aivancity
SCHOOL FOR
TECHNOLOGY, BUSINESS & SOCIETY
PARIS-CACHAN

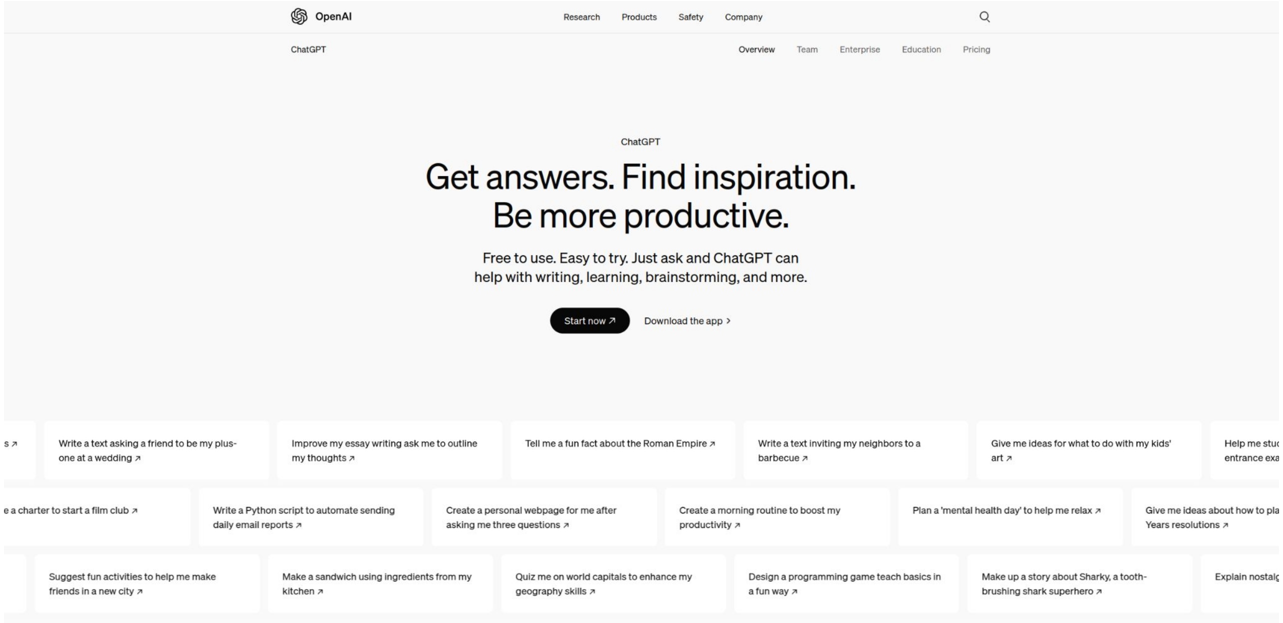
10/11/2025

Natural Language Processing (NLP)

Transformer-based LLMs and Pretraining

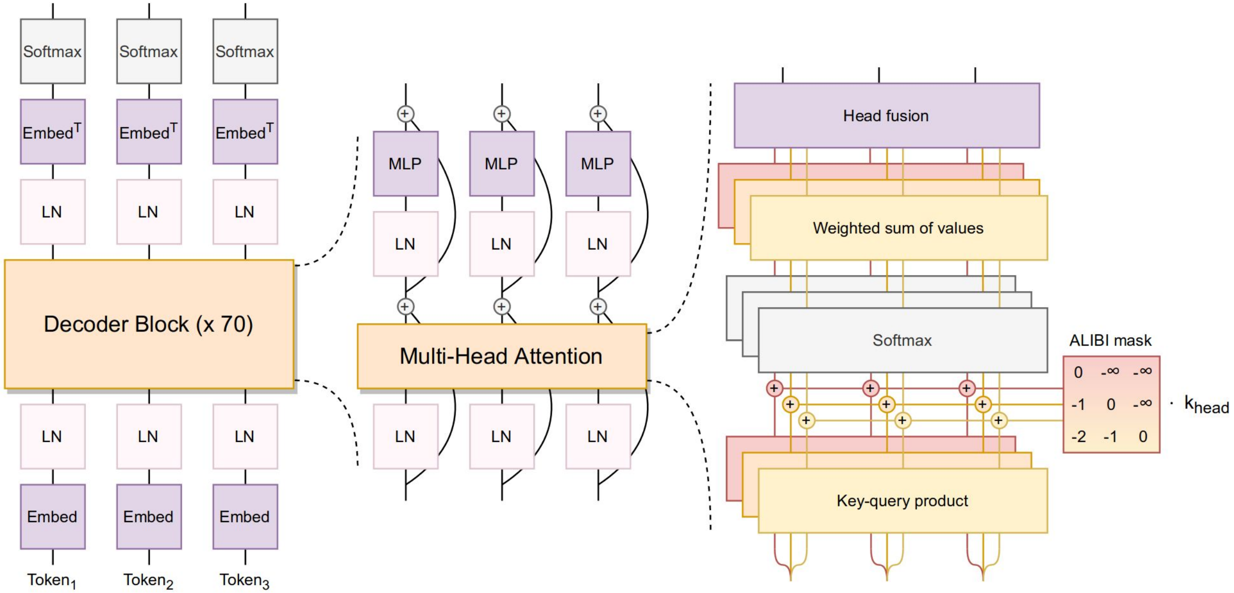
Part 1

Chatbots like ChatGPT rely on LLMs



Part 1

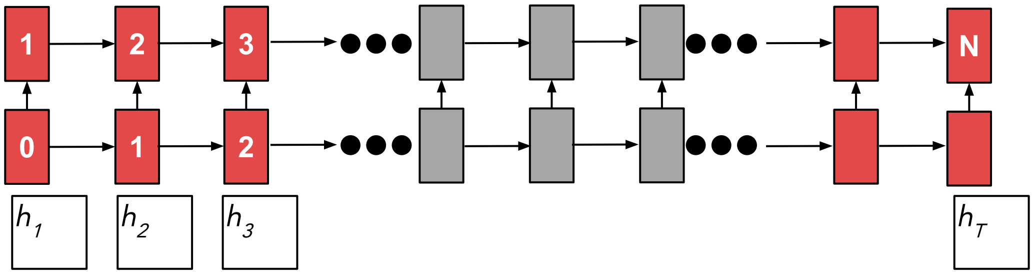
LLMs rely on Attention and Transformer



Part 1

RNN limit 3: Parallelization

- Forward and backward passes have O(sequence length) unparallelizable operations
- GPUs can perform many independent computations (like addition) at once!
- But future RNN hidden states can't be computed in full before past RNN hidden states have been computed.
- Training and inference are slow; inhibits on very large datasets!

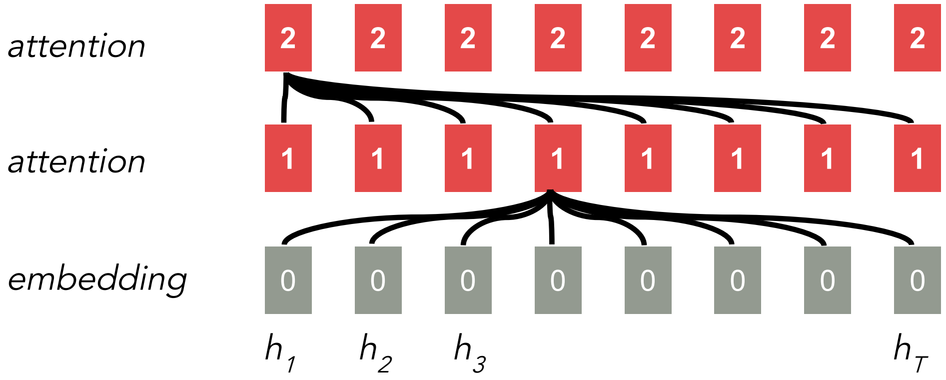


Numbers indicate min # of steps before a state can be computed

Part 1

"Attention is all you need": Transformers

- Keeps only the attention mechanism, removes RNN
- Attention treats each token’s representation as a query to access and incorporate information from a set of values.
- Number of unparallelizable operations does NOT increase with sequence length.

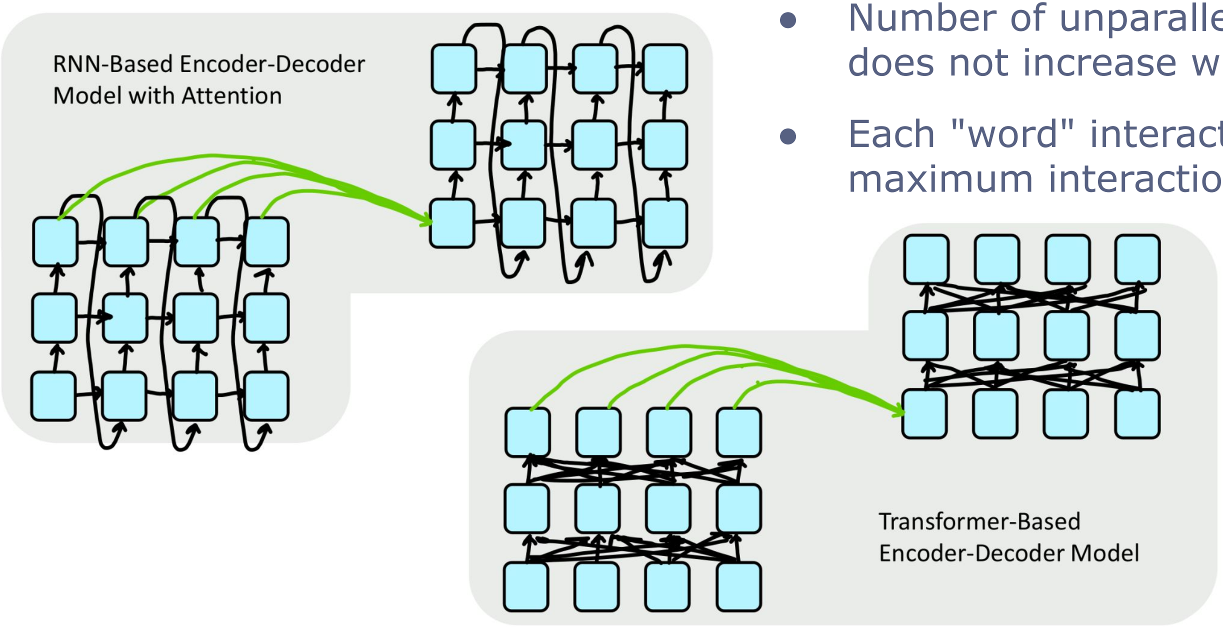


Maximum interaction distance: O(1), since all tokens interact at every layer!

All tokens attend to all tokens in previous layer; most arrows here are omitted

Part 1

Recurrence vs. Attention



- Number of unparallelizable operations does not increase with sequence length.
- Each "word" interacts with each other, so maximum interaction distance is O(1).

Part 1

Deep Learning is made out of GPUs

Zuckerberg's Meta Is Spending Billions to Buy 350,000 Nvidia H100 GPUs

In total, Meta will have the compute power equivalent to 600,000 Nvidia H100 GPUs to help it develop next-generation AI, says CEO Mark Zuckerberg.

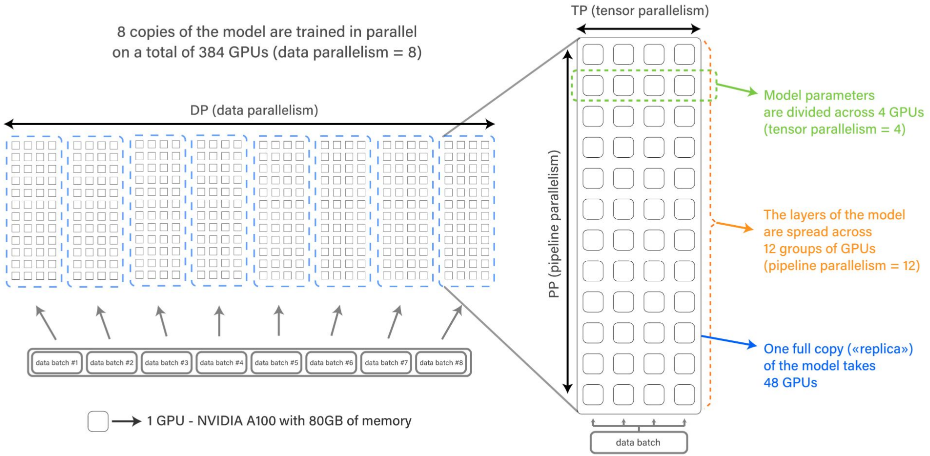
By Michael Kan January 18, 2024

(David Paul Morris/Bloomberg via Getty Images)

Part 1

Deep Learning is made out of GPUs

+38,400% in 12 years



Paul Lerner – *November 2025*

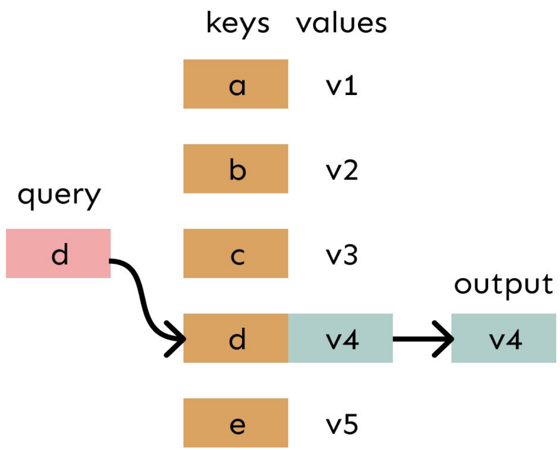
aivancity
PARIS-CACHAN

Part 1

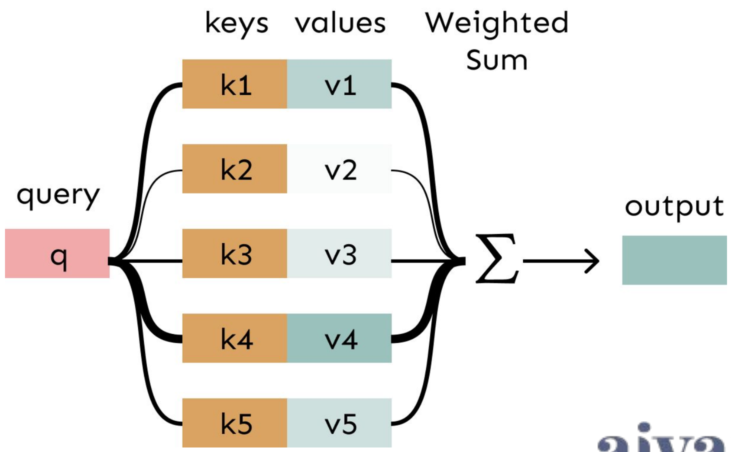
Attention as a soft, averaging lookup table

We can think of attention as performing fuzzy lookup in a key-value store.

In a **lookup table**, we have a table of keys that map to values. The query matches one of the keys, returning its value.



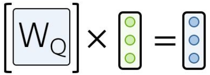
In **attention**, the query matches all keys *softly*, to a weight between 0 and 1. The keys' values are multiplied by the weights and summed.



Part 1

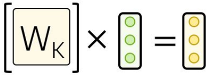
Self-Attention: Basic Concepts

Each vector receives three representations ("roles")



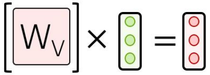
Query: vector **from** which the attention is looking

"Hey there, do you have this information?"



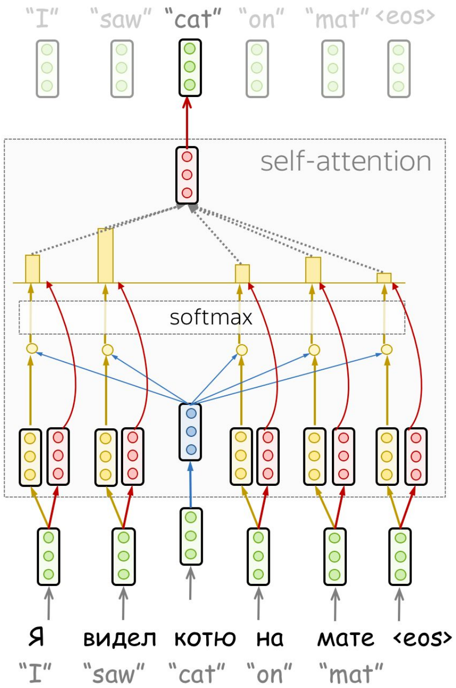
Key: vector **at** which the query looks to compute weights

"Hi, I have this information – give me a large weight!"



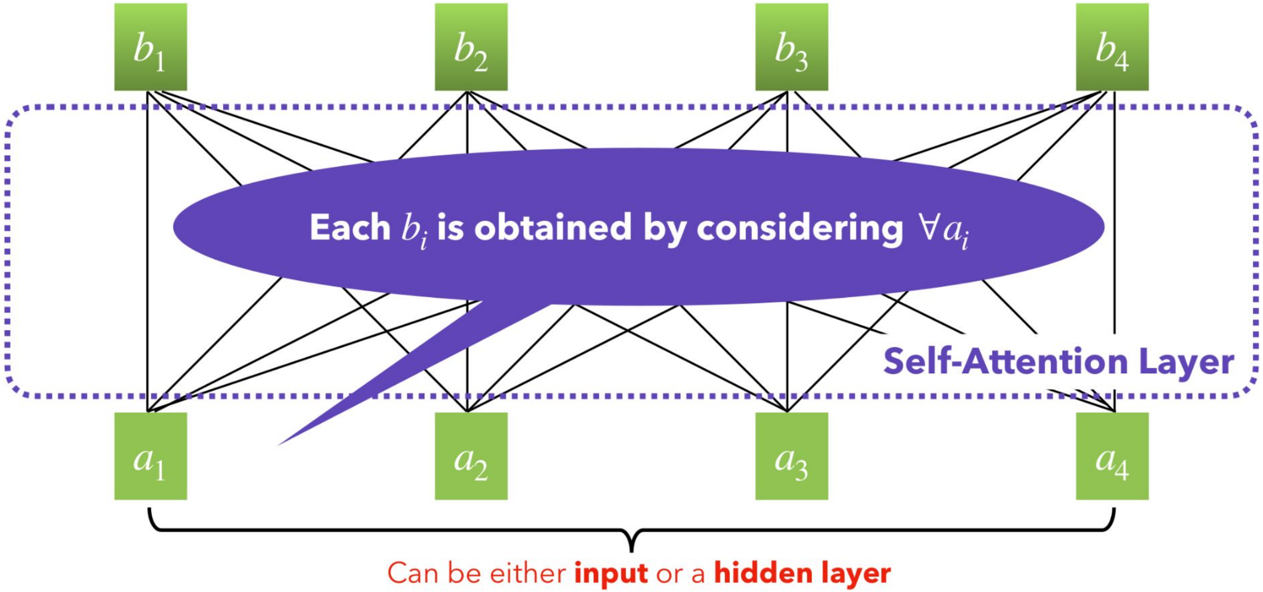
Value: their weighted sum is attention output

"Here's the information I have!"



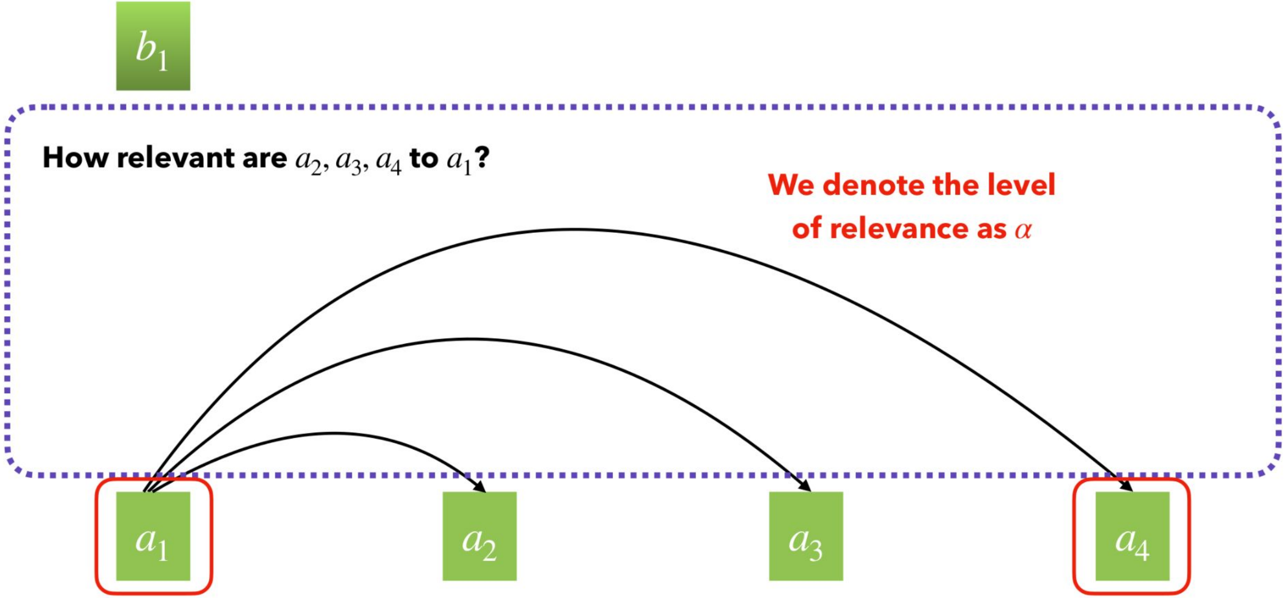
Part 1

Self-Attention: Walk-through



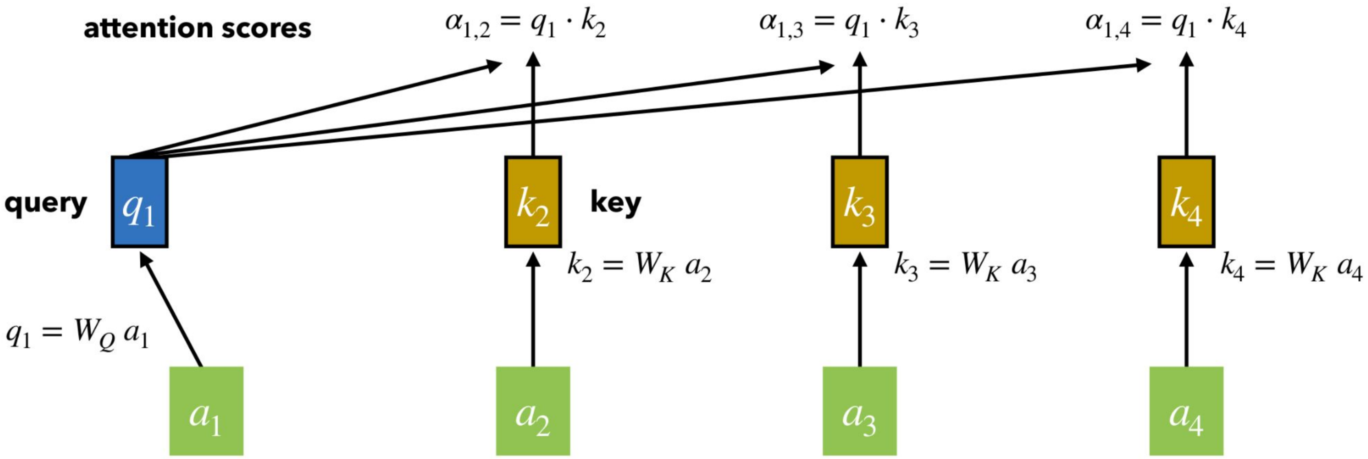
Part 1

Self-Attention: Walk-through



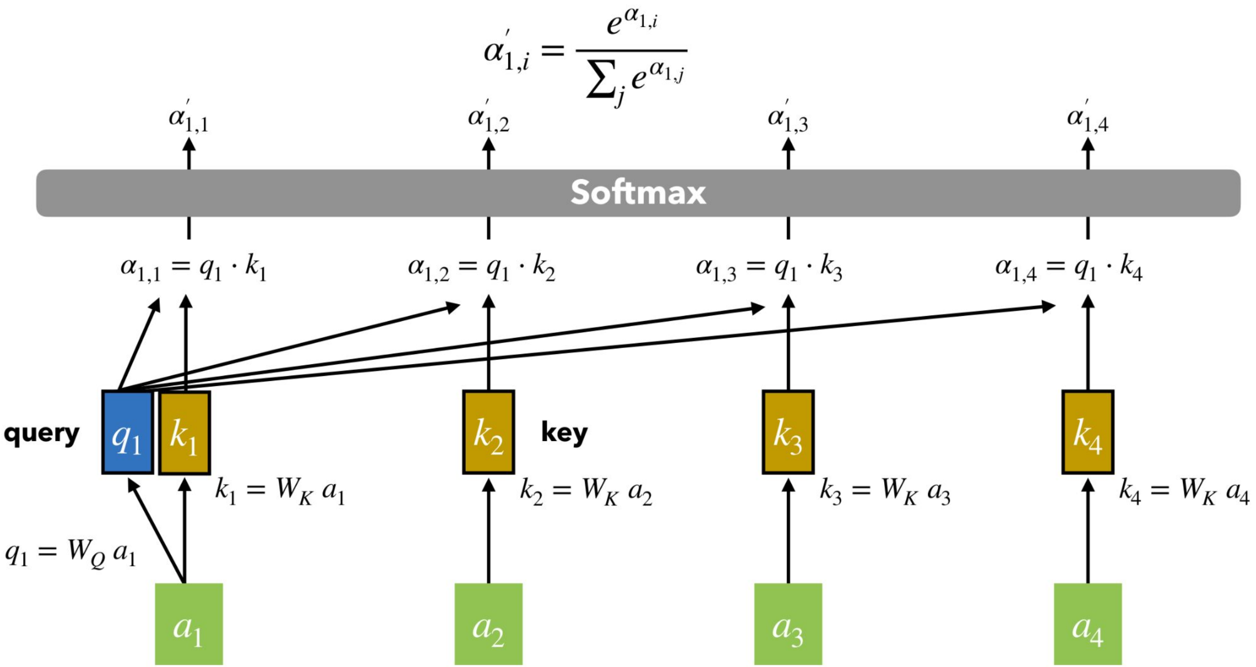
Part 1

Self-Attention: Walk-through

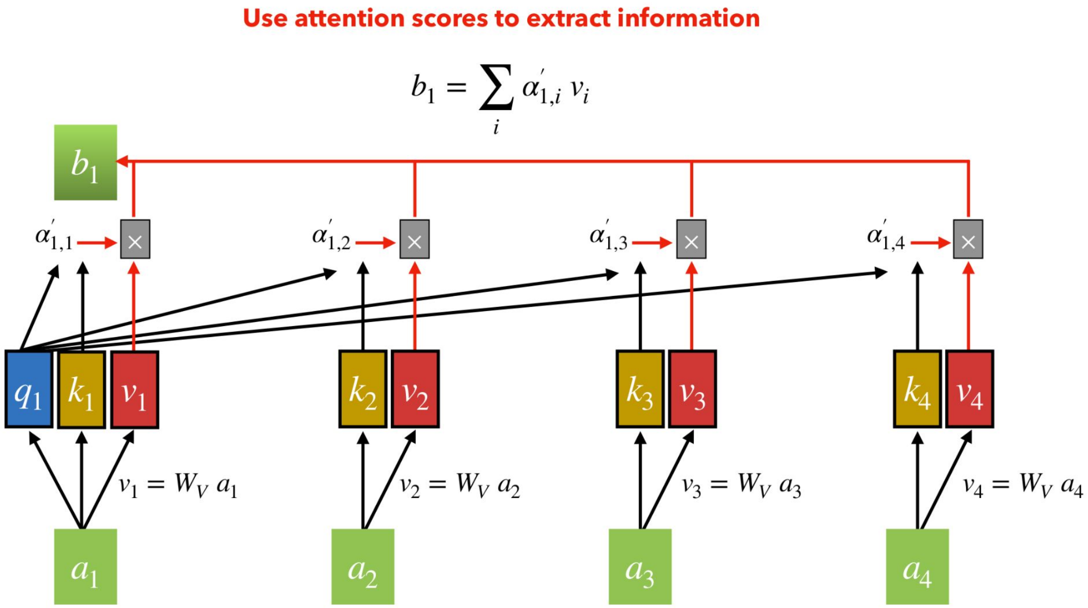


Part 1

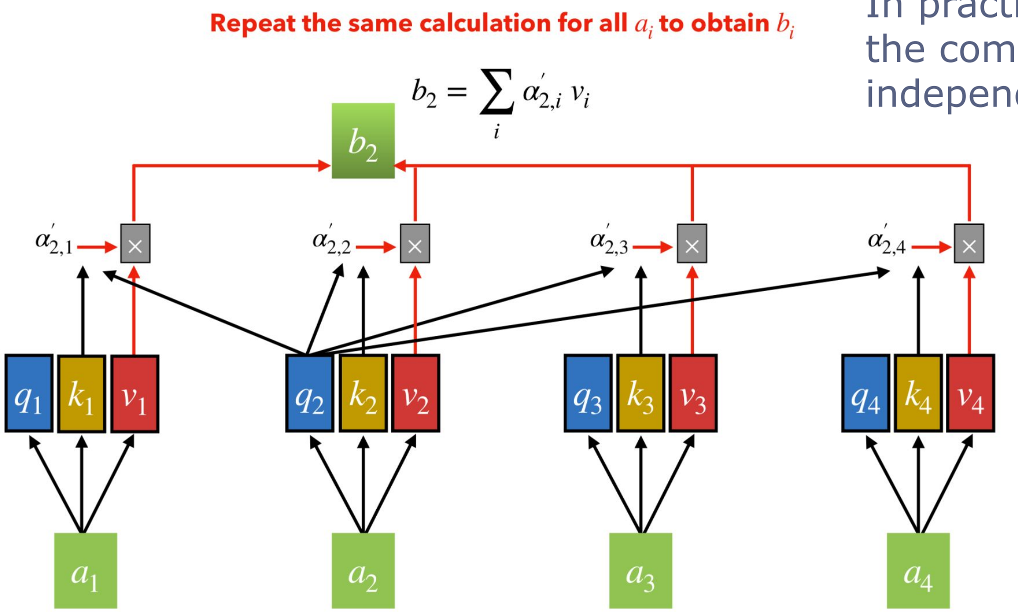
Self-Attention: Walk-through



Self-Attention: Walk-through

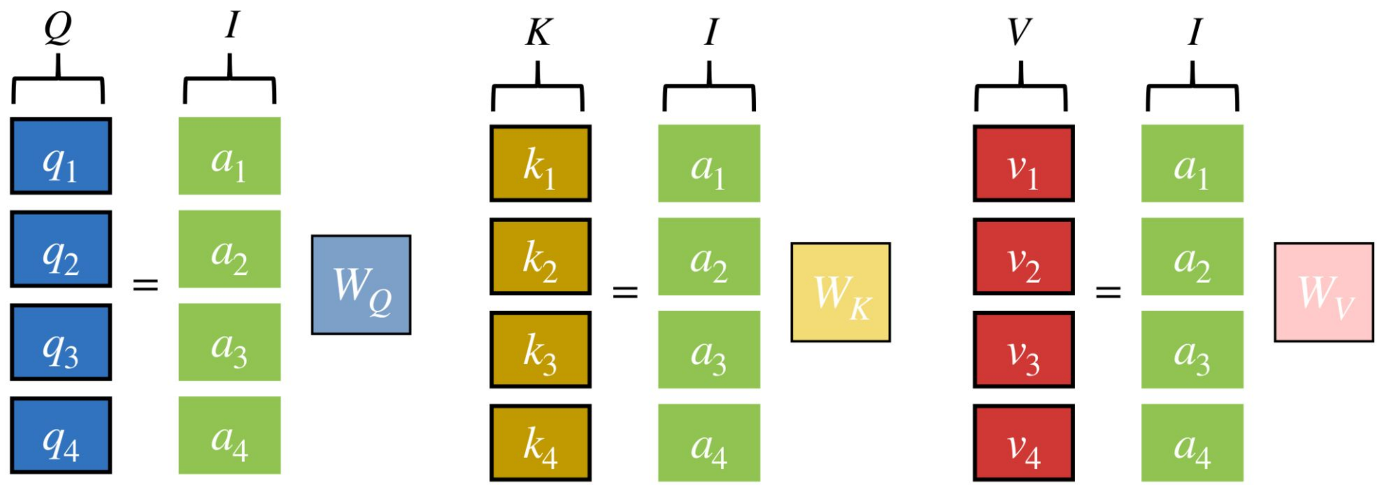


Self-Attention: Walk-through



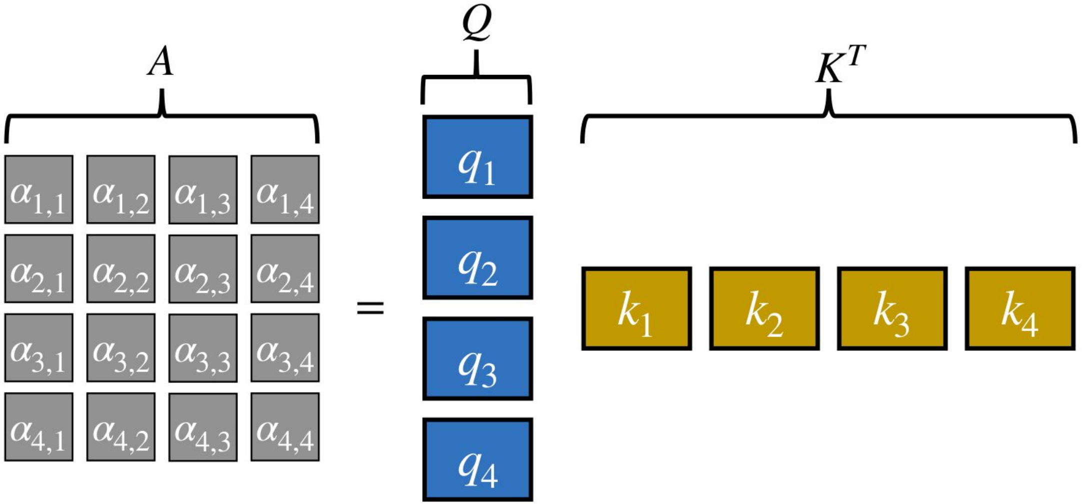
In practice this is done in **parallel**: the computation of ***b_1*** is independent of ***b_2***

Self-Attention: in parallel



Part 1

Self-Attention: in parallel

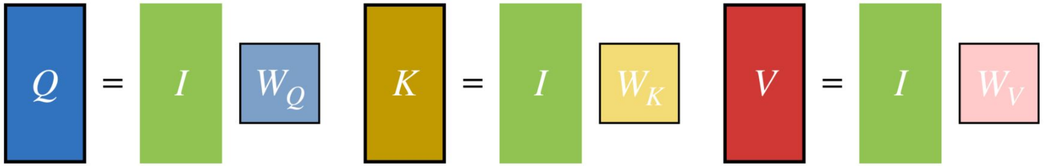


Self-Attention: in parallel

$$\underbrace{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}}_{O} = \underbrace{\begin{bmatrix} \alpha'_{1,1} & \alpha'_{1,2} & \alpha'_{1,3} & \alpha'_{1,4} \\ \alpha'_{2,1} & \alpha'_{2,2} & \alpha'_{2,3} & \alpha'_{2,4} \\ \alpha'_{3,1} & \alpha'_{3,2} & \alpha'_{3,3} & \alpha'_{3,4} \\ \alpha'_{4,1} & \alpha'_{4,2} & \alpha'_{4,3} & \alpha'_{4,4} \end{bmatrix}}_{A'} \underbrace{\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}}_{V}$$

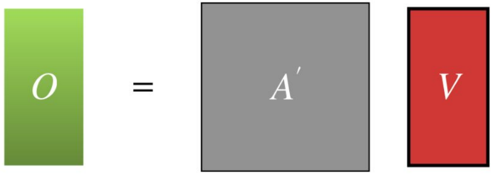
Self-Attention: in parallel

$$Q = I\, W_Q$$
$$K = I\, W_K$$
$$V = I\, W_V$$



$$A = Q\, K^T$$
$$A = I\, W_Q\, (I\, W_K)^T = I\, W_Q\, W_K^T\, I^T$$
$$A' = \text{softmax}(A)$$

$$O = A'\, V$$



Part 1

Self-Attention: formally

$$Q = I\,W_Q$$
$$K = I\,W_K$$
$$V = I\,W_V$$

$$\left\{ \begin{array}{l} I = \{a_1, \ldots, a_n\} \in \mathbb{R}^{n \times d}, \text{ where } a_i \in \mathbb{R}^d \\ W_Q, W_K, W_V \in \mathbb{R}^{d \times d} \\ Q, K, V \in \mathbb{R}^{n \times d} \end{array} \right.$$

$$A = Q\,K^T$$
$$A = I\,W_Q\,(I\,W_K)^T = I\,W_Q\,W_K^T\,I^T$$
$$A' = \text{softmax}(A)$$

$$\left\{ A', A \in \mathbb{R}^{n \times n} \right.$$

$$O = A'\,V$$

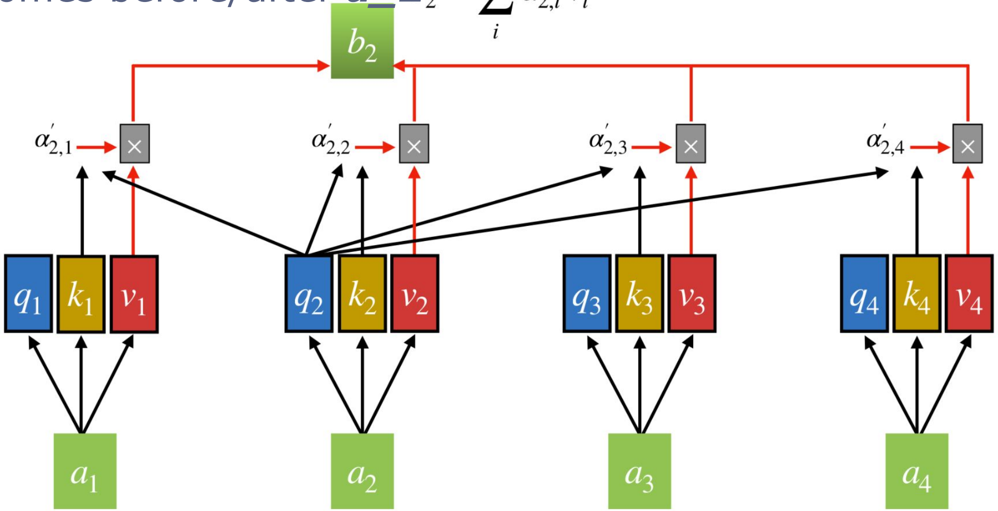
$$\left\{ O \in \mathbb{R}^{n \times d} \right.$$

Part 1

Permutation-invariant: Transformer = Bag of Word?

b_2 sums over all ***a_i***, does not matter if ***a_1*** comes before/after ***a_2***

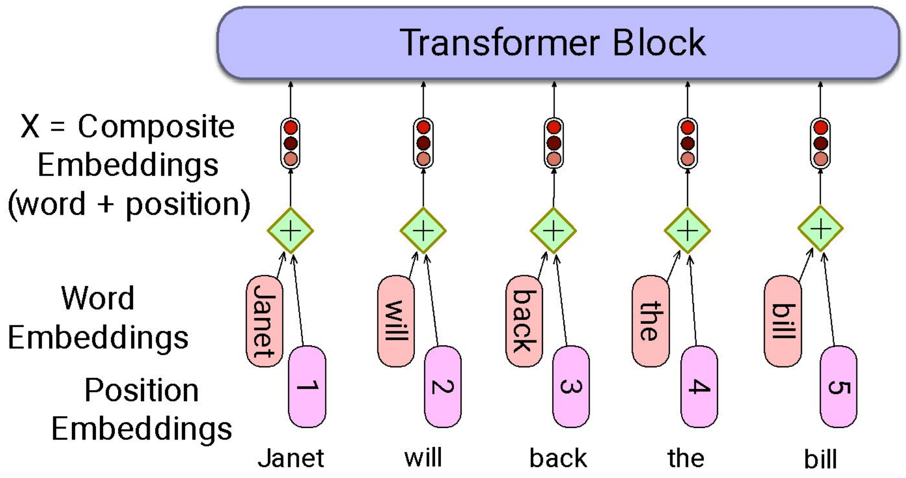
$$b_2 = \sum_i \alpha'_{2,i} v_i$$

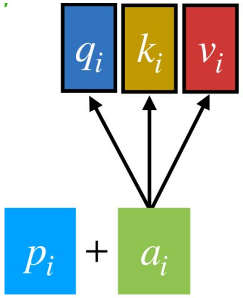


Part 1

Position Encoding

- Most basic method: At the 1st layer, add an embedding of the position to the word embedding (BERT, GPT-3)
- Typically initialized randomly and learned like any other parameter of the model
- Despite adding position, several papers argue that Transformer models are **permutation invariant** / do not model the order of words
- More recent methods we won't cover modify self-attention (RoPE, ALiBi)

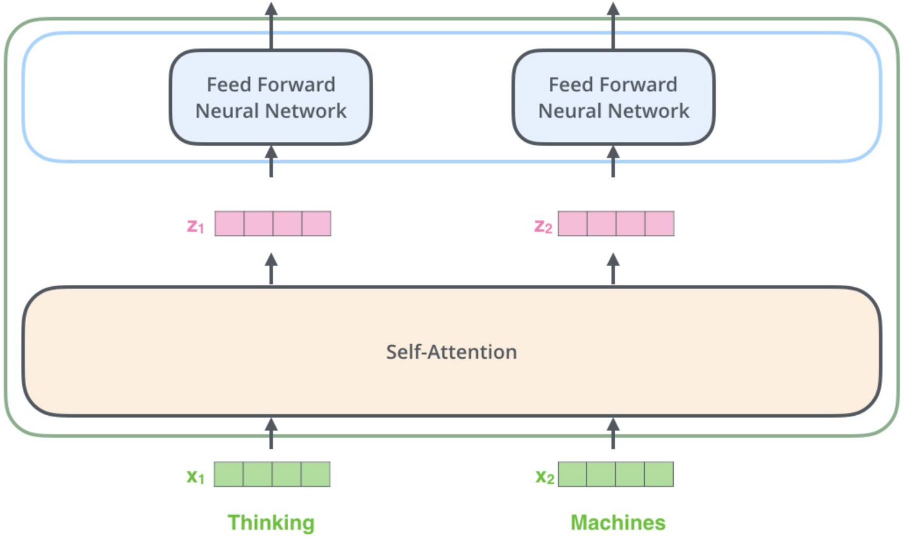




Part 1

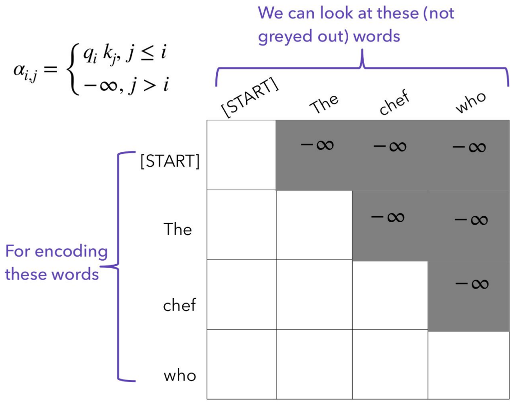
Attention is almost all you need

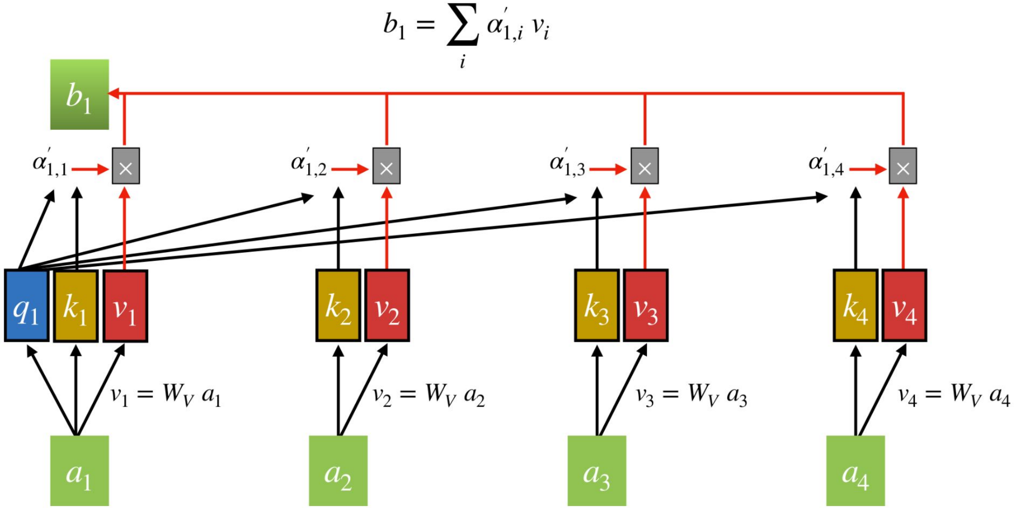
- Since there are no element-wise non-linearities, self-attention is simply performing a re-averaging of the value vectors
- Apply a feedforward layer to the output of attention, providing non-linear activation (and additional expressive power)



Self-attention? What about causality?

- ***b_1*** depends on ***a_2*** and ***a_3***... but the goal is the **generate *a_2*** and ***a_3***
- **Mask** attention scores of future words to preserve **causality**

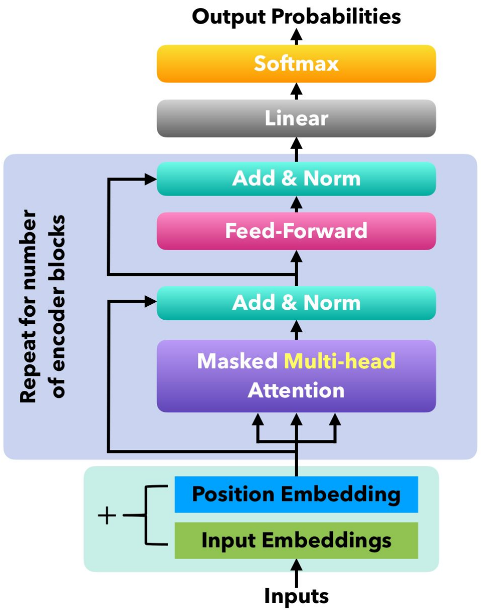




Part 1

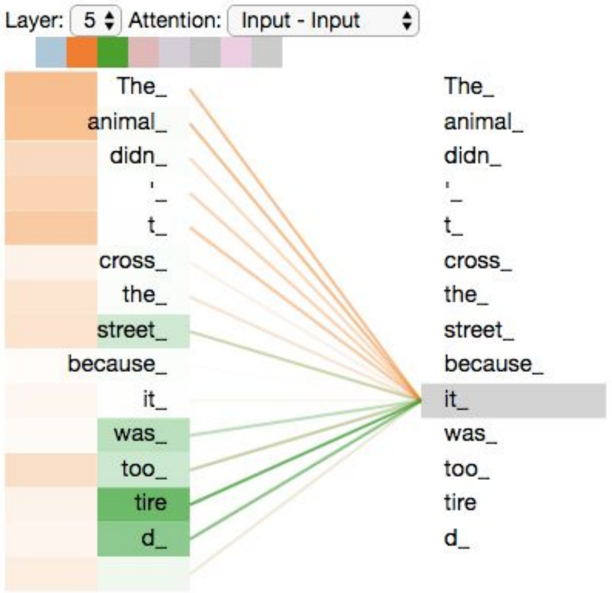
Putting the pieces together

- Positional Encoding: otherwise permutation-invariant
- *(Multi-head)* Self-attention: essential part to model relations between words
  - masked for causality
- *Residual Connection:* for stable training/deeper networks
- Feedforward for nonlinearity/expressiveness
- Linear/softmax: output layer back to vocabulary dimension



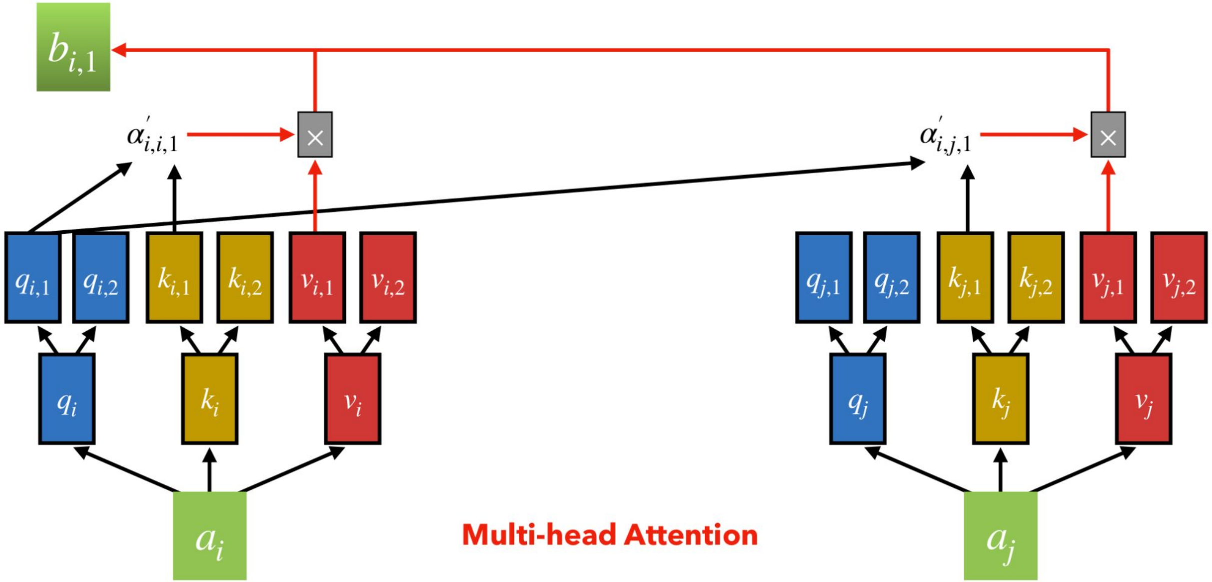
Why Multi-Head Attention?

- What if we want to look in multiple places in the sentence at once?
- For word i, self-attention "looks" where $x\_i\ Q\_i\ K\_j$ is high, but maybe we want to focus on different j for different reasons?
- orange head for the coreference "The animal"
- green head for the object "tired"



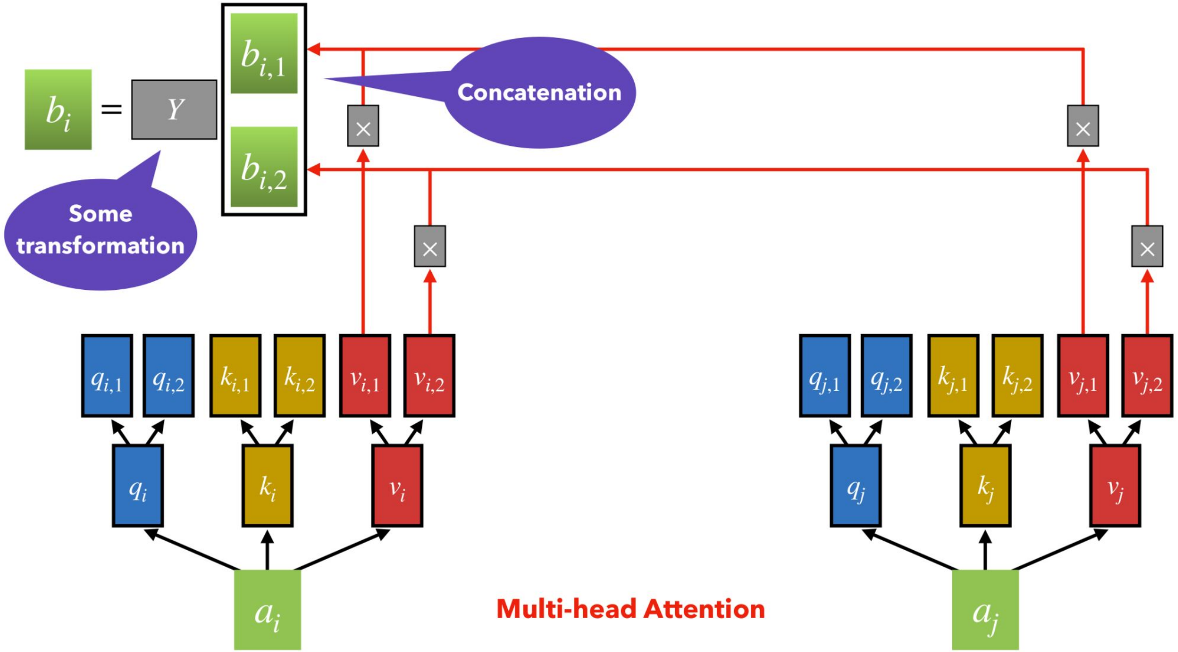
Part 1

Multi-Head Attention: Walk-through



Part 1

Multi-Head Attention: Walk-through



Multi-Head Attention: formally

$$Q^l = I\, W_Q^l$$
$$K^l = I\, W_K^l$$
$$V^l = I\, W_V^l$$

$I = \{a_1, \ldots, a_n\} \in \mathbb{R}^{n \times d}$, where $a_i \in \mathbb{R}^d$

$W_Q^l, W_K^l, W_V^l \in \mathbb{R}^{d \times \frac{d}{h}}$

$Q^l, K^l, V^l \in \mathbb{R}^{n \times \frac{d}{h}}$

Multiple attention "heads" can be defined via multiple **W*** matrices

$$A^l = Q^l\, K^{l^T}$$
$$A^{l'} = \text{softmax}(A^l)$$

$A^{l'}, A^l \in \mathbb{R}^{n \times n}$

$$O^l = A^{l'}\, V^l$$

$O^l \in \mathbb{R}^{n \times \frac{d}{h}}$

Each attention head performs attention independently

$$O = [O^1; \ldots; O^h]\, Y$$

$Y \in \mathbb{R}^{d \times d}$

$[O^1; \ldots; O^h] \in \mathbb{R}^{n \times d}$

$O \in \mathbb{R}^{n \times d}$

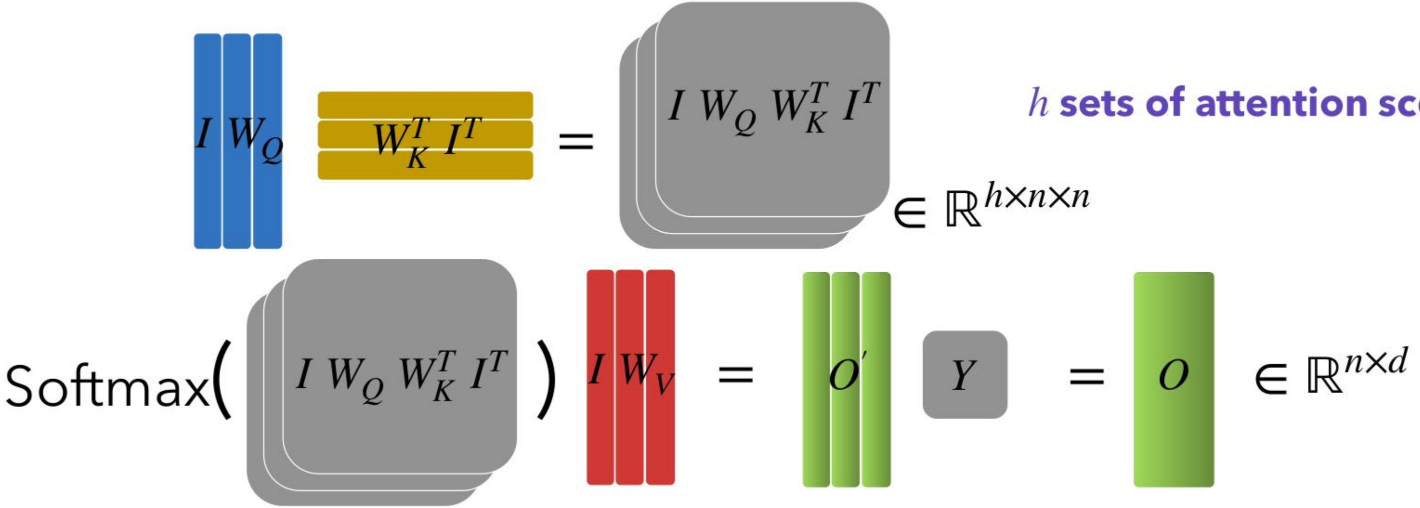
Their results are concatenated

Part 1

Multi-Head Attention: in parallel (as always)

compute $I\ W_Q \in \mathbb{R}^{n\times d}$, and then reshape to $\mathbb{R}^{n\times h\times \frac{d}{h}}$

Then we transpose to $\mathbb{R}^{h\times n\times \frac{d}{h}}$; **now the head axis is like a batch axis**



***h* sets of attention scores!**

Part 1

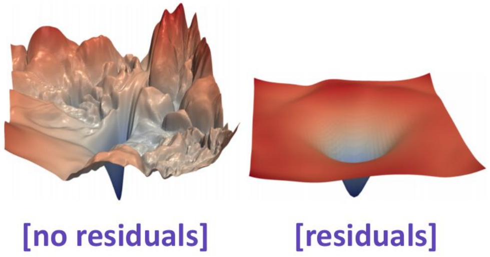
Residual Connections for stable training

- Residual connections are a trick to help models train better.
  - Instead of $X^{(i)} = \text{Layer}(X^{(i-1)})$ (where i represents the layer)

  - We let $X^{(i)} = X^{(i-1)} + \text{Layer}(X^{(i-1)})$ (so we only have to learn "the residual" from the previous layer)

  - Gradient is great through the residual connection; it's 1!
  - Bias towards the identity function!

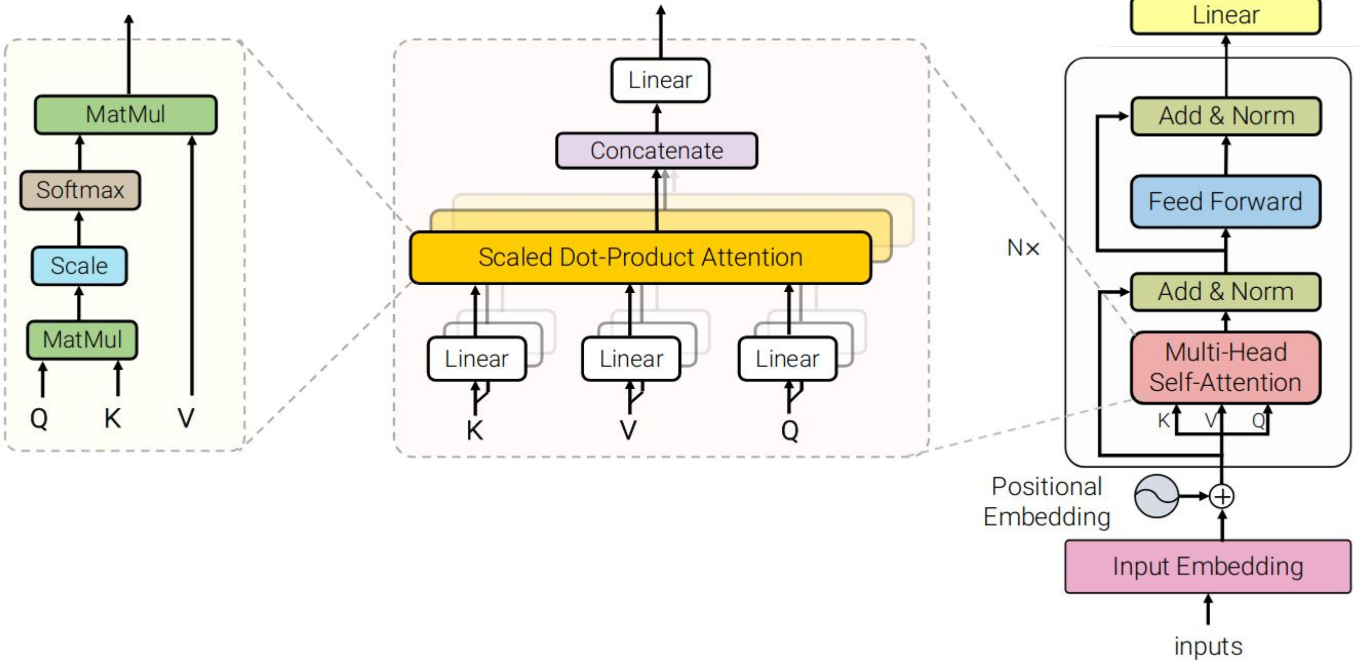
Remember the Cell state in LSTM



[Loss landscape visualization, Li et al., 2018, on a ResNet]

Part 1

Voilà



MatMul
Softmax
Scale
MatMul
Q
K
V
Linear
Concatenate
Scaled Dot-Product Attention
Linear
Linear
Linear
K
V
Q
Output
Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Nx
Add & Norm
Multi-Head
Self-Attention
K
V
Q
Positional
Embedding
Input Embedding
inputs

Transformer for Machine Translation

Source: An admitting privilege is the right of a doctor to admit a patient to a hospital or a medical centre to carry out a diagnosis or a procedure, based on his status as a health care worker at a hospital.

Reference: Le privilège d’admission est le droit d’un médecin, en vertu de son statut de membre soignant d’un hôpital, d’admettre un patient dans un hôpital ou un centre médical afin d’y délivrer un diagnostic ou un traitement.

RNNsearch-50: Un privilège d’admission est le droit d’un médecin d’admettre un patient à un hôpital ou un centre médical pour effectuer un diagnostic ou une procédure, selon son statut de travailleur des soins de santé à l’hôpital.

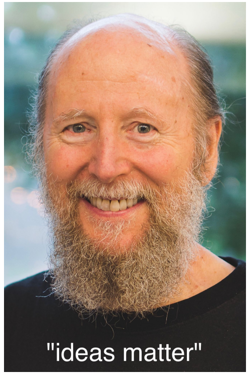
Transformer (fairseq wmt14.en-fr): Un privilège d'admission est le droit d'un médecin d'admettre un patient dans un hôpital ou un centre médical pour y effectuer un diagnostic ou une intervention, en fonction de son statut de travailleur de la santé dans un hôpital.

Vaswani et al. 2017

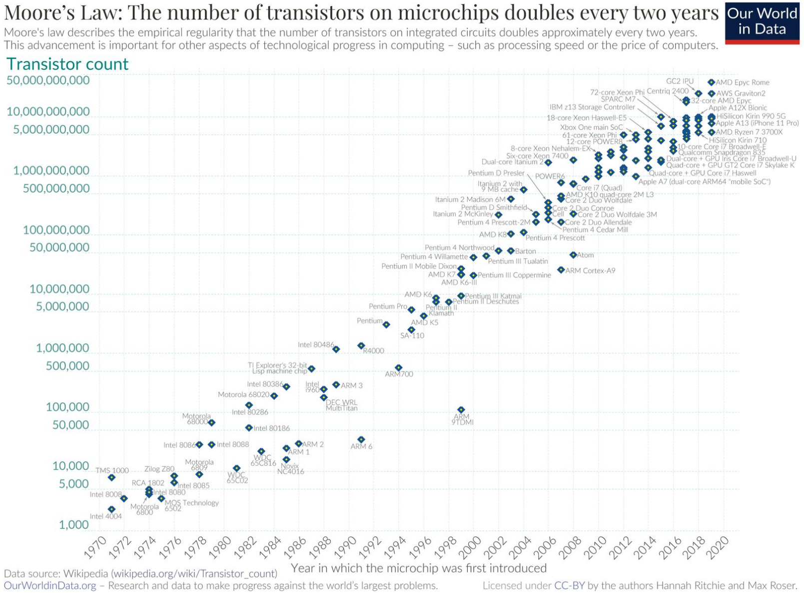
Part 1

Scaling Transformers (the bitter lesson)

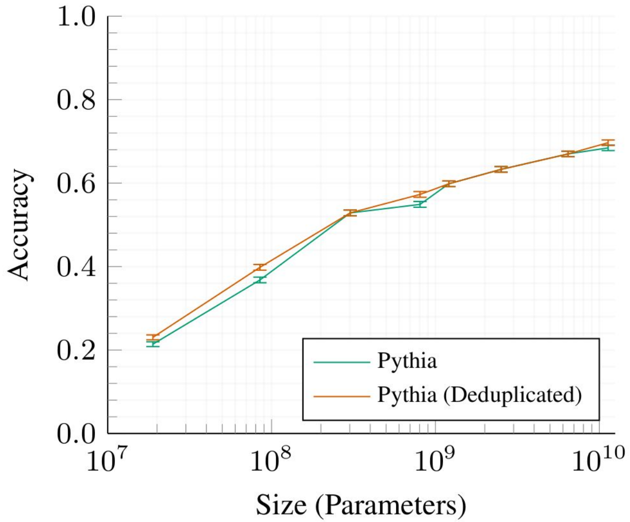
"The biggest lesson that can be read from 70 years of AI research is that **general methods that leverage computation are ultimately the most effective**, and by a large margin"



Rich Sutton

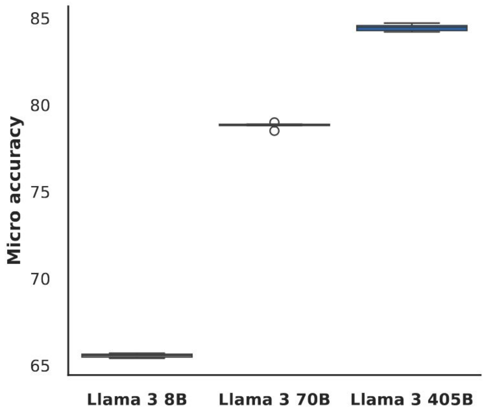


Scaling Transformers (the bitter lesson)

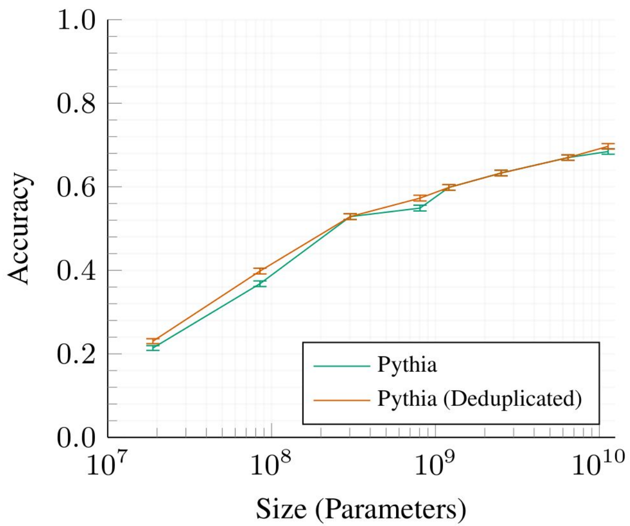


(a) LAMBADA (OpenAI)

There seems to be no limit, from millions to billions to trillions of parameters

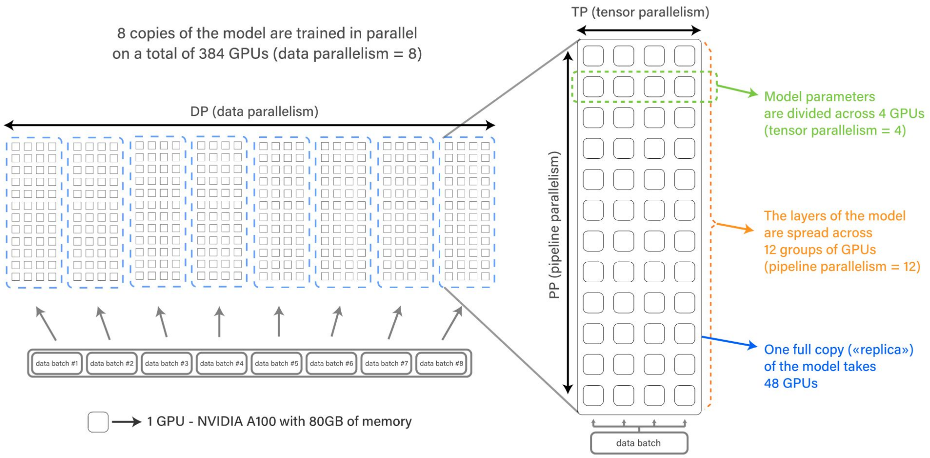


Scaling Transformers: buy more GPUs



(a) LAMBADA (OpenAI)

From 8 × 32GB V100 GPUs (RoBERTa) to 16,384 × 94GB H100 GPUs (Llama 3)



Limit: Teacher Forcing

The	3 %	think	11 %	to	35 %	the	29 %	bathroom	3 %	and	14 %	I	21 %
When	2,5 %	was	5 %	back	8 %	a	9 %	doctor	2%	with	9	It	6
They	2 %	went	2 %	into	5 %	see	5 %	hospital	2 %	,	8 %	The	3 %
…	…	am	1 %	through	4 %	my	3 %	store	1,5 %	to	7 %	There	3 %
I	1 %	will	1 %	out	3 %	bed	2 %	…	…	…	…	…	…
…	…	like	0,5 %	on	2 %	school	1 %	park	0,5 %	.	6 %	STOP	1 %
Banana	0,1 %	…	…	…	…%	…	…	…	…	…	…	…	…

Transformer

START | I | went | to | the | park | . | STOP

Train-test mismatch: exposure bias

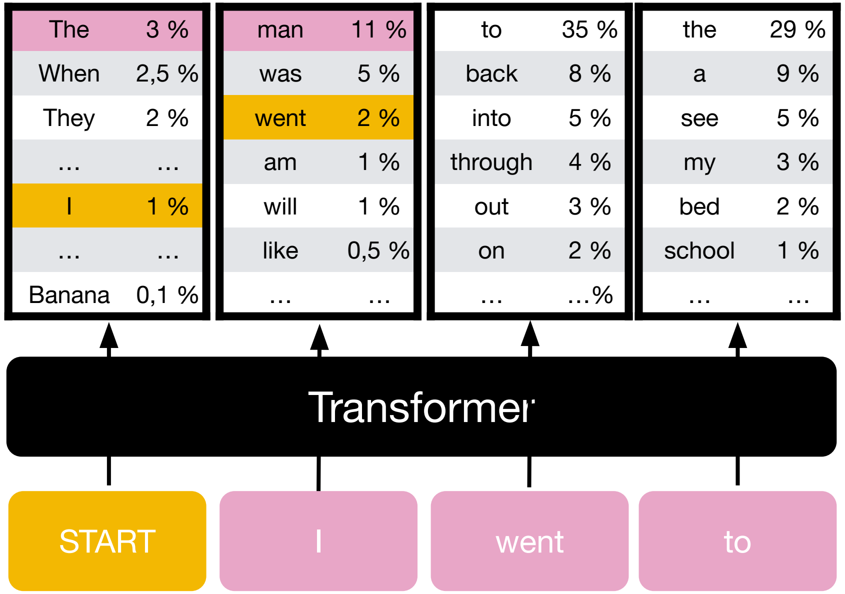
The	3 %	think	11 %	to	35 %	the	29 %	bathroom	3 %	and	14 %	I	21 %
When	2,5 %	was	5 %	back	8 %	a	9 %	doctor	2%	with	9	It	6
They	2 %	went	2 %	into	5 %	see	5 %	hospital	2 %	,	8 %	The	3 %
…	…	am	1 %	through	4 %	my	3 %	store	1,5 %	to	7 %	There	3 %
I	1 %	will	1 %	out	3 %	bed	2 %	…	…	…	…	…	…
…	…	like	0,5 %	on	2 %	school	1 %	park	0,5 %	.	6 %	STOP	1 %
Banana	0,1 %	…	…	…	…%	…	…	…	…	…	…	…	…

Transformer

START	I	went	to	the	park	.	STOP

Part 1

Not easily solved, unlike for RNN

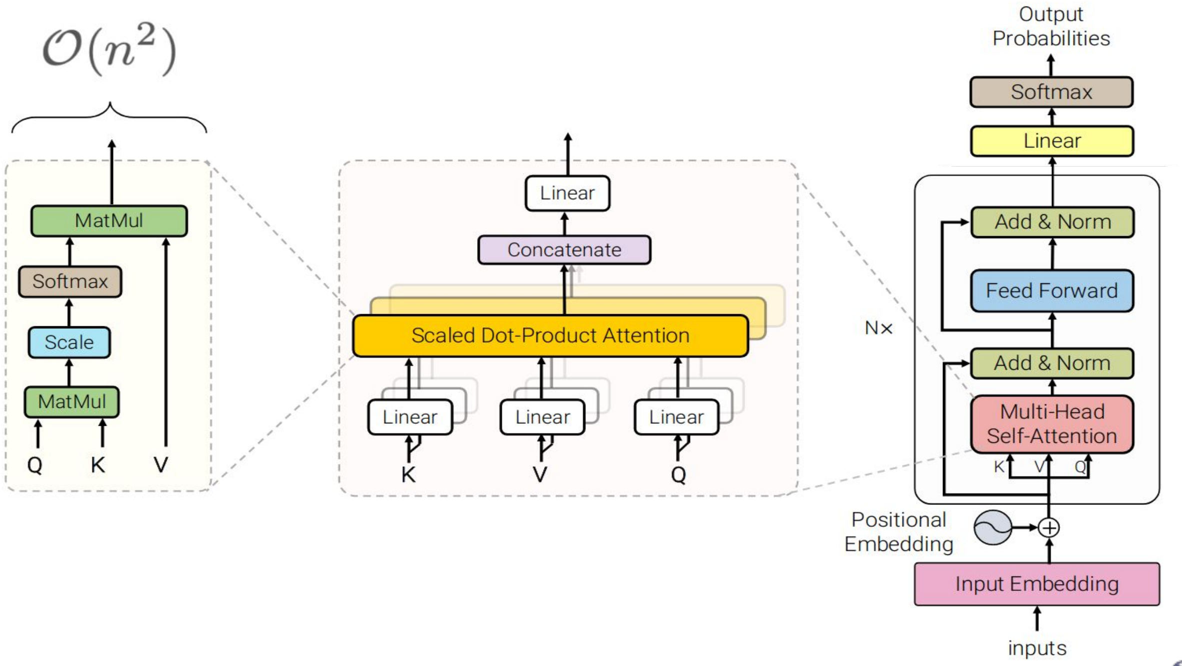


- Because computation is done in parallel, we cannot access the generation of the model
- Teacher forcing is done systematically, model are subject to exposure bias

Mihaylova, T., & Martins, A. F. T. (2019). Scheduled Sampling for Transformers.

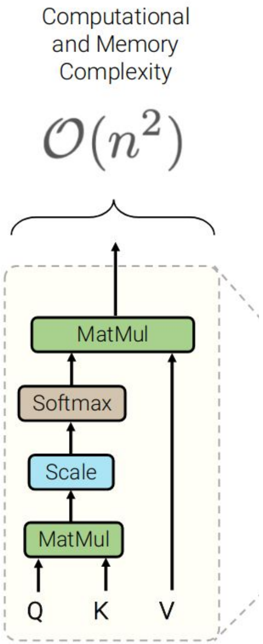
Part 1

Limit: Quadratic complexity



Part 1

Limit: Quadratic complexity

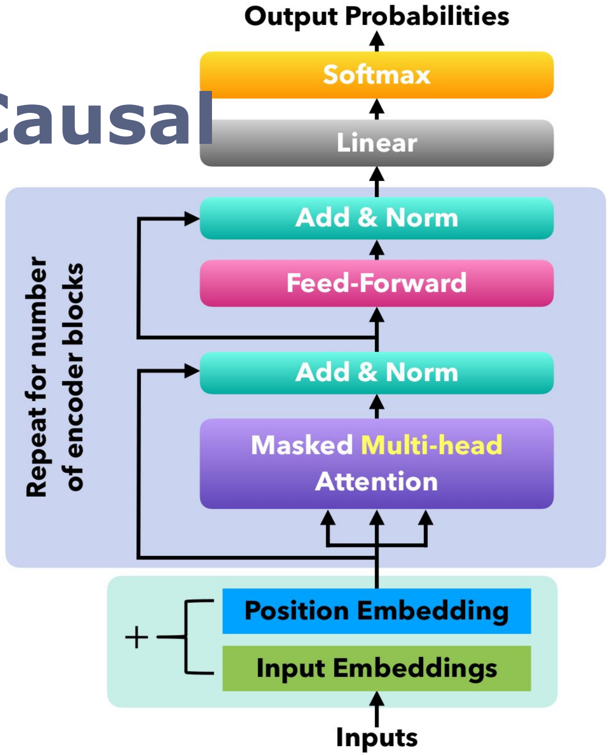


- Computing all pairs of interactions means our computation grows **quadratically** with the sequence length!
- for recurrent models, it only grew **linearly**
- Large body of work on this question (Tay et al., 2020) "Efficient Transformers: A Survey"
- But vanilla Transformer still used in state-of-the-art LLMs

Part 2

Transformer A: (Autoregressive) Decoder/Causal

- Described previously: main architecture for LLMs (**GPT-3**, Llama-*, and many many many more)
- **Causal**/unidirectional mask: can only see past words
- First purpose: **Language Modeling** / autoregressive generation
- But now every task of NLP is cast as Language Modeling, even classification



Part 2

Causal Language Modeling

$p(x\|\text{START})$		$p(x\|\text{START I})$		$p(x\|\cdots\text{went})$		$p(x\|\cdots\text{to})$		$p(x\|\cdots\text{the})$		$p(x\|\cdots\text{park})$		$p(x\|\text{START I went to the park.})$	
The	3 %	think	11 %	to	35 %	the	29 %	bathroom	3 %	and	14 %	I	21 %
When	2,5 %	was	5 %	back	8 %	a	9 %	doctor	2%	with	9	It	6
They	2 %	went	2 %	into	5 %	see	5 %	hospital	2 %	,	8 %	The	3 %
…	…	am	1 %	through	4 %	my	3 %	store	1,5 %	to	7 %	There	3 %
I	1 %	will	1 %	out	3 %	bed	2 %	…	…	…	…	…	…
…	…	like	0,5 %	on	2 %	school	1 %	park	0,5 %	.	6 %	STOP	1 %
Banana	0,1 %	…	…	…	…%	…	…	…	…	…	…	…	…

Transformer Decoder

START | I | went | to | the | park | . | STOP

Part 2

Decoder for: Language Modeling / autoregressive generation

Title: United Methodists Agree to Historic Split
Subtitle: Those who oppose gay marriage will form their own denomination
Article: **After two days of intense debate, the United Methodist Church has agreed to a historic split - one that is expected to end in the creation of a new denomination, one that will be "theologically and socially conservative," according to The Washington Post. The majority of delegates attending the church's annual General Conference in May voted to strengthen a ban on the ordination of LGBTQ clergy and to write new rules that will "discipline" clergy who officiate at same-sex weddings. But those who opposed these measures have a new plan: They say they will form a separate denomination by 2020, calling their church the Christian Methodist denomination.**

GPT-3 (2020)

Part 2

Decoder for: Actually everything (but we'll come back to that)

- Classification: "I like this movie"

  → `"I like this movie,` **`it was`** `{good/bad}"`

- Question Answering: "When was Dante born?"

  → **`"Dante was born in ____"`**

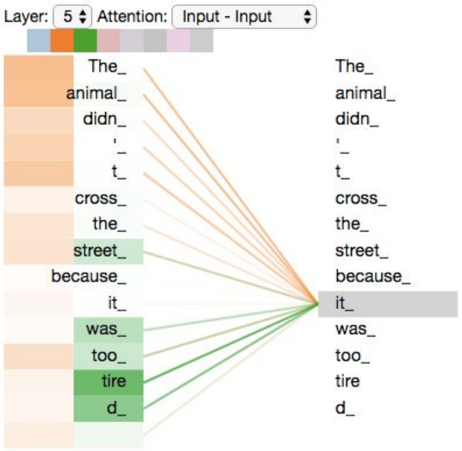
- Translation: "I like pasta"

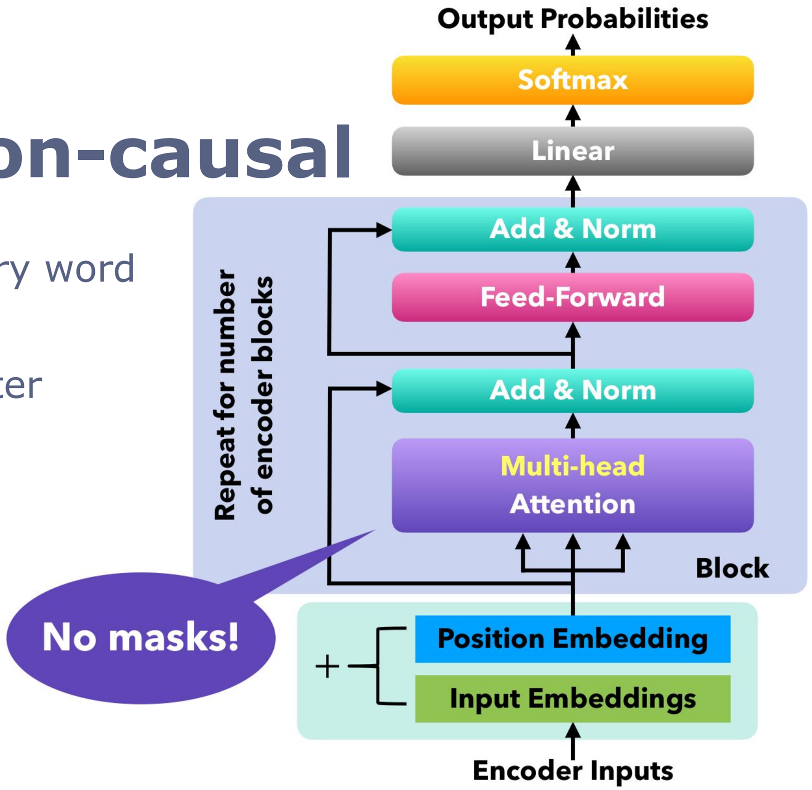
  → **`"The translation of`** `'I like pasta'` **`in French is ____"`**

Part 2

Transformer B: (Bidirectional) Encoder/non-causal

- Removes the mask from self-attention: now every word can see future and past
- Use for classification (but now words have a better context, unlike bag of words)
- Famous examples: **BERT**, m**BERT**, Ro**BERT**a, De**BERT**a, Camem**BERT**, ...





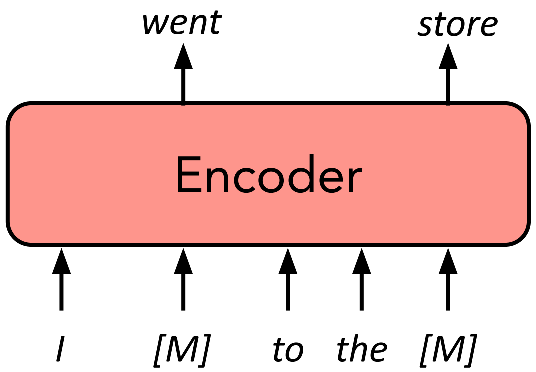
Paul Lerner – *November 2025*

aivancity PARIS-CACHAN

Part 2

Masked Language Modeling

- How to encode information from both **bidirectional** contexts?
- General Idea: **text reconstruction!**



$$h_1, \ldots, h_T = \text{Encoder}(w_1, \ldots, w_T)$$
$$y_i \sim Aw_i + b$$

Only add loss terms from the masked tokens. If $\tilde{x}$ is the masked version of x, we're learning $p_\theta(x \mid \tilde{x})$. Called **Masked Language model (MLM)**.

Masked Language Modeling

hyperparameter

- Choose a random **15%** of tokens to predict.

hyperparameter

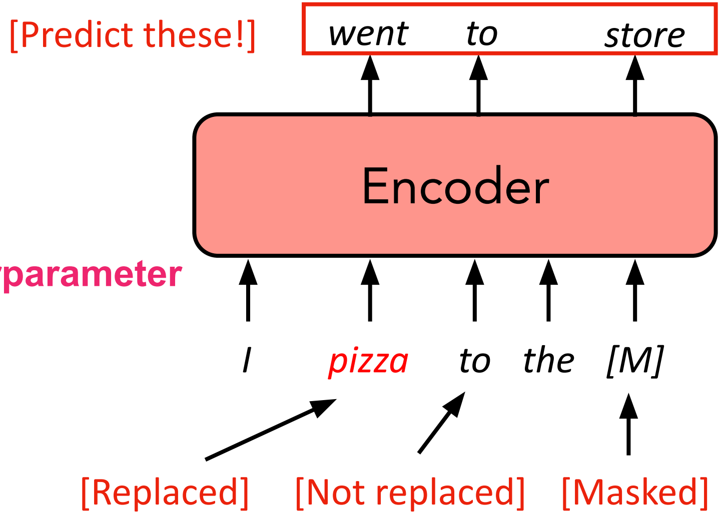
- For each chosen token:
  - Replace it with [MASK] 80% of the time.

hyperparameter

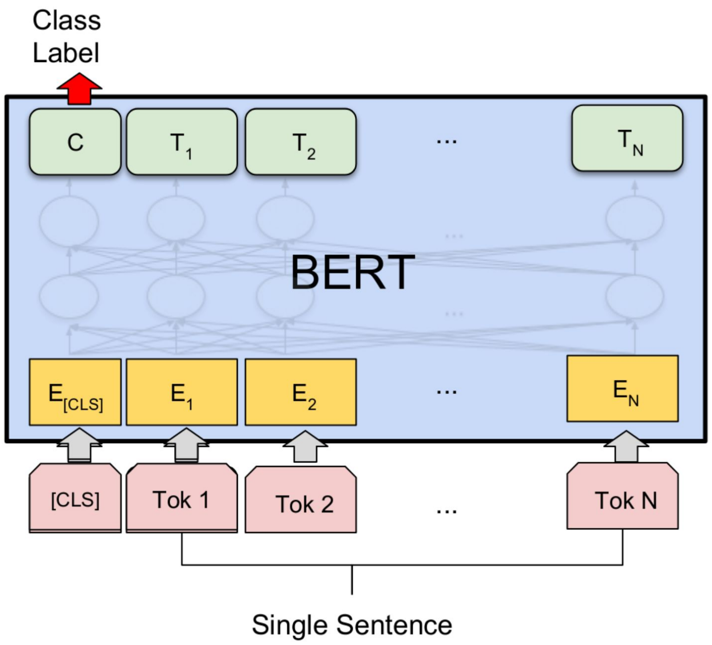
  - Replace it with a random token 10% of the time.

hyperparameter

  - Leave it unchanged 10% of the time (but still predict it!)
- Only learns from **15%** of tokens per step



Fine-tuning Encoder for: Sentiment Analysis



I just loved every minute of this film.

A strangely compelling and brilliantly acted psychological drama.

Preaches to two completely different choirs at the same time, which is a pretty amazing accomplishment.

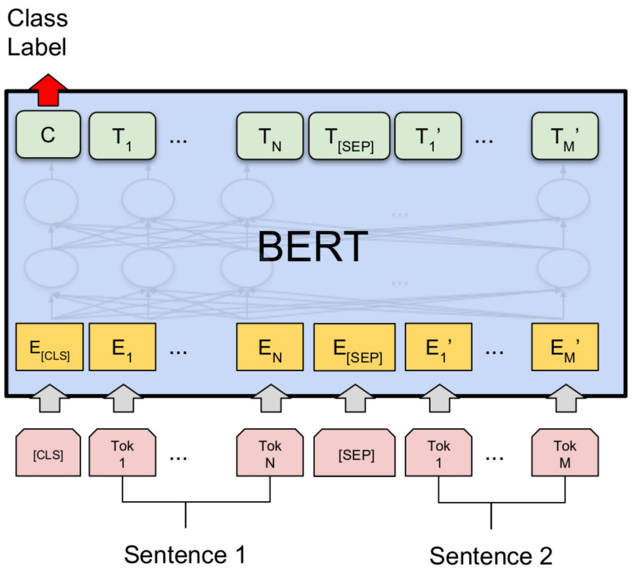
An instant candidate for the worst movie of the year.

The film seems a dead weight.

I found it slow, drab, and melodramatic.

Part 2

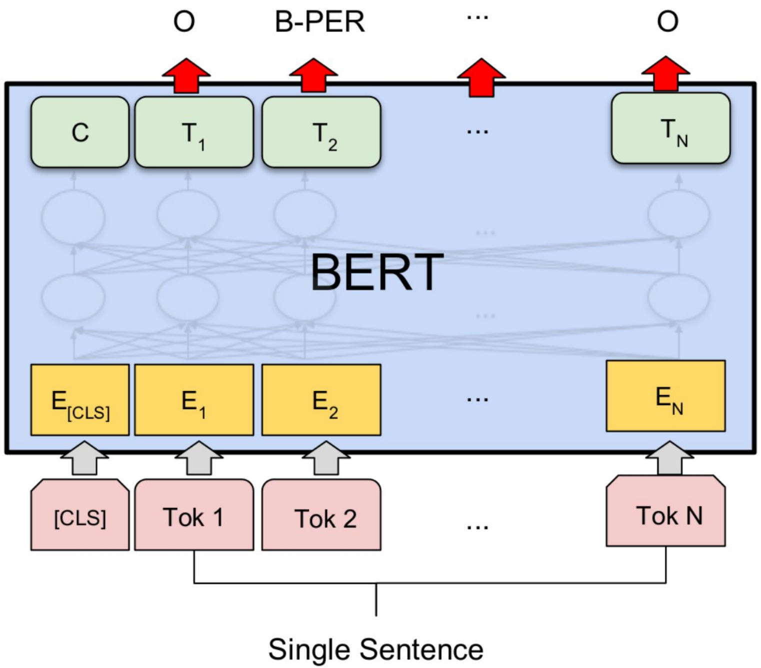
Fine-tuning Encoder for: Natural Language Inference



Met my first girlfriend that way.	✗	I didn't meet my first girlfriend until later.
At 8:34, the Boston Center controller received a third transmission from American 11	✓	The Boston Center controller got a third transmission from American 11.
someone else noticed it and i said well i guess that's true and it was somewhat melodious in other words it wasn't just you know it was really funny	✗	No one noticed and it wasn't funny at all.

Part 2

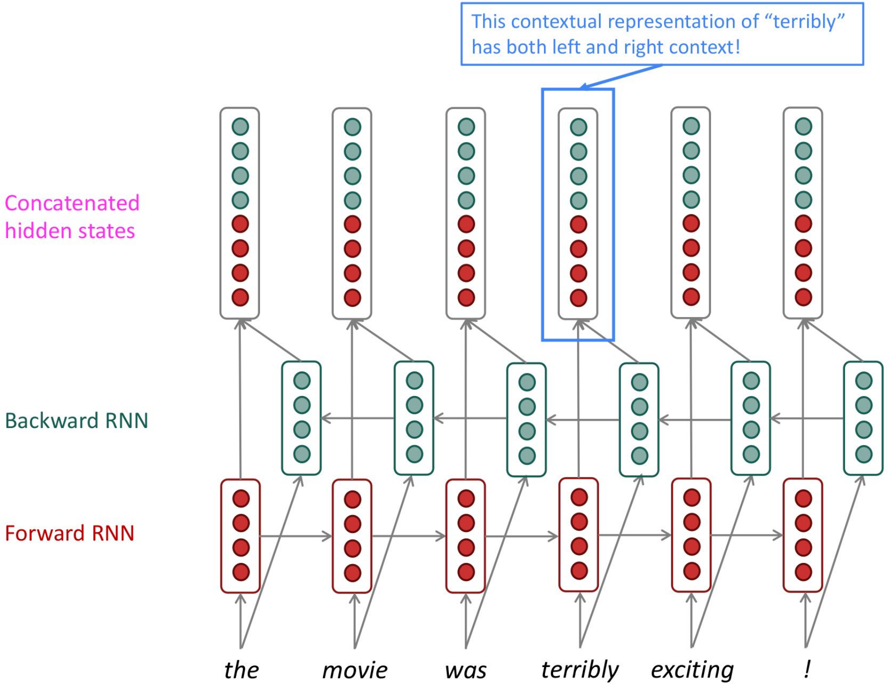
Fine-tuning Encoder for: Named Entity Recognition



Washington is the capital of the USA. It hosts the White House.

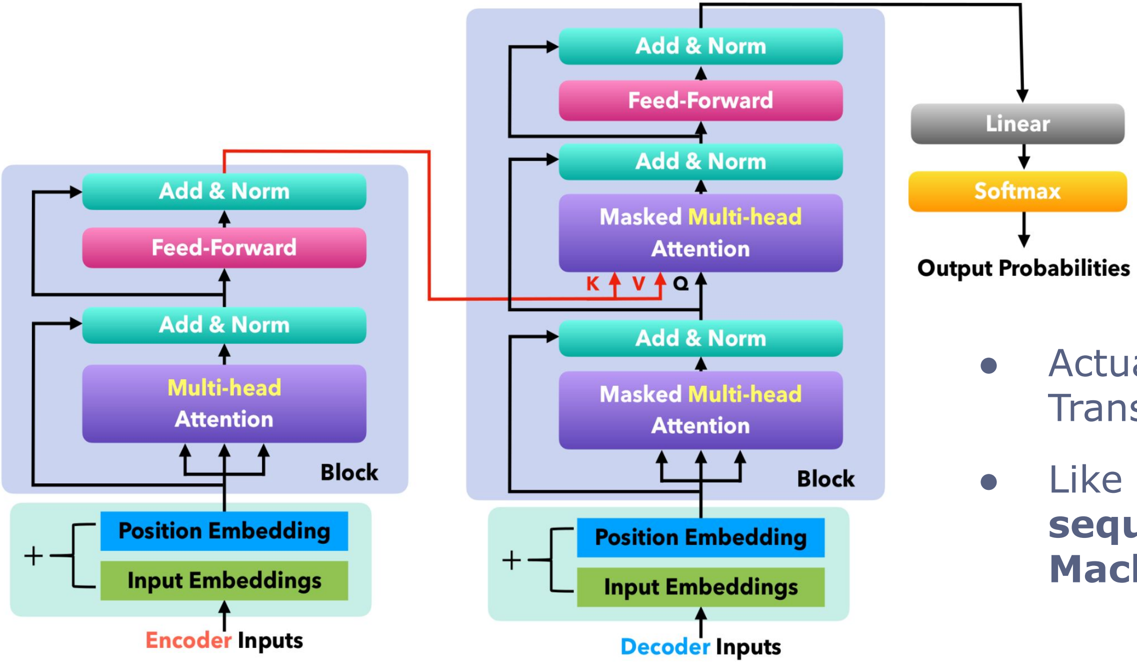
Note on Bidirectional RNNs

- RNNs could also be bidirectional but you then needed two!
- → long sequential (unparallelizable) operations, although still ***O(n)*** theoretically



Part 2

Transformer C: Encoder-Decoder

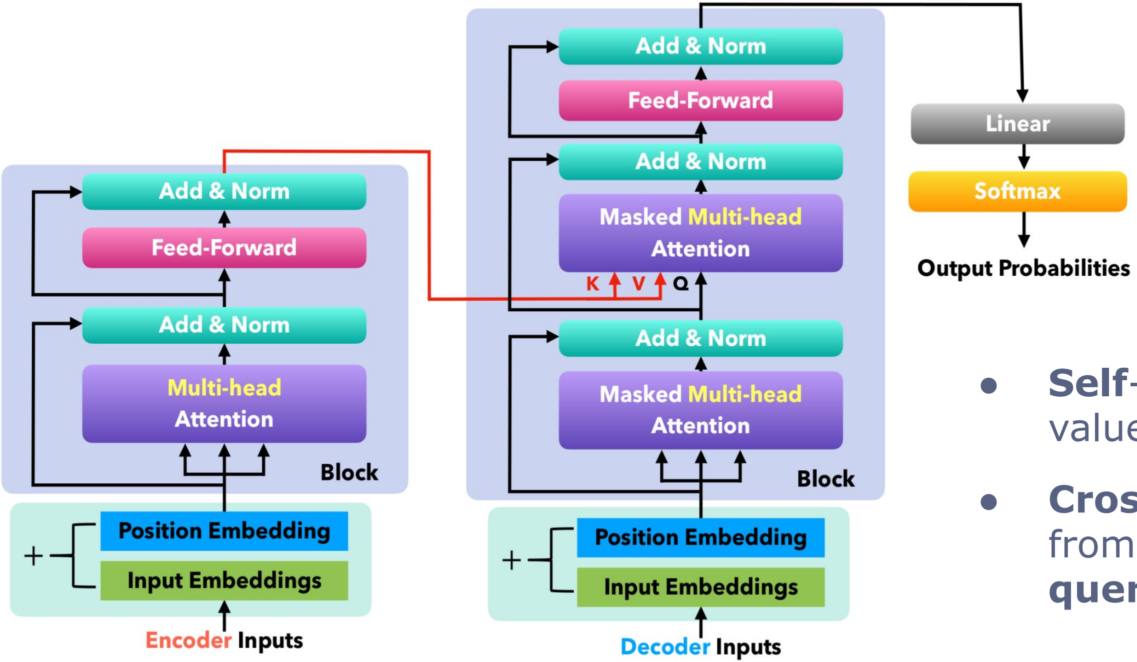


- Famous examples: T5, BART, BARThez, …

- Actually the first variant proposed for Translation by Vaswani et al. (2017)
- Like an RNN Encoder-Decoder, use for **sequence-to-sequence** tasks like **Machine Translation**

Part 2

Cross-Attention in Encoder-Decoder

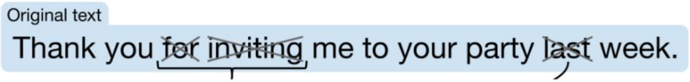


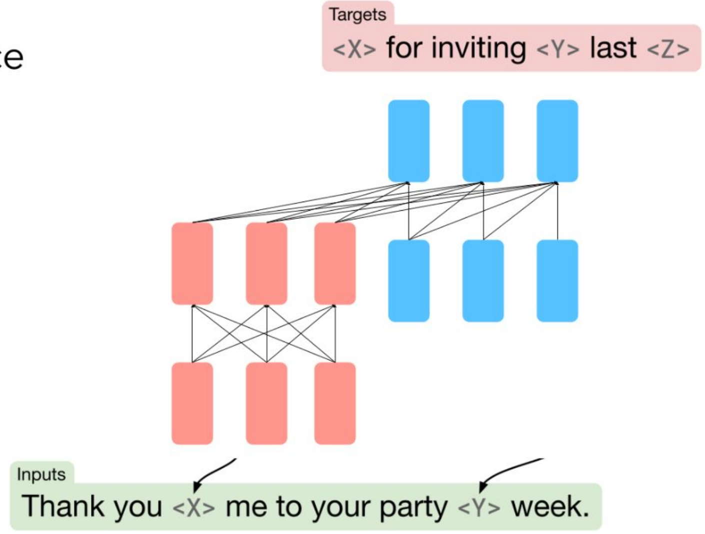
- **Self**-attention: queries, keys, and values come **from the same source**
- **Cross**-Attention: *keys* and *values* are from *Encoder* (like a memory); **queries** are from **Decoder**

Part 2

Text Denoising

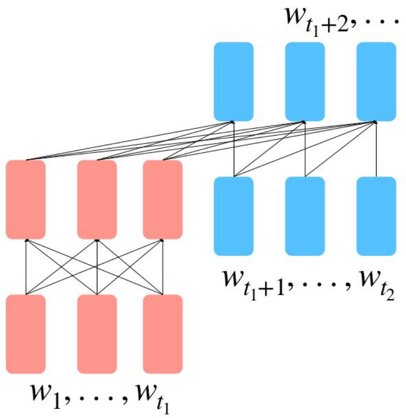
- **Text span corruption (denoising):** Replace different-length spans from the input with unique placeholders (e.g., <extra_id_0>); decode out the masked spans.
    - Done during **text preprocessing**: training uses **language modeling** objective at the decoder side





Encoder-Decoder Training

- Encoder builds a representation of the source and gives it to the decoder
- Decoder uses the source representation to generate the target sentence
- The **encoder** portion benefits from **bidirectional** context; the decoder portion is used to train the whole model through **language modeling**



$$h_1, \ldots, h_{t_1} = \text{Encoder}(w_1, \ldots, w_{t_1})$$
$$h_{t_1+1}, \ldots, h_{t_2} = \text{Decoder}(w_{t_1+1}, \ldots, w_{t_2}, h_1, \ldots, h_{t_1})$$
$$y_i \sim Ah_i + b, i > t$$

Part 2

Encoder-Decoder for: Translation

translate English to French: This image section from an infrared recording by the Spitzer telescope shows a "family portrait" of countless generations of stars: the oldest stars are seen as blue dots, while more difficult to identify are the pink-coloured "new-borns" in the star delivery room.

→

Ce détail d’une photographie infrarouge prise par le télescope Spitzer montre un "portrait de famille" des innombrables générations d’étoiles: les plus vieilles étoiles sont en bleu et les points roses, plus difficiles à identifier, sont les "nouveau-nés" dans la salle d’accouchement de l’univers.

T5 (2020)

Part 2

Encoder-Decoder for: Summarization

summarize: marouane fellaini and adnan januzaj continue to show the world they are not just teammates but also best mates. the manchester united and belgium duo both posted pictures of themselves out at a restaurant on monday night ahead of their game against newcastle on wednesday . januzaj poses in the middle of fellaini and a friend looking like somebody who failed to receive the memo about it being a jackson 5 themed night. [...]

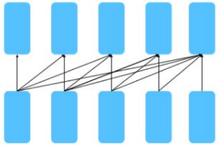
→

the belgian duo took to the dance floor on monday night with some friends . manchester united face newcastle in the premier league on wednesday . [...]

T5 (2020)

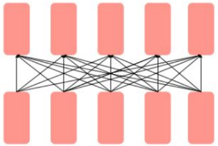
3 main objectives for 3 architectures

Decoder



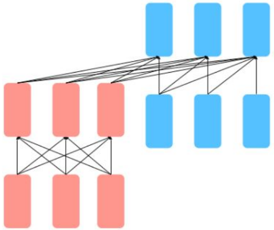
- Language modeling; can only condition on the past context

Encoder



- Bidirectional; can condition on the future context

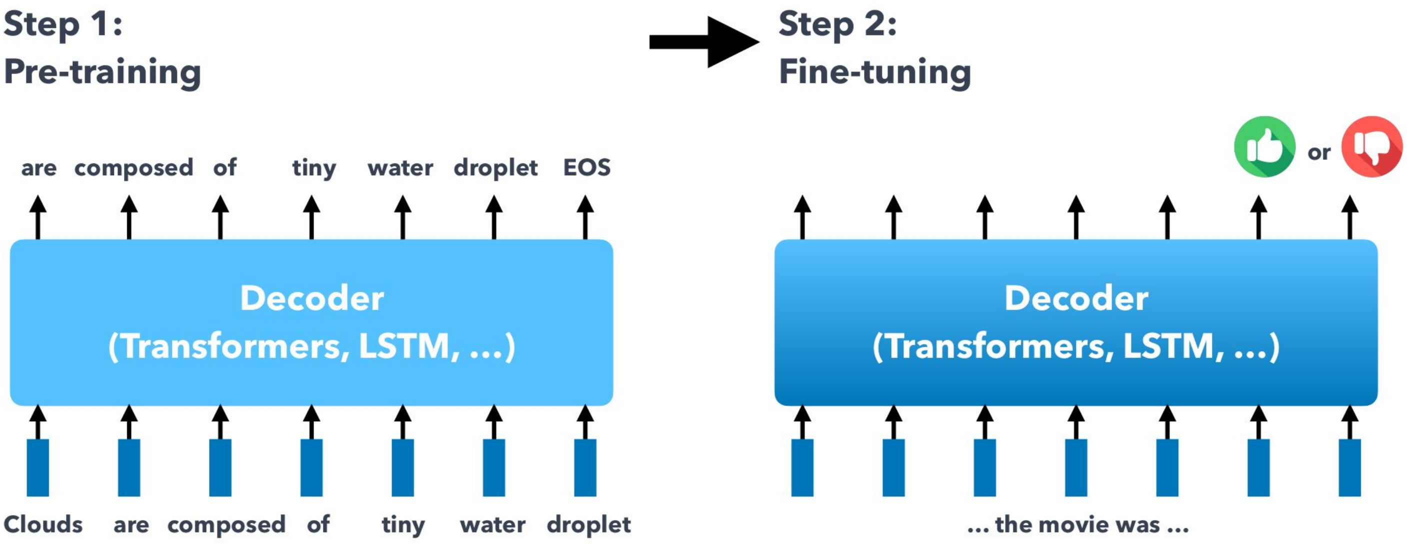
Encoder-Decoder



- Map two sequences of different length together

Part 3

Fine-Tuning

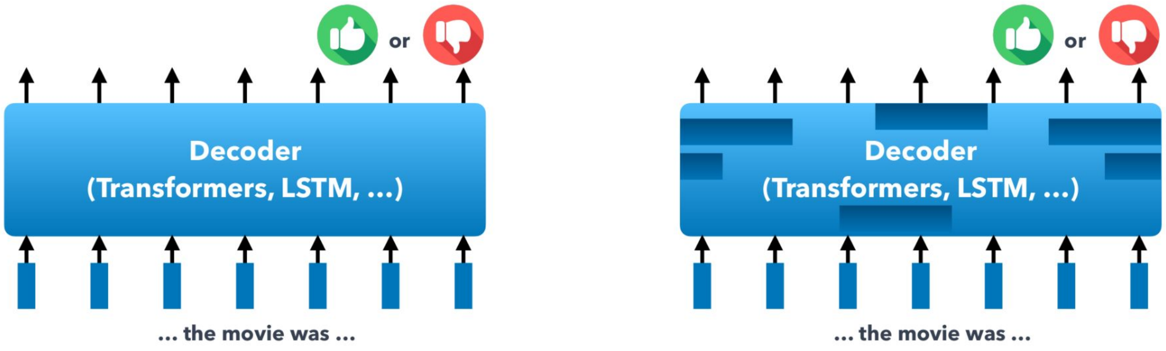


Abundant data; learn general language

Limited data; adapt to the task

Parameter-Efficient Fine-Tuning (PEFT)

Instead of updating all parameters in the massive neural network (up to many billions of parameters), **can we make fine-tuning more efficient?**



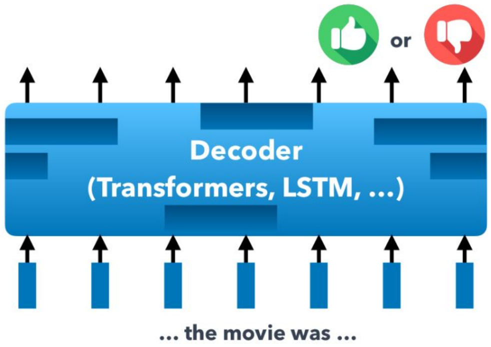
Full Fine-tuning

Updating all parameters

Parameter-Efficient Fine-tuning

Updating a few existing or new parameters

Parameter-Efficient Fine-Tuning (PEFT)



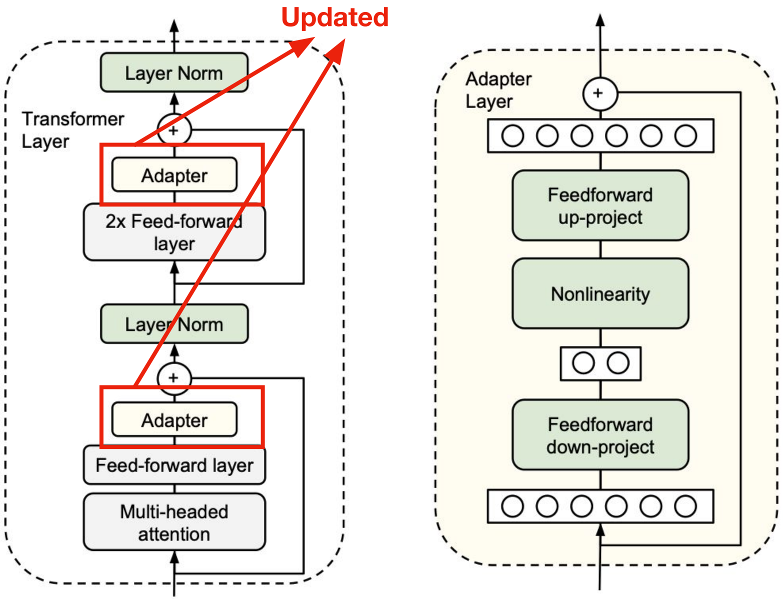
Parameter-Efficient Fine-tuning

Updating a few existing or new parameters

- **More efficient at fine-tuning & inference time**
- **Less overfitting** by keeping the majority of parameters learned during pre-training

Part 3

PEFT v1: Adapters

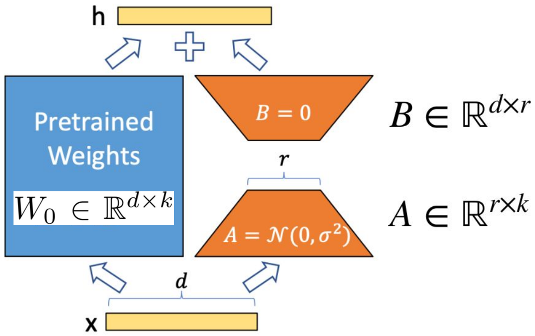


- Injecting **new layers** (randomly initialized) into the original network, keeping **other parameters frozen**
- only learn the **Residual**

Rebuffi et al. (2017); Houlsby et al. (2019)

Part 3

PEFT v2: Low-Rank Adaptation (LoRA)



where rank $r \ll min(d, k)$

Frozen **Updated**

$$W_0 + \Delta W = \boxed{W_0} + \boxed{BA}$$

- Like Adapter but "low-rank" (***r***) and combined with pretrained weights
- After training, weights are combined → same inference speed as pretrained model

$$h = W_0 x + BAx$$

$$h = (W_0 + BA)x$$

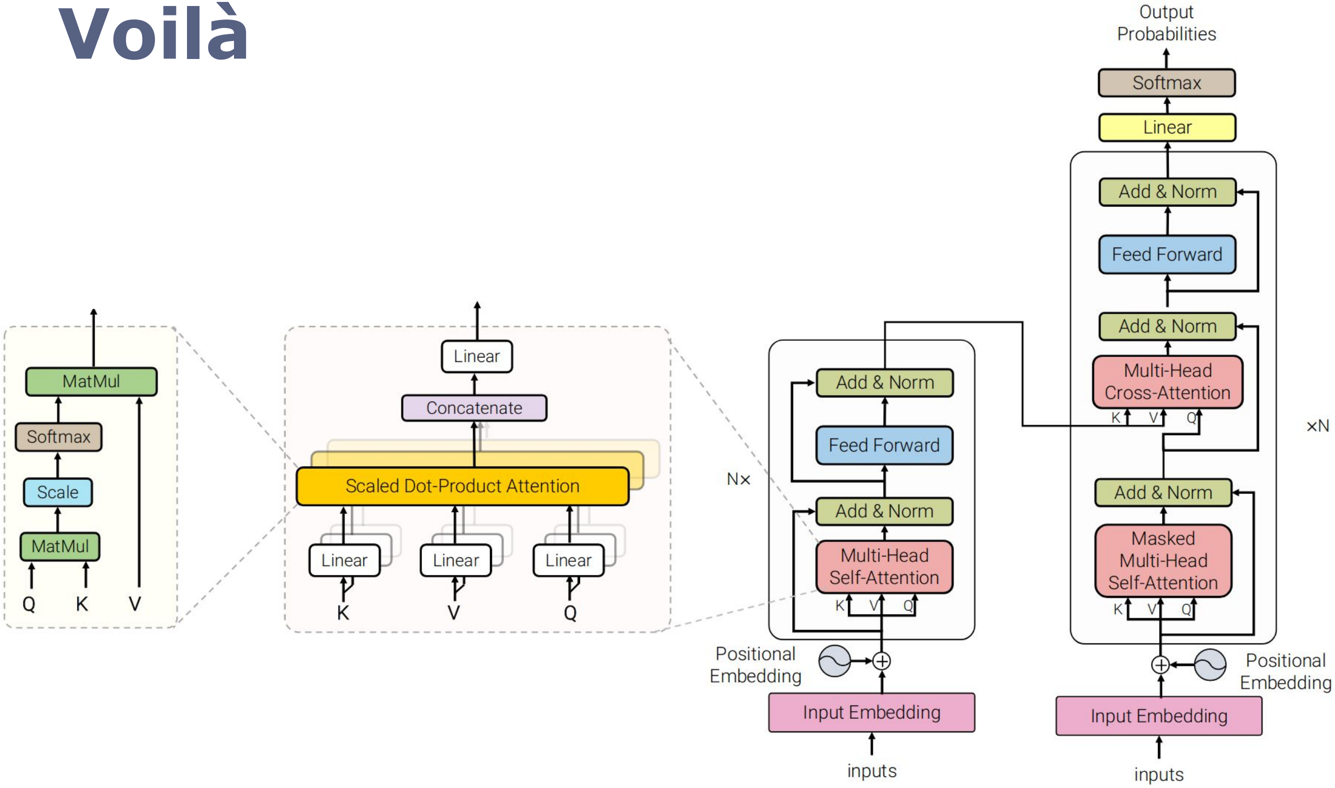
Part 3

Summarizing

- Transformer leverages attention in a parallelized way:
  scales as long as you buy enough GPUs
- Language Modeling is a powerful self-supervised pretraining method, which scales well (did not find limit yet)
- Every NLP task can be framed as Language Modeling but:
  - (Bidirectional) Encoders are better suited for classification
  - Encoder-Decoders are better suited for sequence-to-sequence (Translation)
- We do not need to fine-tune the entire model (LoRA/PEFT)

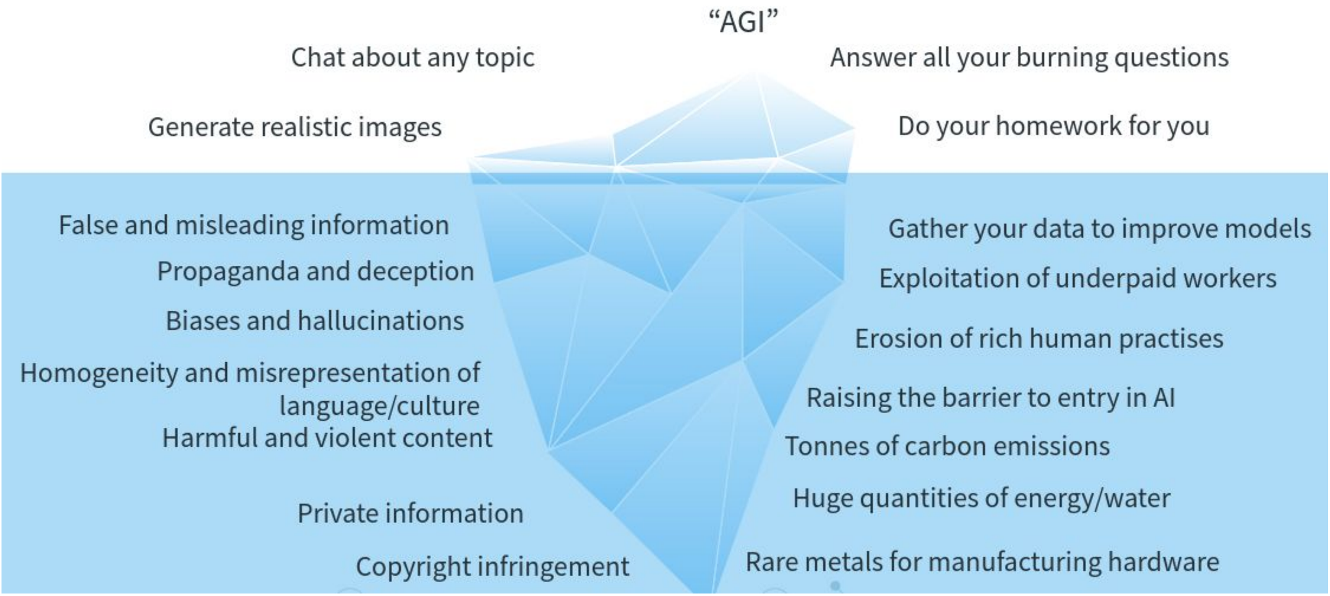
Part 3

Voilà


MatMul
Softmax
Scale
MatMul
Q
K
V
Linear
Concatenate
Scaled Dot-Product Attention
Linear
Linear
Linear
K
V
Q
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head
Self-Attention
K
V
Q
Positional
Embedding
Input Embedding
inputs
Output
Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head
Cross-Attention
K
V
Q
×N
Add & Norm
Masked
Multi-Head
Self-Attention
K
V
Q
Positional
Embedding
Input Embedding
inputs

Part 3

Next class: Ethical, social, and environmental issues + perspectives



Part 0

Acknowledgements

This class directly builds upon:

- **Jurafsky, D., & Martin, J. H.** (2024). Speech and Language Processing : An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models (3rd éd.).
- **Eisenstein, J.** (2019). Natural Language Processing. 587.
- **Yejin Choi**. (Winter 2024). CSE 447/517: Natural Language Processing (University of Washington - Paul G. Allen School of Computer Science & Engineering)
- **Noah Smith**. (Winter 2023). CSE 447/517: Natural Language Processing (University of Washington - Paul G. Allen School of Computer Science & Engineering)
- **Benoît Sagot**. (2023-2024). *Apprendre les langues aux machines* (Collège de France)
- **Chris Manning**. (Spring 2024). Stanford CS224N: Natural Language Processing with Deep Learning
- Classes where I was/am Teacher Assistant:
  - **Christopher Kermorvant**. Machine Learning for Natural Language Processing (ENSAE)
  - **François Landes** and **Kim Gerdes**. Introduction to Machine Learning and NLP (Paris-Saclay)

Also inspired by:

- My PhD thesis: *Répondre aux questions visuelles à propos d'entités nommées* (2023)
- **Noah Smith** (2023): Introduction to Sequence Models (LxMLS)
- **Kyunghyun Cho**: Transformers and Large Pretrained Models (LxMLS 2023), Neural Machine Translation (ALPS 2021)
- My former PhD advisors **Olivier Ferret** and **Camille Guinaudeau** and postdoc advisor **François Yvon**
- My former colleagues at LISN

aivancity

PARIS-CACHAN

advancing education in artificial intelligence

Part 1

Perplexity

- How "hard" is the task of recognizing digits '0,1,2,..,9' uniformly at random?

$d \sim \mathrm{Uniform}(0, 9)$

$$\mathrm{PP}(d_1, \dots, d_N) = p(d_1, \dots, d_N)^{-\frac{1}{N}} = \left(\frac{1}{10}^N\right)^{-\frac{1}{N}} = \frac{1}{10}^{-1} = 10$$

- Perplexity: 10
- Using entropy (replacing the estimated distribution with the known true dist.):

$$H(D, D) = -\sum_{d \in D} p(d) \log p(d) = -\sum_{i=0}^{9} p(i) \log p(i) = -\sum_{i=0}^{9} \frac{1}{10} \log\left(\frac{1}{10}\right) = -\log\left(\frac{1}{10}\right)$$

$$PP(D) = e^{H(D,D)} = e^{-\log\left(\frac{1}{10}\right)} = (e^{\log\left(\frac{1}{10}\right)})^{-1} = \left(\frac{1}{10}\right)^{-1} = 10$$

<u>Same result!</u>

Part 1

LSTM with Attention: formally

We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$

On timestep *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$

We get the attention scores $\boldsymbol{e}^t$ for this step:

$$\boldsymbol{e}^t = [\boldsymbol{s}_t^T \boldsymbol{h}_1, \ldots, \boldsymbol{s}_t^T \boldsymbol{h}_N] \in \mathbb{R}^N$$

We take softmax to get the attention distribution α^t for this step (this is a probability distribution and sums to 1)

$$\alpha^t = \mathrm{softmax}(\boldsymbol{e}^t) \in \mathbb{R}^N$$

We use α^t to take a weighted sum of the encoder hidden states to get the attention output $\boldsymbol{a}_t$

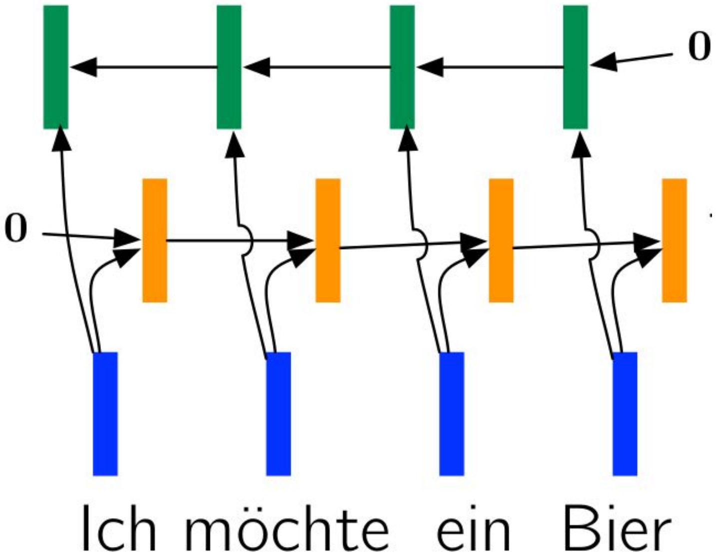
$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

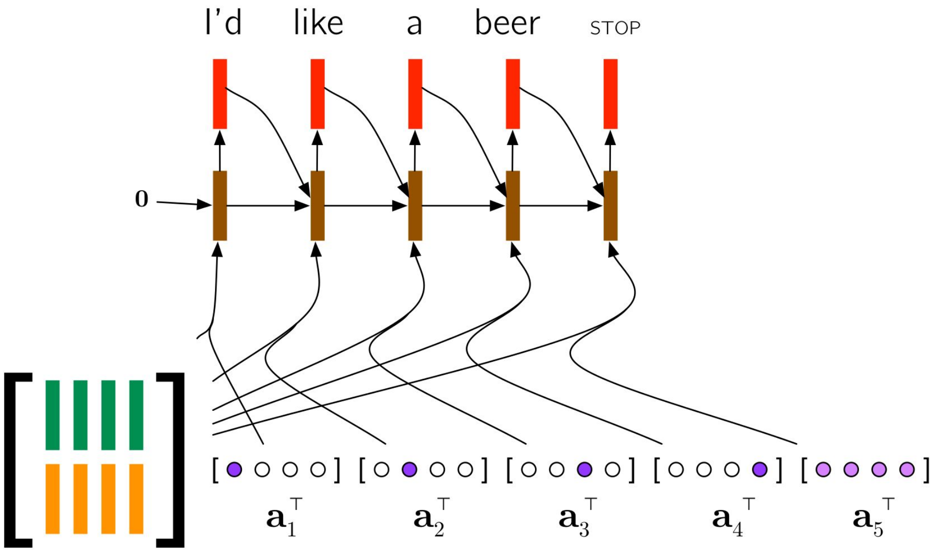
Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state s_t and proceed as in the non-attention seq2seq model

$$[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$

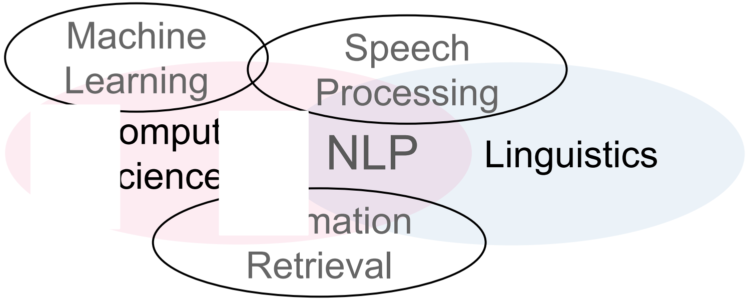
Part 1

Sequence-to-Sequence (Translation)





Why Sequence Models? and Attention?



- Close to Speech Processing (Automatic Speech Recognition etc.)
- Close to Information Retrieval (Search engines like Google)
- Driven by Statistical/Machine Learning methods since the 90s (Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., & Mercer, R. L. (1993). The Mathematics of Statistical Machine Translation : Parameter Estimation. Computational Linguistics, 19(2), 263-311.)
- Driven by Deep Learning since 2013 (Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. Advances in Neural Information Processing Systems)

Part 2

Masked Language Modeling: Transfer