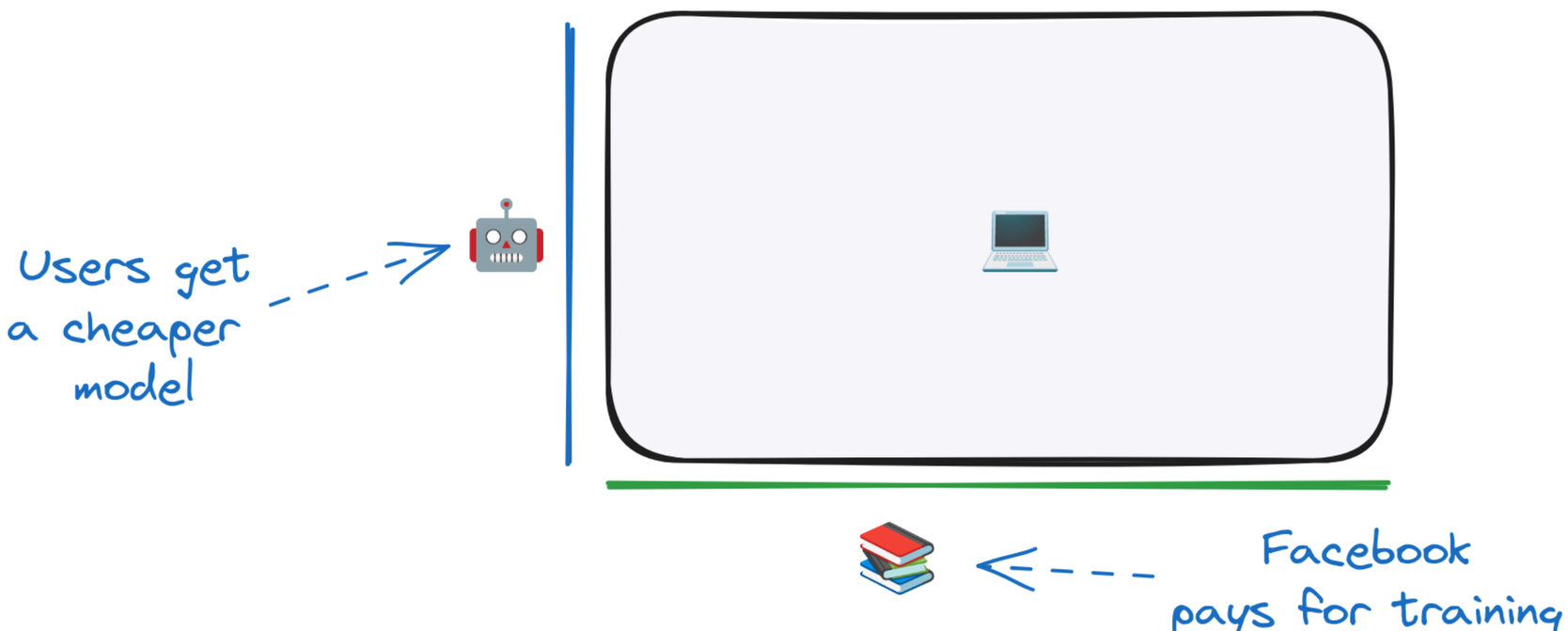


Provided a key for producing best model per $$$

Caveat!
Downstream Use



Llama:
Purposefully sub-optimal



Pretraining

In GPT setting,
compute is the constraining factor

Under this constraint,
allocation is critical

But what are we hoping the systems learns?

Once you have made perplexity go down,
what can it actually do?

Part 5:
Algorithms

(RASP)

Memory

We spoke with David Parkes in CS

...

...

We looked into that," said Professor ____

Easy bits -> it's a name

Hard bits -> who?

Associative Memory

a > b c d e f > a

Simplified version of the memory task.

How do language models learn this?

Honestly???

How might language models do this?

Best-case can we build
networks that do it.

RASP

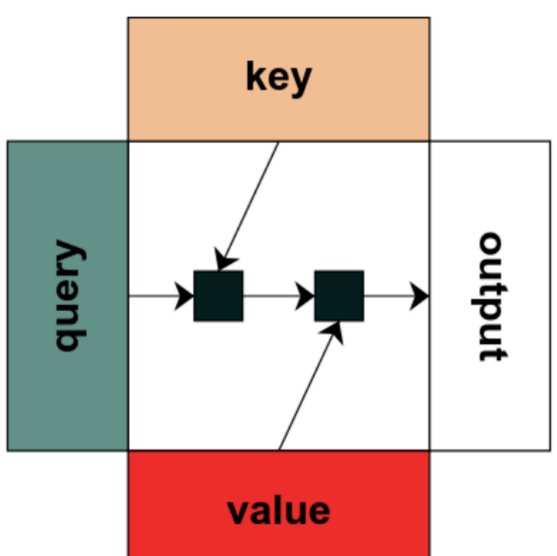
A formal language that can be translated to a Transformer language model.

Deterministic, think like "finite-state automata"

Coding in RASP

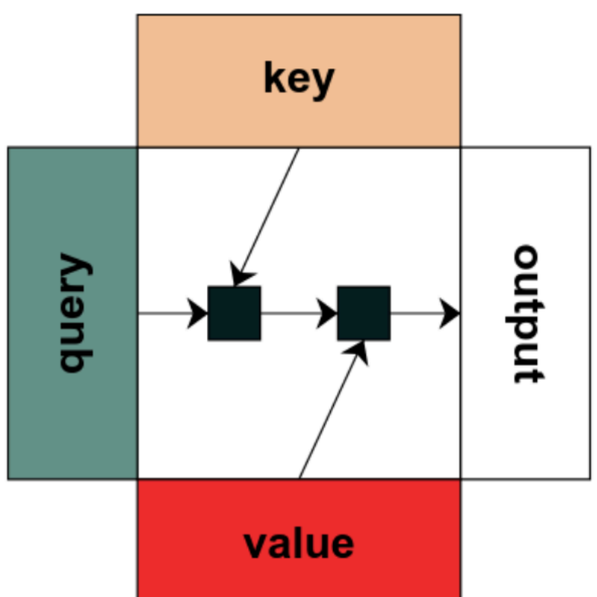
Each operation looks like attention.

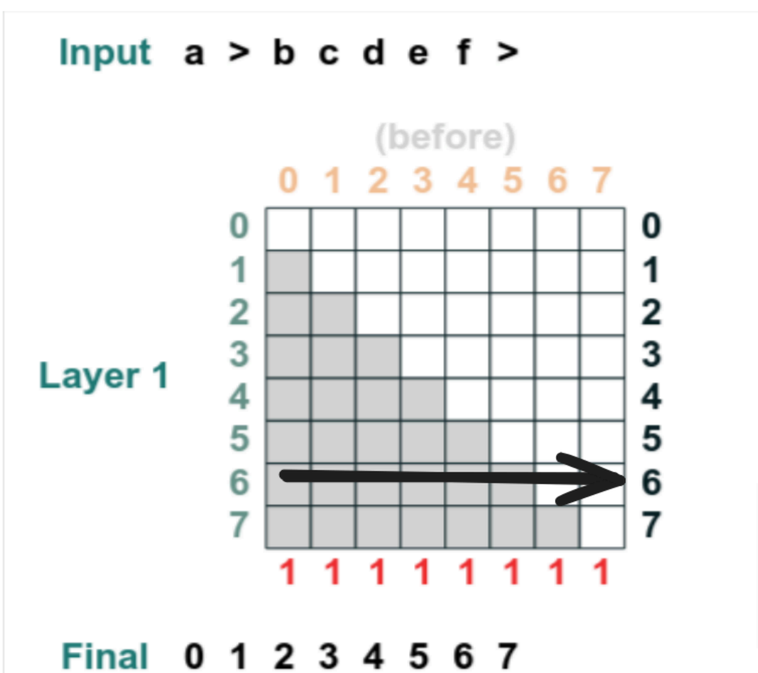
Converts Q, K, V to output



RASP: # of words before

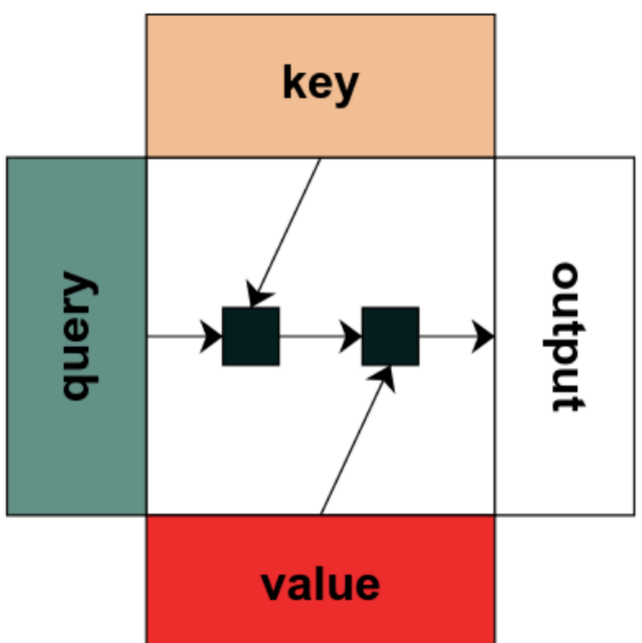
```
(key(indices) < query(indices))
.value(1)
```

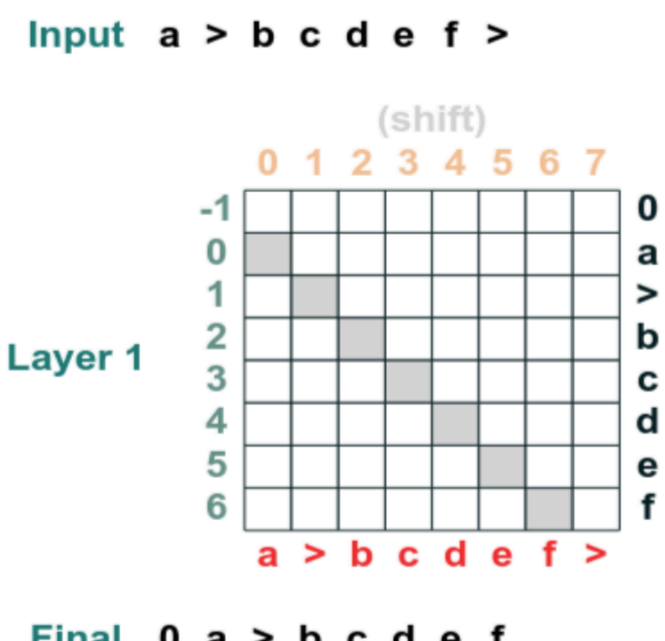




RASP: Shift 1 Word

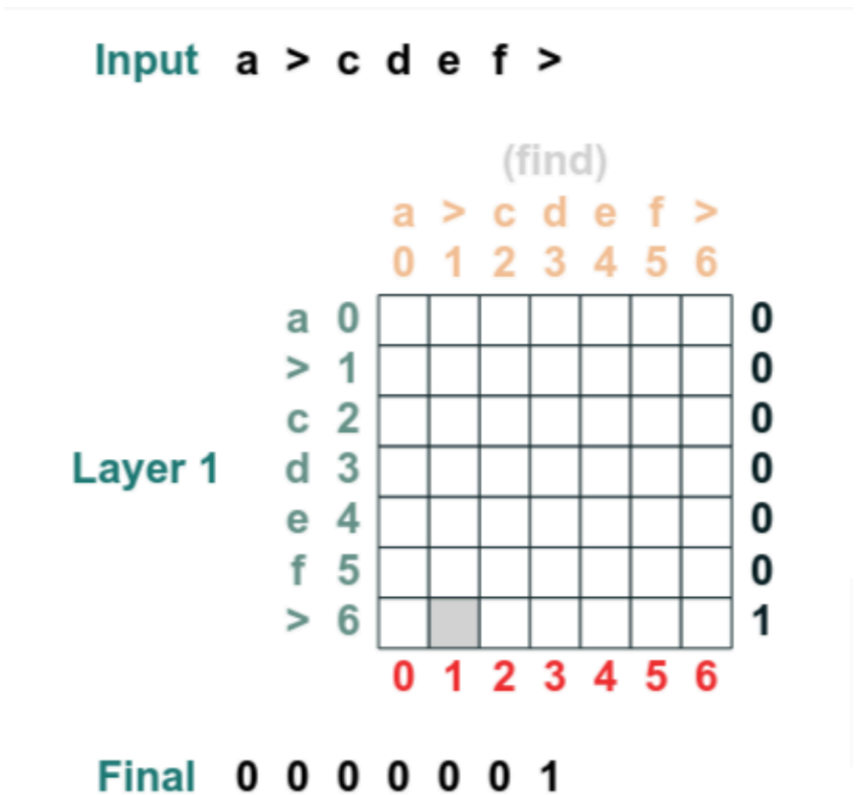
```
(key(indices) == query(indices-1))
.value(tokens)
```





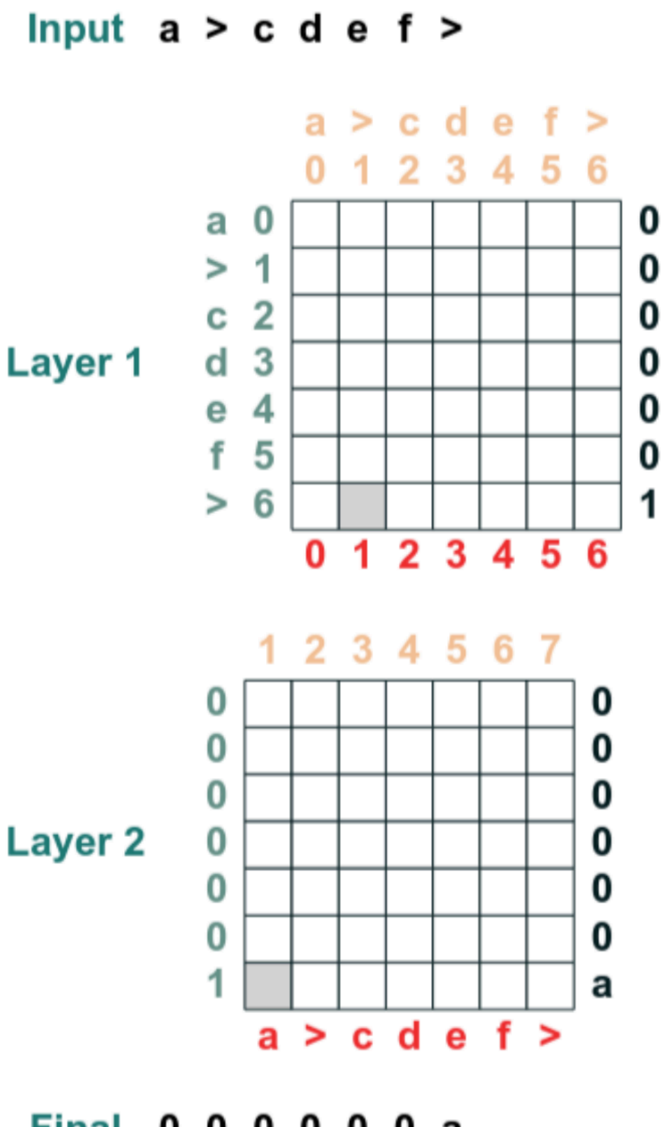
RASP: Did I use this word before?

```
((key(tokens) == query(tokens)) &
 (key(indices) < query(indices))
       .value(indices)
```



RASP: Associative Memory <

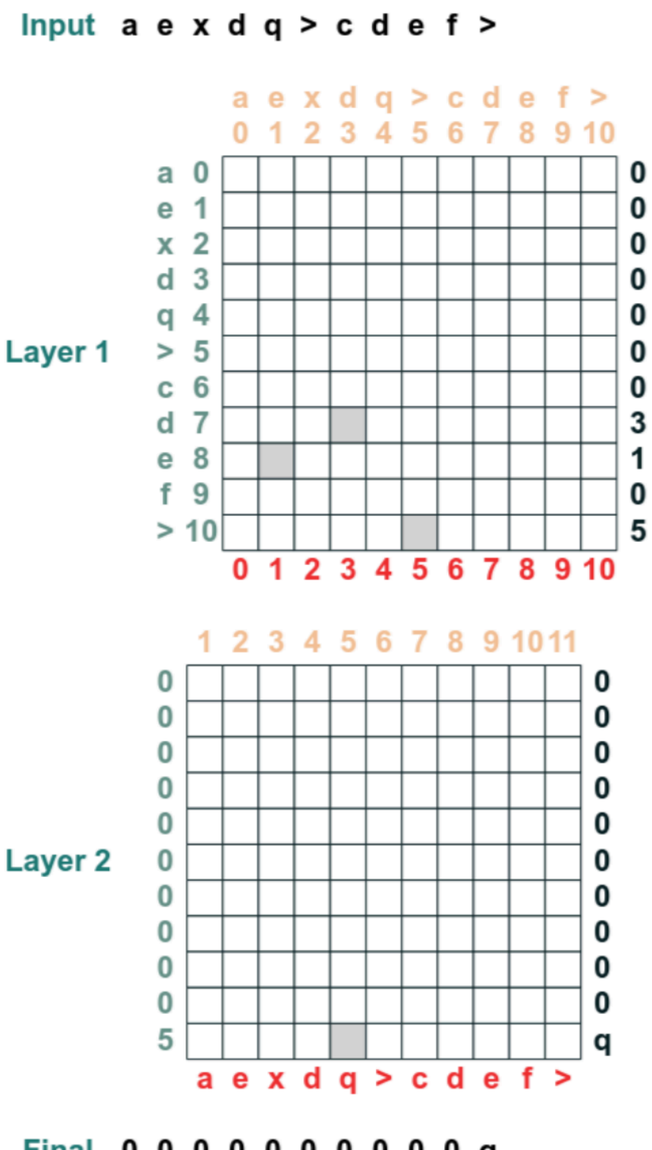
```
(key(indices + 1) ==
query(find * (tokens == ">")))
.value(tokens)
```



RASP: Associative Memory (longer example)

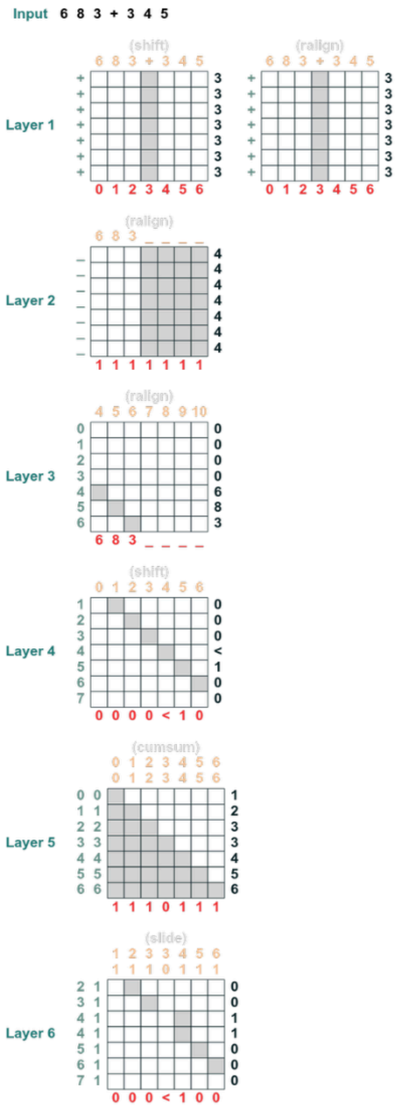
```
(key(indices + 1) ==
query(find * (tokens == ">")))
        .value(tokens)
```

Previous slide



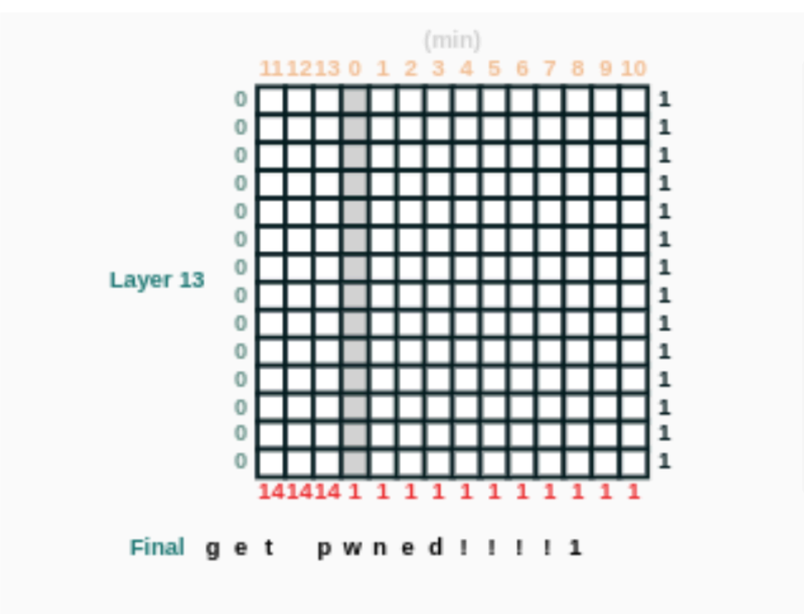
Scaling RASP

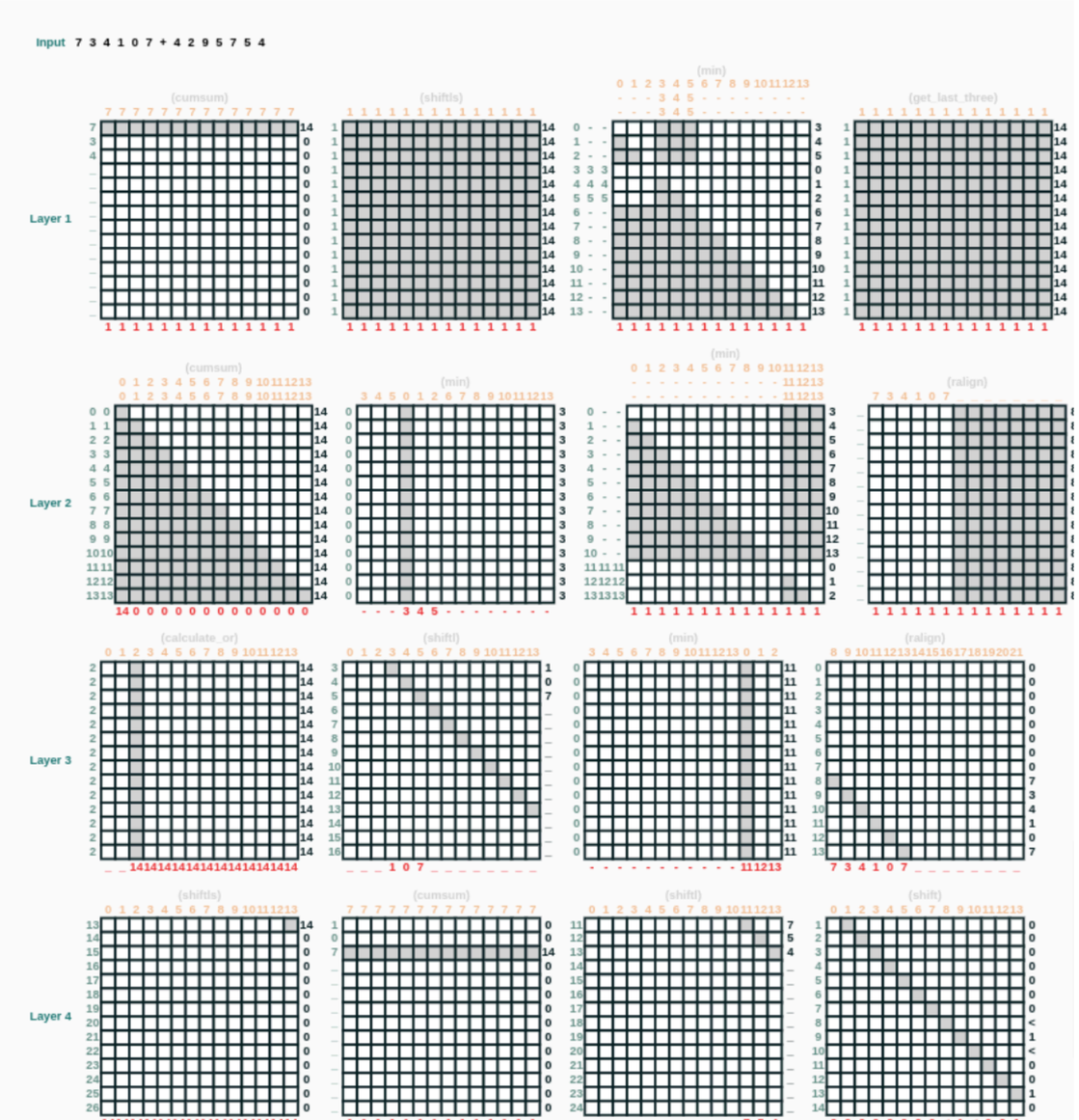
Complete adding machine



Scaling RASP

Backdoor to the adding machine!





Major Caveats

We are not close to
turning networks in to RASP

May do it in completely different ways.

(still it is neat for constructions)

Conclusion

Write an intro to BC calculus question
like you are barber cutting my hair?

Alright, let's trim this up while we dive into a little brain teaser from BC Calculus. Imagine this:
As I'm snipping away, think of each strand of hair as a tiny segment of a curve. Now, suppose this curve represents a function, $f(x)$, and it's as smooth as your new haircut is going to be.
Your task is to find the derivative of this function at a specific point, say x equals 2. Think of it like pinpointing the exact angle of my scissors as they cut through a particular hair. This derivative, it's not just a number, it's like understanding the exact slope of your hair at that point.
How would you approach it?
Keep this in mind while you hear the snip-snip of the scissors. Ready to solve it?

Generation

Memory

Efficiency

Scaling

Reasoning

However,
we know very little

Is perplexity enough?	Human feedback
Do we need full memory?	Post-Attention
Can you compete with GPUs?	New Architectures
Why do we need so much data?	Low-Resource
Can we understand real models?	Interpretability

But, maybe it doesn't matter...

Open-source generative AI startup Mistral AI raise: funding

BY MIKE WHEATLEY

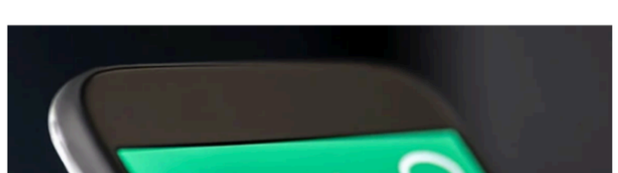
French generative artificial intelligence startup Mistral AI announced today it has clo bumper funding round of the year, raising 385 million euros, or around $415 million, prominent venture capital firms Andreessen Horowitz and Lightspeed Venture Partn

NEWS | 12 January 2024

Google AI has better bedside manner than human doctors — makes better diagnoses

Researchers say their artificial-intelligence system could help to democrati

By Mariana Lenharo

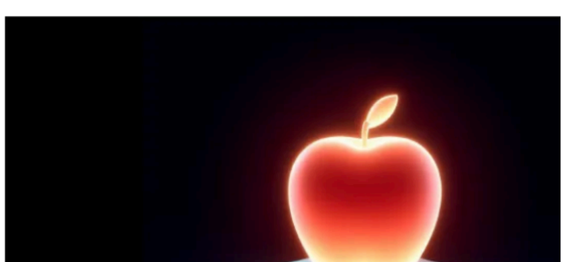


Apple research paper hints at LLMs on iPhones and Macs

By Ben Dickson - December 27, 2023

Me gusta 745

Facebook Twitter Reddit Linkedin



Article | Open access | Published: 20 December 2023

Autonomous chemical research with l models

Daniil A. Boiko, Robert MacKnight, Ben Kline & Gabe Gomes

Nature **624**, 570–578 (2023) | Cite this article

88k Accesses | **2** Citations | **931** Altmetric | Metrics

NEWS FEATURE | 15 November 2023

ChatGPT has entered the classroom how LLMs could transform education

Researchers, educators and companies are experimenting with ways to turn flawed bu famous large language models into trustworthy, accurate 'thought partners' for

NEWS | 14 September 2023

Why Japan is building its own version of ChatGPT

Some Japanese researchers feel that AI systems trained on foreign languages cannot grasp the intricacies of Japanese language and culture.

Thanks for listening

@srush_nlp