Deep Learning in Life Sciences

Attention mechanisms and transformers

D. Malchiodi, 18/03/2024

CHARLES UNIVERSITY

SORBONNE UNIVERSITÉ

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

UNIVERSITY OF WARSAW

UNIVERSITÀ DEGLI STUDI DI MILANO

4eu+ EUROPEAN UNIVERSITY ALLIANCE

Who am I?

dario.malchiodi@unimi.it
https://malchiodi.di.unimi.it

TEACHING

Associate professor @unimi (statistics & data analysis, algorithms for massive datasets)

RESEARCH

Data-driven induction of non-classical sets, compression of ML models, negative example selection, application of ML to medicine, veterinary, forensics & cultural heritage. Visiting scientist @uca @inria

POPULARIZATION OF COMPUTING

Italian National Science and Technology museum, RadioPopolare, ALaDDIn

The starting point

arXiv > cs > arXiv:1706.03762

Search...

Help | Advanc

Computer Science > Computation and Language

[Submitted on 12 Jun 2017 (v1), last revised 2 Aug 2023 (this version, v7)]

Attention Is All You Need

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Source: Vaswani et al., 2017 [1]

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

Abstract

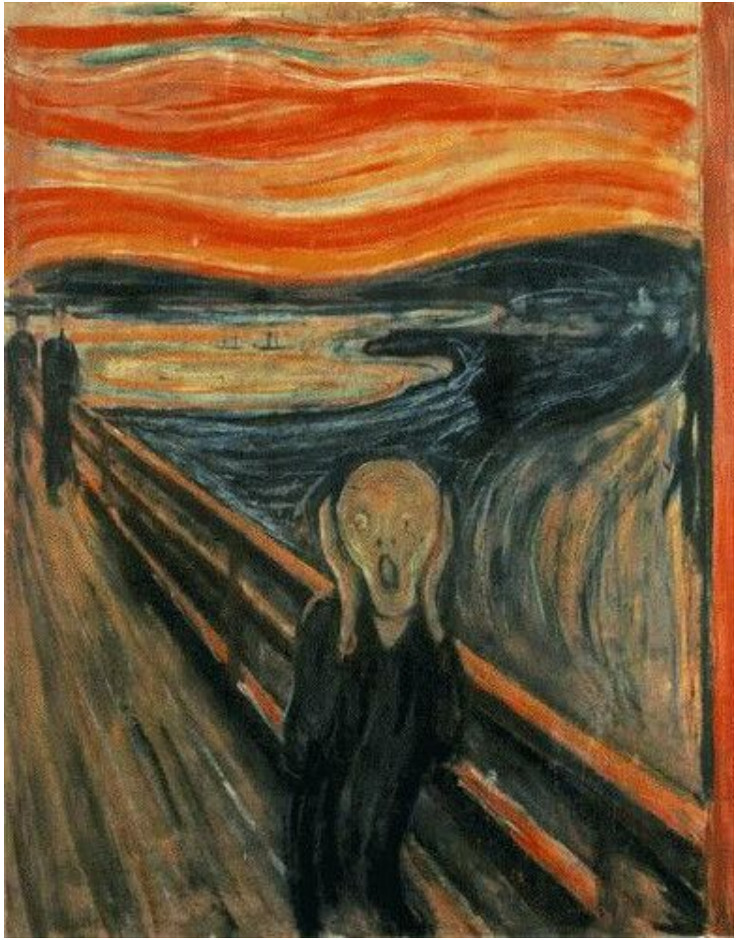
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Transformers

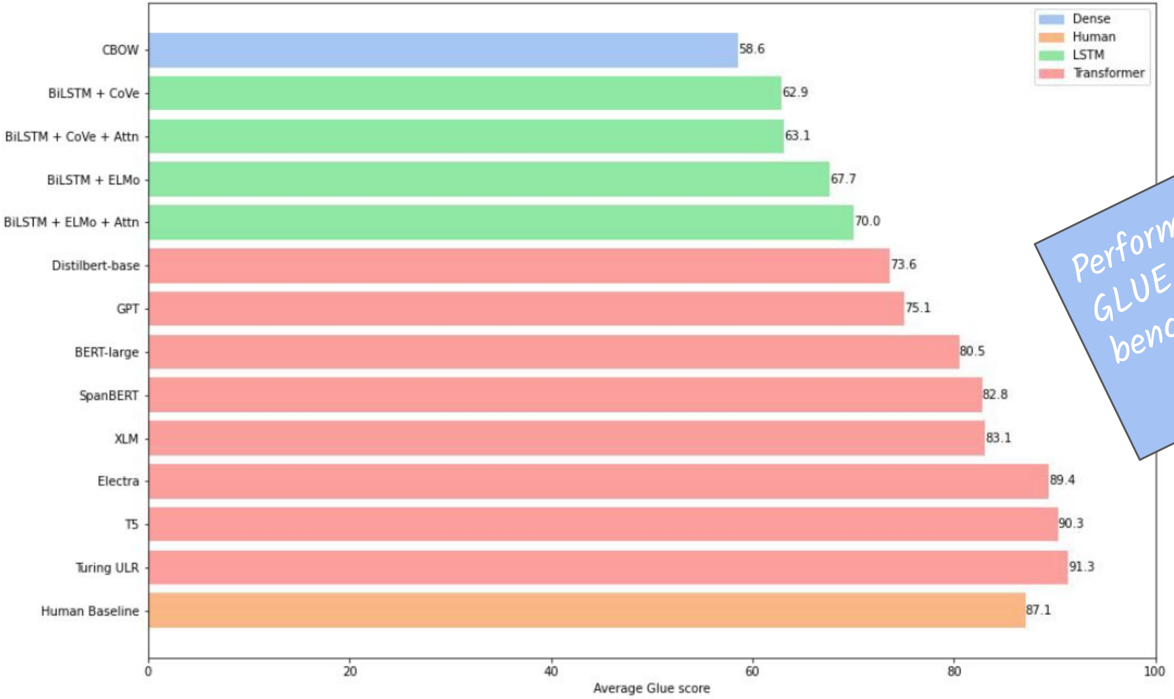
- Neural networks with architectures tailored for generative AI.
- Introduced five years ago, become rapidly popular.
- Most known example: chatGPT.
- Strong aspects:
  - highly efficient,
  - parallel hardware can be used for training.
- Weak aspects
  - huge in size,
  - training has high costs (time, energy, computational resources)
- Wanna try it? **Hugging Face** is your friend.

A note of caution

– At first, it ~~might~~ will seem a nightmare.
– It is complicated, rather than complex.
– In the end, a transformer is nothing but a highly structured neural network.



Original application field: NLP

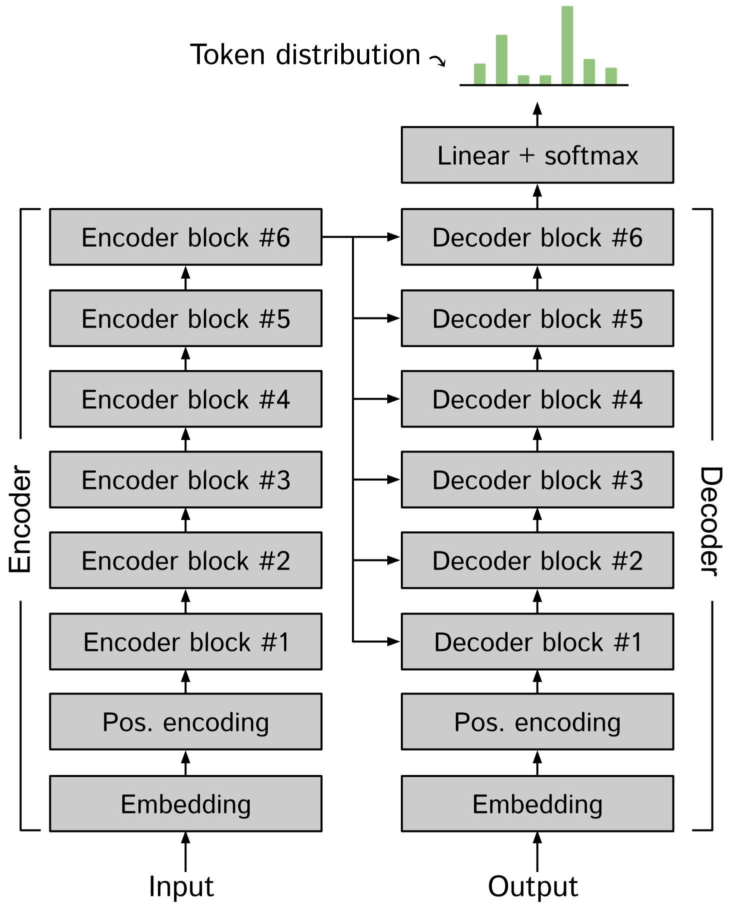


Performance on the GLUE NLP benchmark.

Source: Formation FIDLE [5]

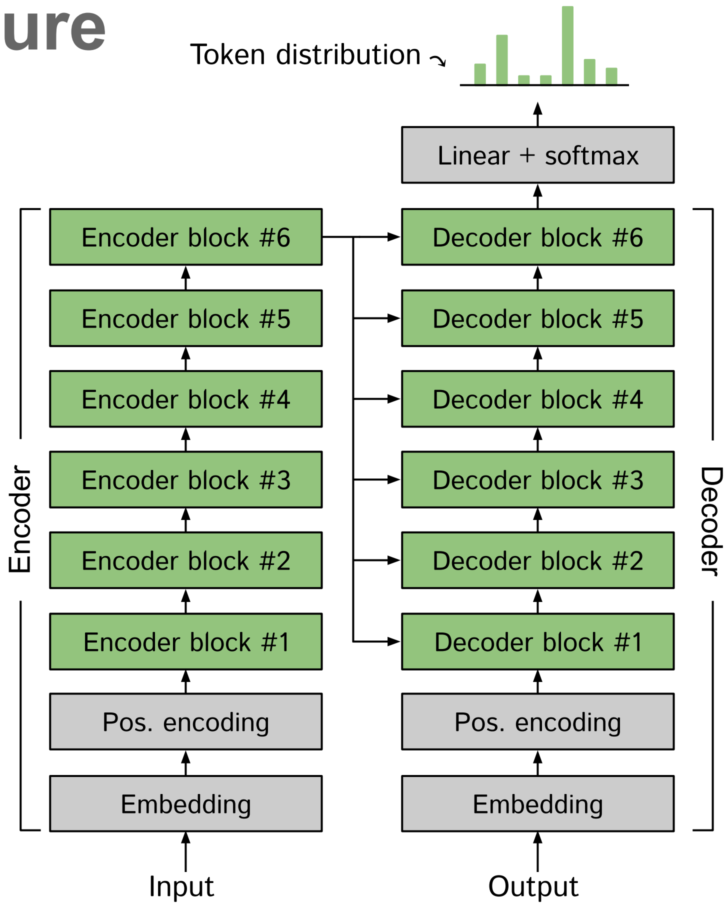
The overall architecture

- Reference: seminal paper from Vaswani et al. [1]
- *It is* intimidating!
- There are repeated patterns.
- Though each block kind needs to be further explained.
- We will do it later, don't worry!
- Idea: as in FFNN or autoencoders: build more and more abstract representations.
- Output is in the model?!?



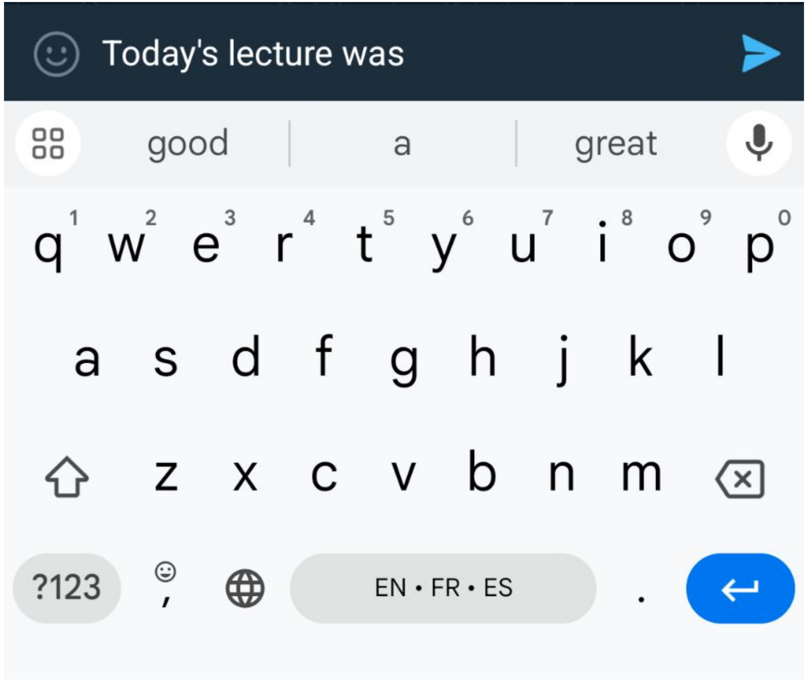
The overall (abstract) architecture

- Reference: seminal paper from Vaswani et al. [1]
- *It is* intimidating!
- There are repeated patterns.
- Though each block kind needs to be further explained.
- We will do it later, don't worry!
- Idea: as in FFNN or autoencoders: build more and more abstract representations.
- Output is in the model?!?



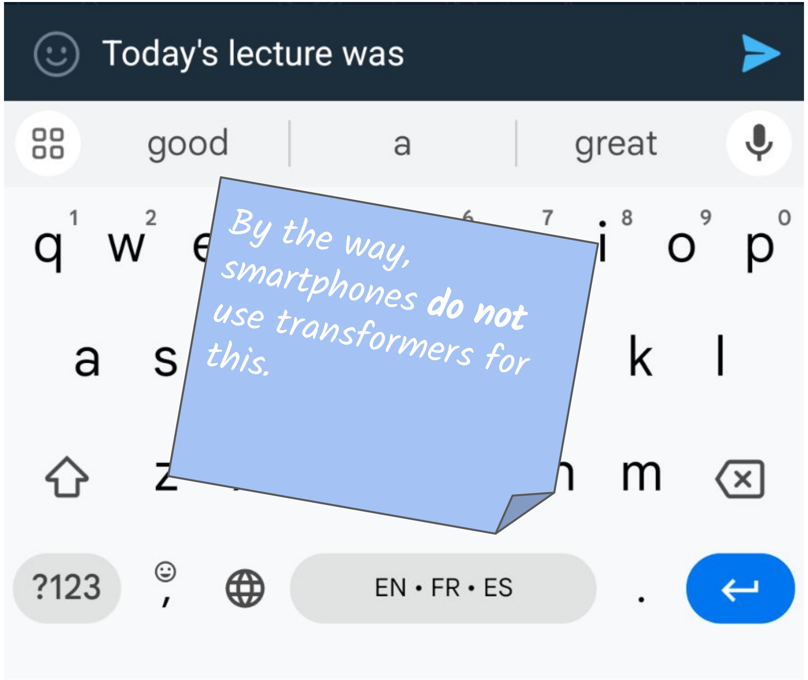
The reference problem

- Textual sequence prediction (generative AI).
- Given the beginning of a text, give suggestions for the next words (possibly completing the sentence).
- But there is more than that (e.g., synthesis of images or sound).
- Everyday example: augmented keyboards in smartphones.



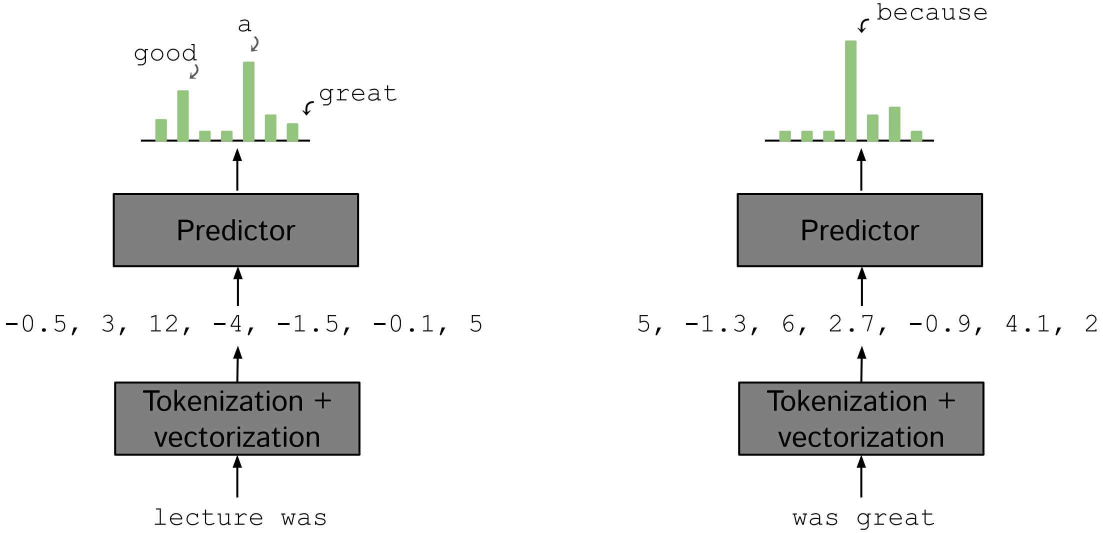
The reference problem

- Textual sequence prediction (generative AI)
- Given the beginning of a text, give suggestions for the next word
- But there is more than that (e.g., synthesis of images or sound)
- Everyday example: augmented keyboards in smartphones



How is this problem tackled?

– Text vectorization + next token prediction (via softmax output).
– Bonus: text generation via autoregression.



– Most of the times, predictor is a RNN.
– Not suitable for long text generation.
– Lack of long-term context.

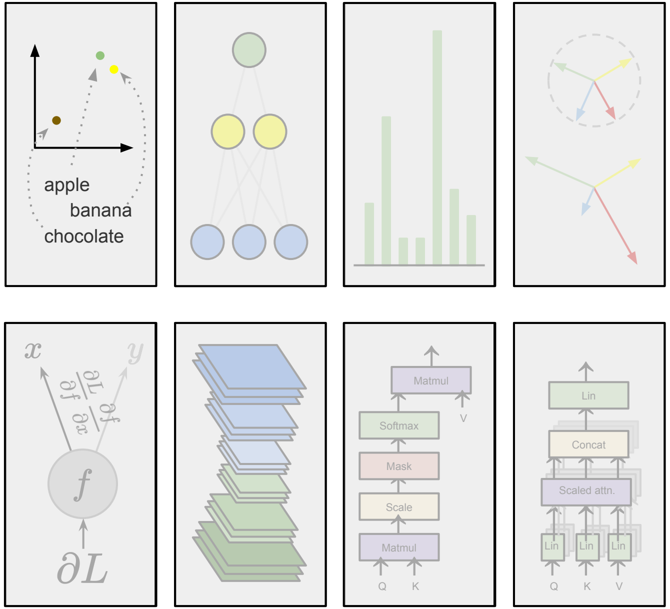
Results depend on the choice of
– tokens
– n-grams

Transformer ingredients

- Embeddings.
- Feed-forward neural networks.
- Soft-max activation.
- Normalization.
- Backpropagation.
- Encoder/decoder architecture.
- Attention.
- Multi-head attention.

You know this

New stuff!

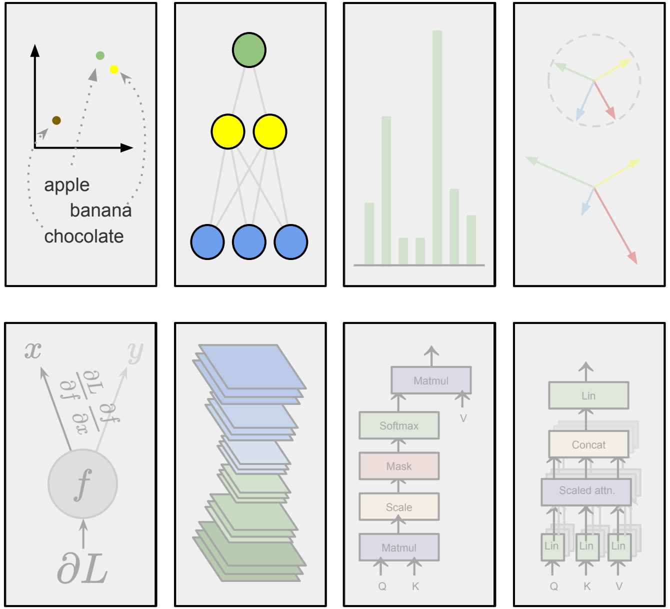


Transformer ingredients

- Embeddings.
- Feed-forward neural networks.
- Soft-max activation.
- Normalization.
- Backpropagation.
- Encoder/decoder architecture.
- Attention.
- Multi-head attention.

You know this

New stuff!

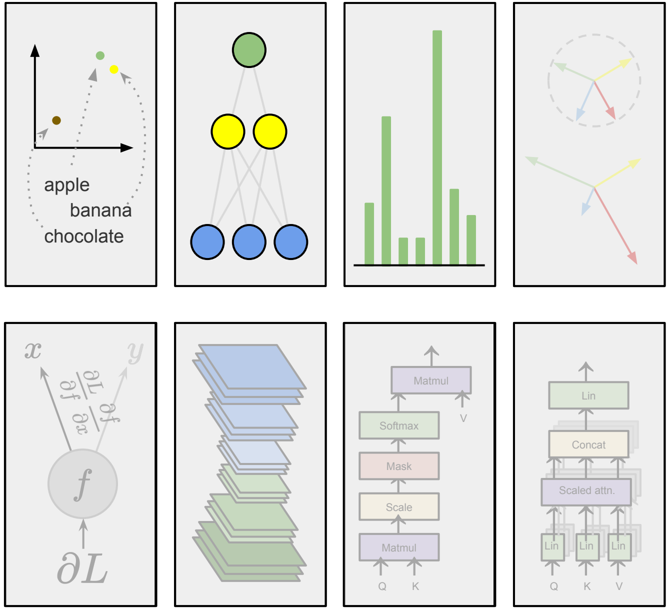


Transformer ingredients

- Embeddings.
- Feed-forward neural networks.
- Soft-max activation.
- Normalization.
- Backpropagation.
- Encoder/decoder architecture.
- Attention.
- Multi-head attention.

You know this

New stuff!

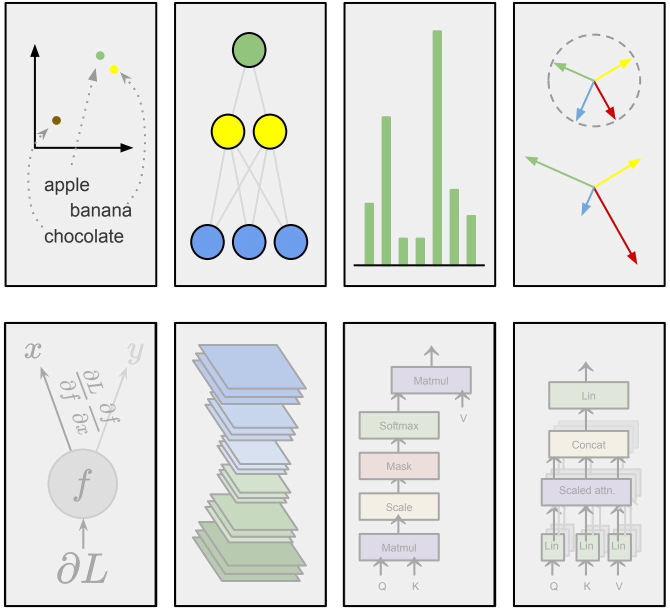


Transformer ingredients

- Embeddings.
- Feed-forward neural networks.
- Soft-max activation.
- Normalization.
- Backpropagation.
- Encoder/decoder architecture.
- Attention.
- Multi-head attention.

You know this

New stuff!

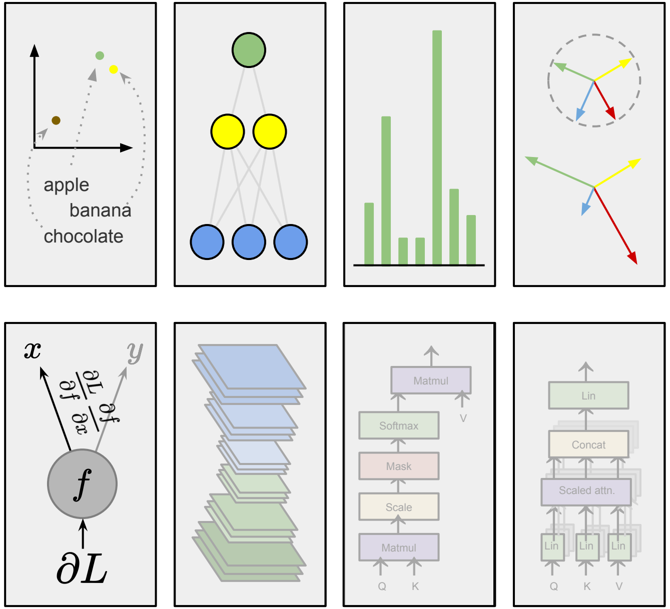


Transformer ingredients

- Embeddings.
- Feed-forward neural networks.
- Soft-max activation.
- Normalization.
- Backpropagation.
- Encoder/decoder architecture.
- Attention.
- Multi-head attention.

You know this

New stuff!



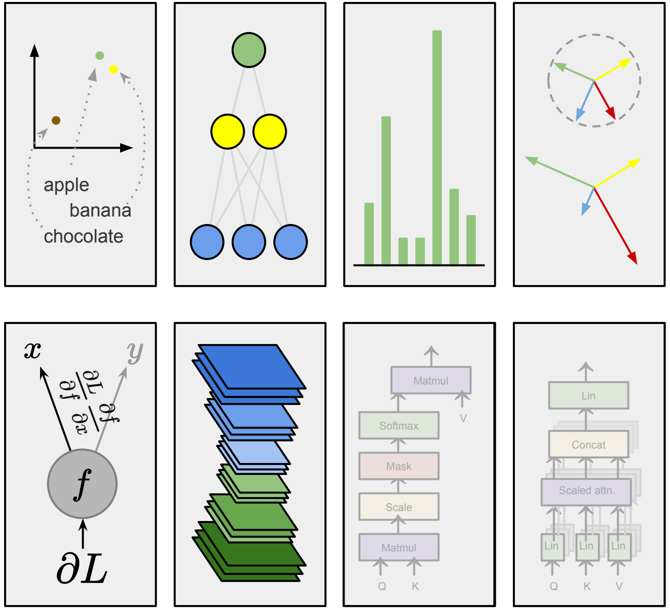
Transformer ingredients

- Embeddings.
- Feed-forward neural networks.
- Soft-max activation.
- Normalization.
- Backpropagation.
- Encoder/decoder architecture.

You know this

- Attention.
- Multi-head attention.

New stuff!

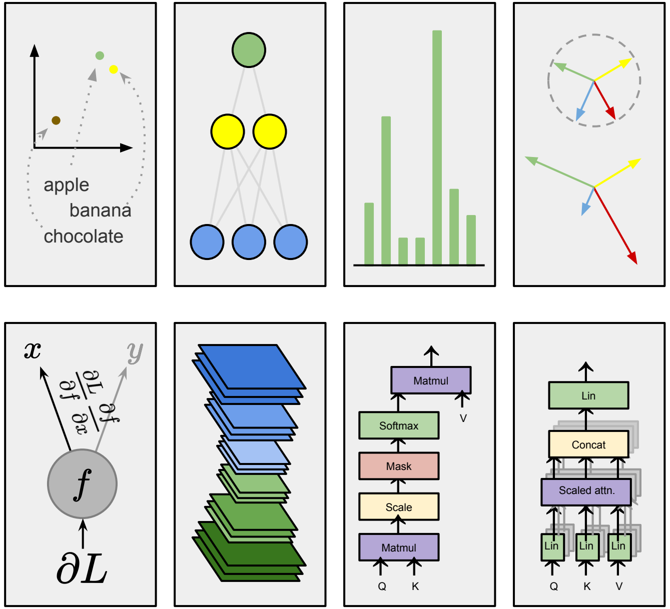


Transformer ingredients

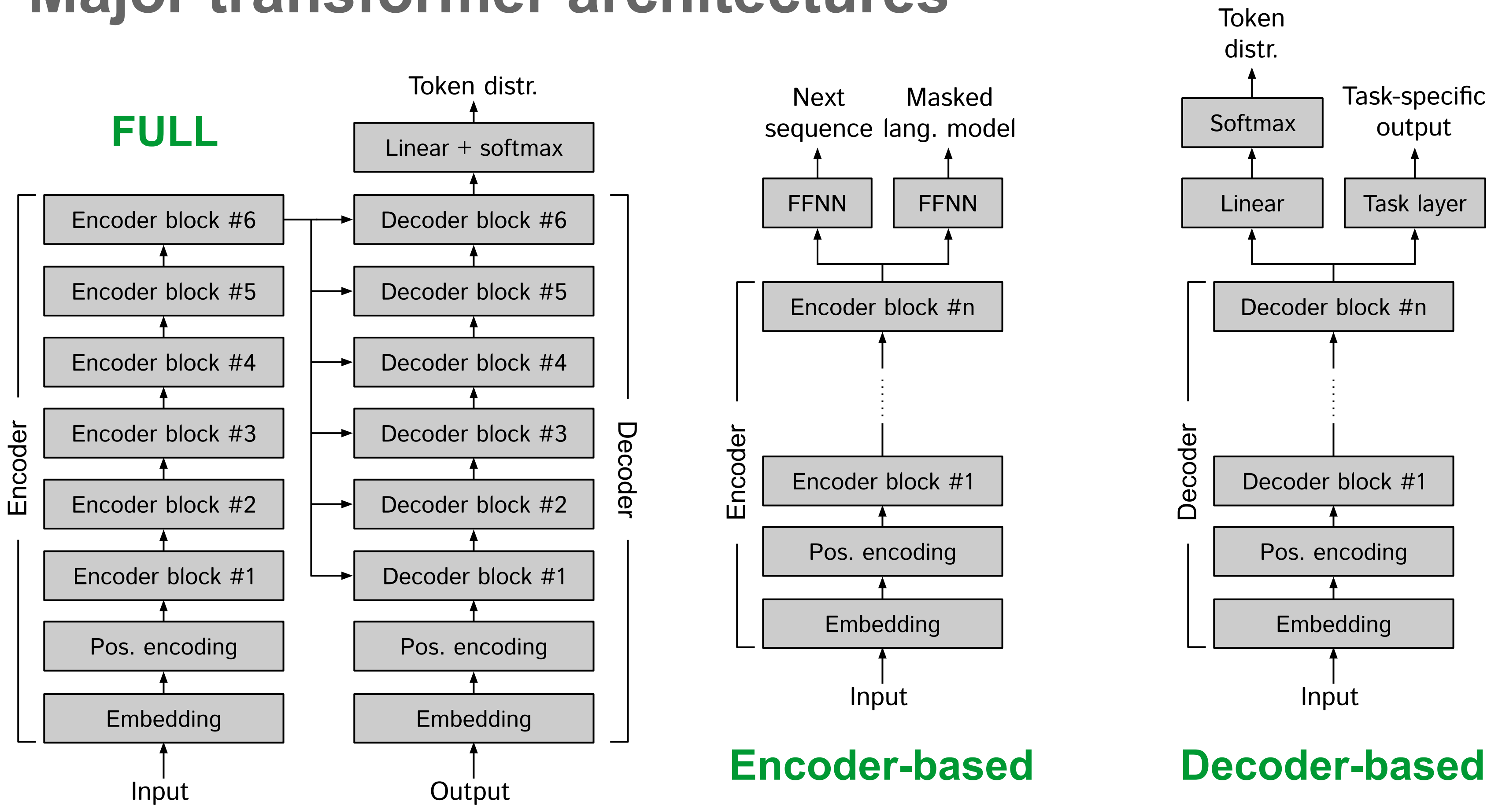
- Embeddings.
- Feed-forward neural networks.
- Soft-max activation.
- Normalization.
- Backpropagation.
- Encoder/decoder architecture.
- Attention.
- Multi-head attention.

You know this

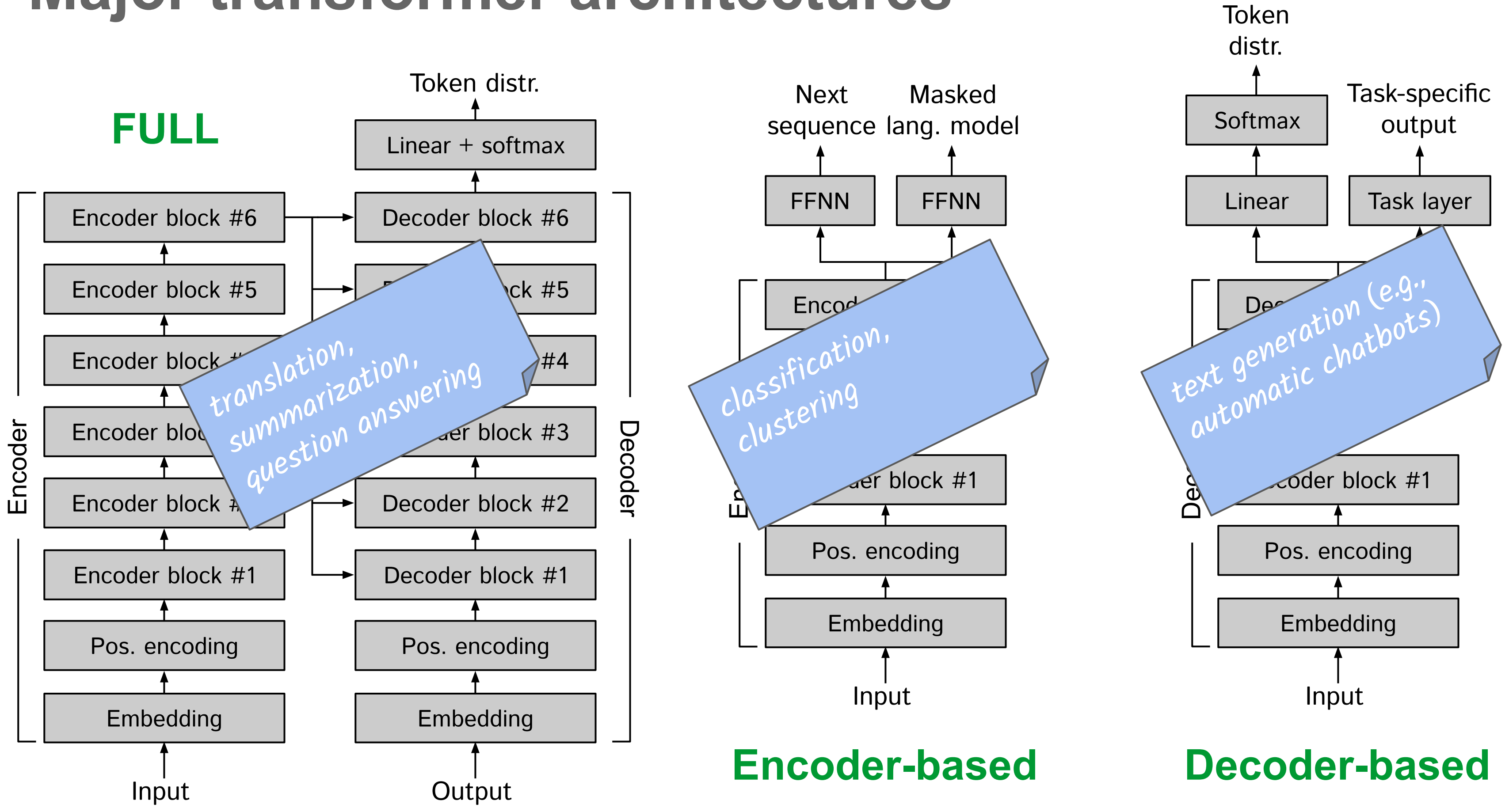
New stuff!



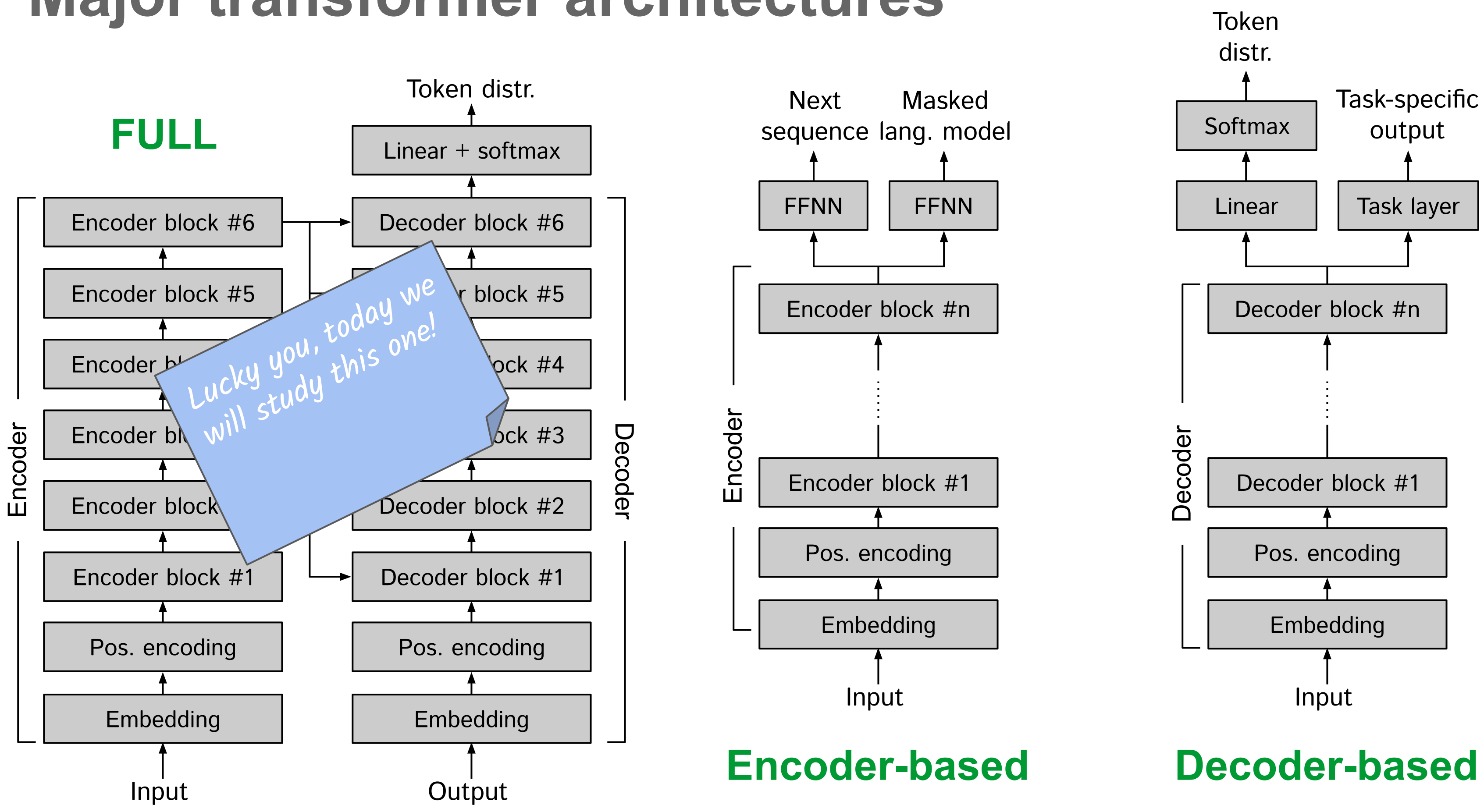
Major transformer architectures



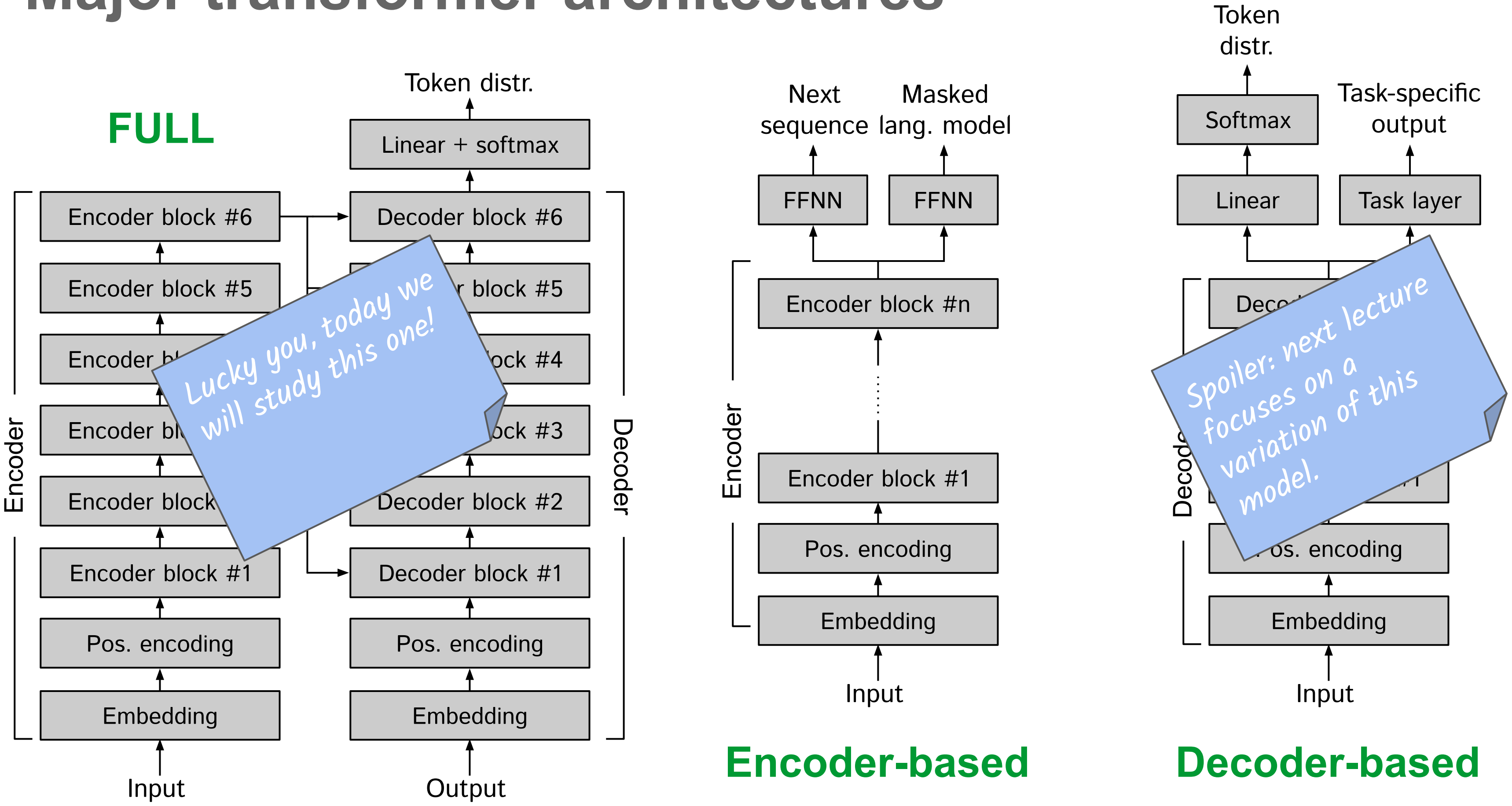
Major transformer architectures



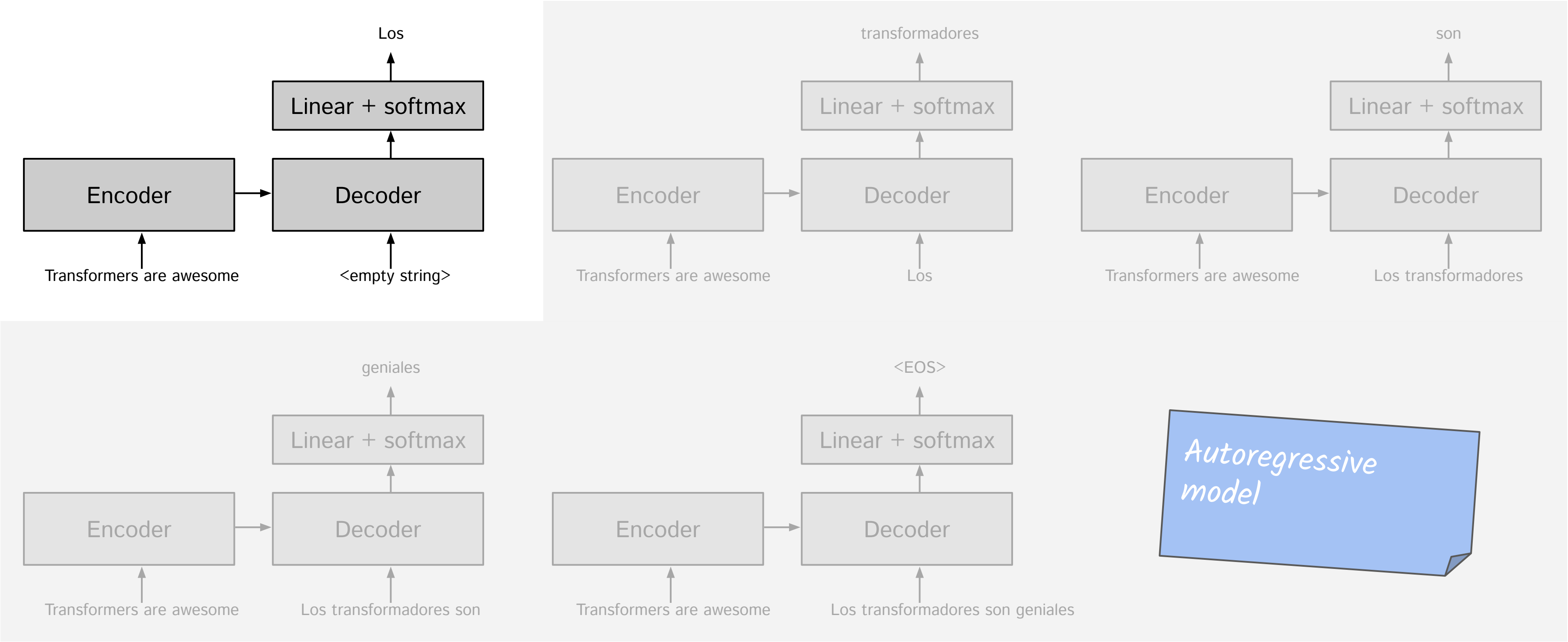
Major transformer architectures



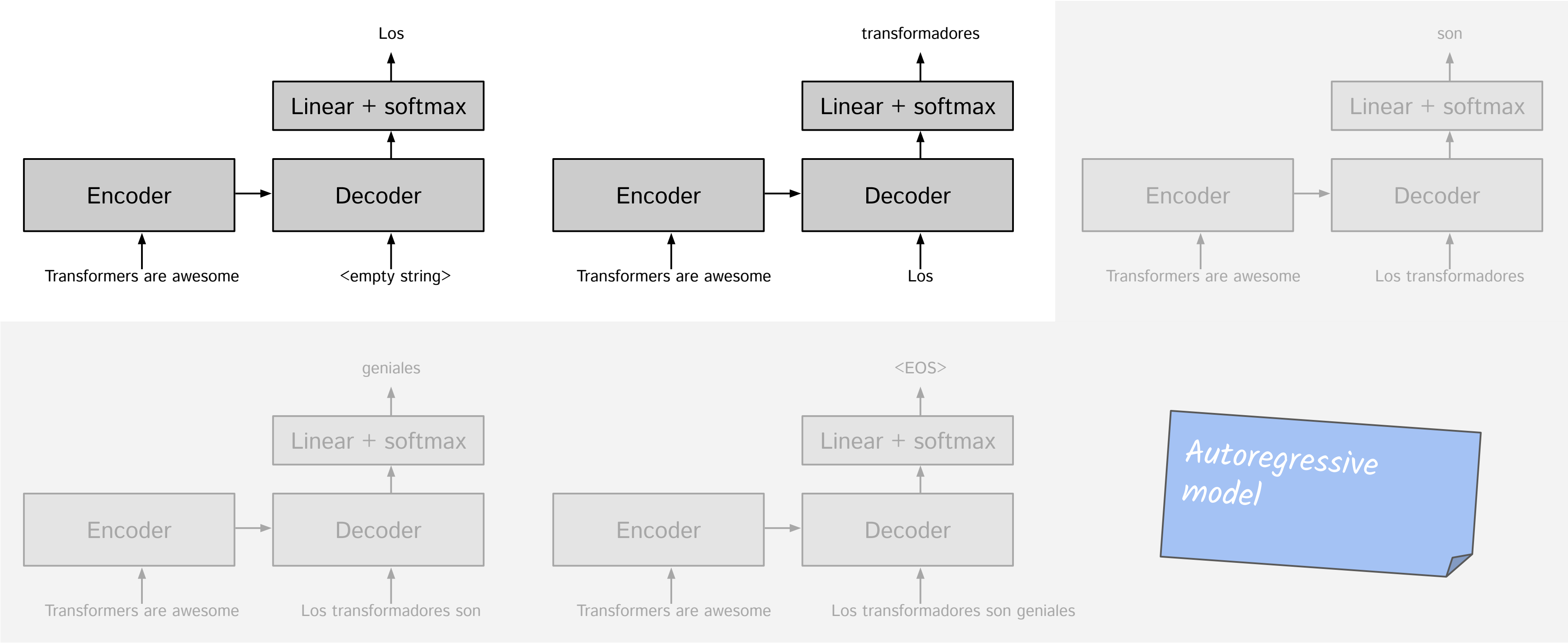
Major transformer architectures



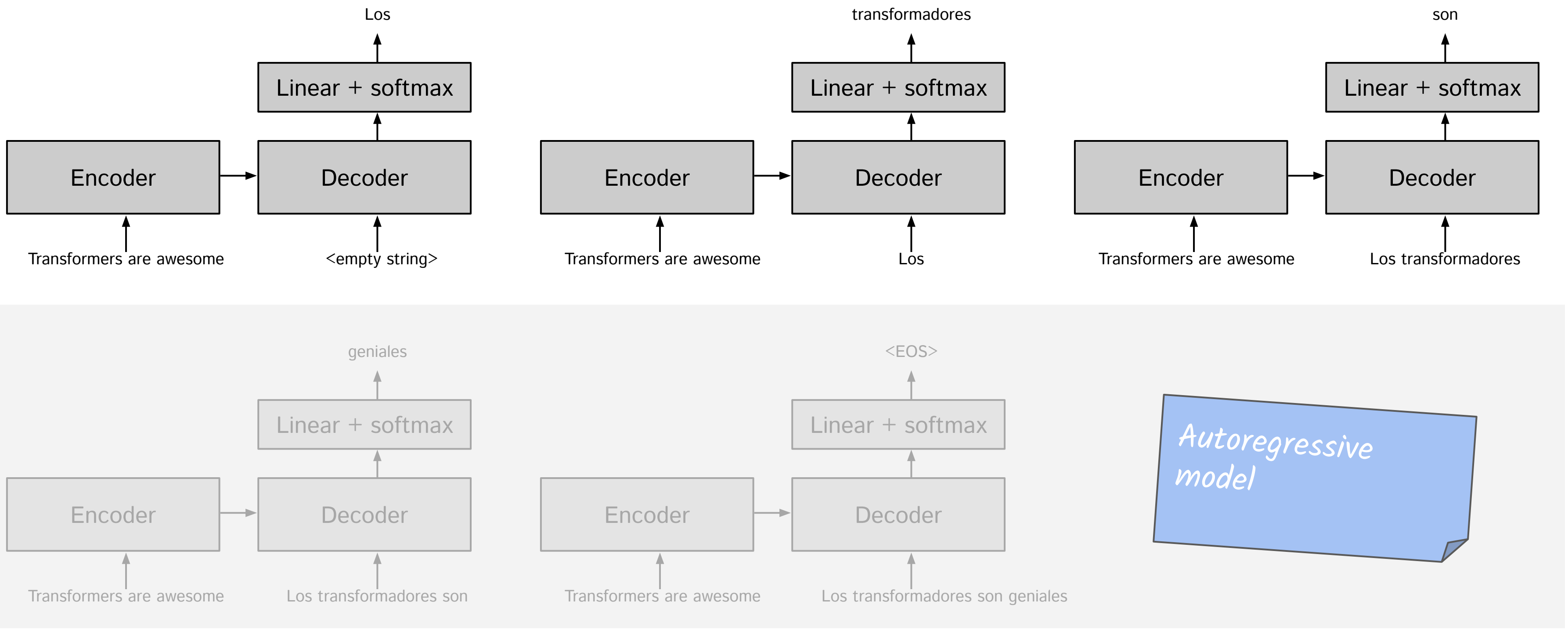
How is a transformer queried?



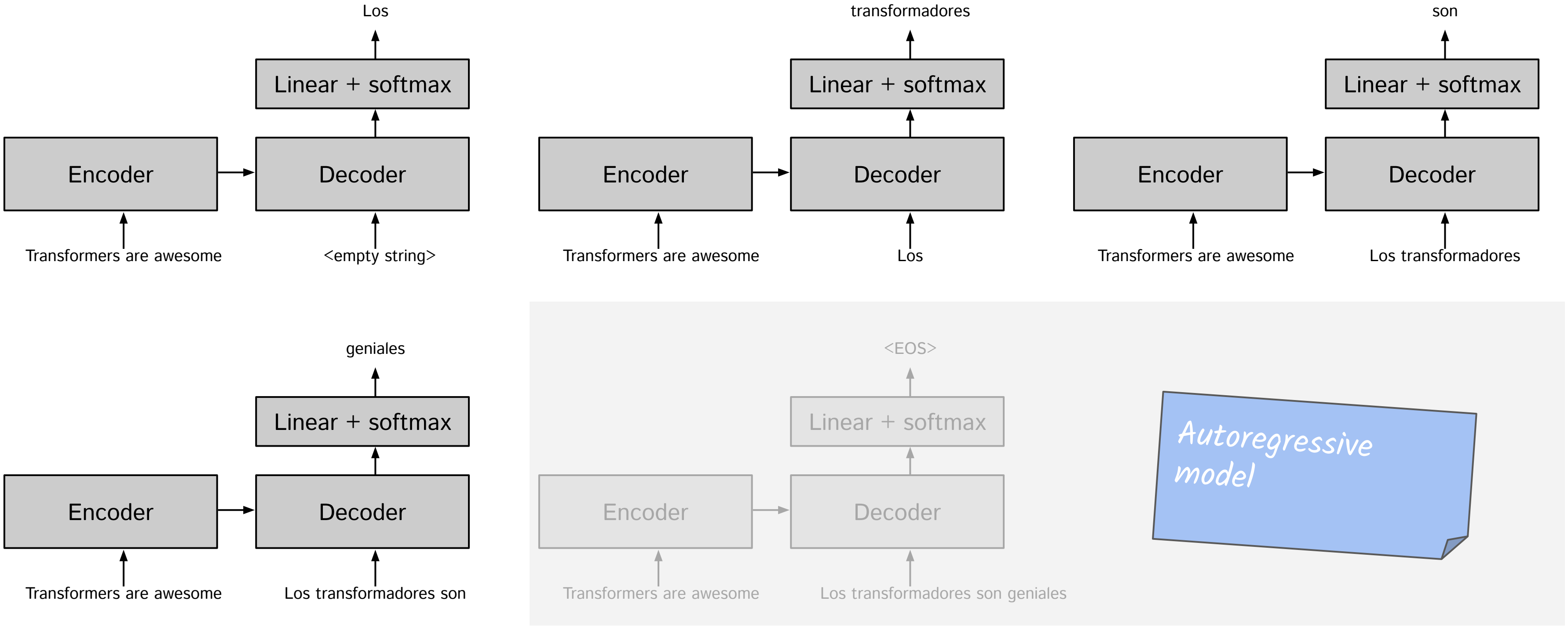
How is a transformer queried?



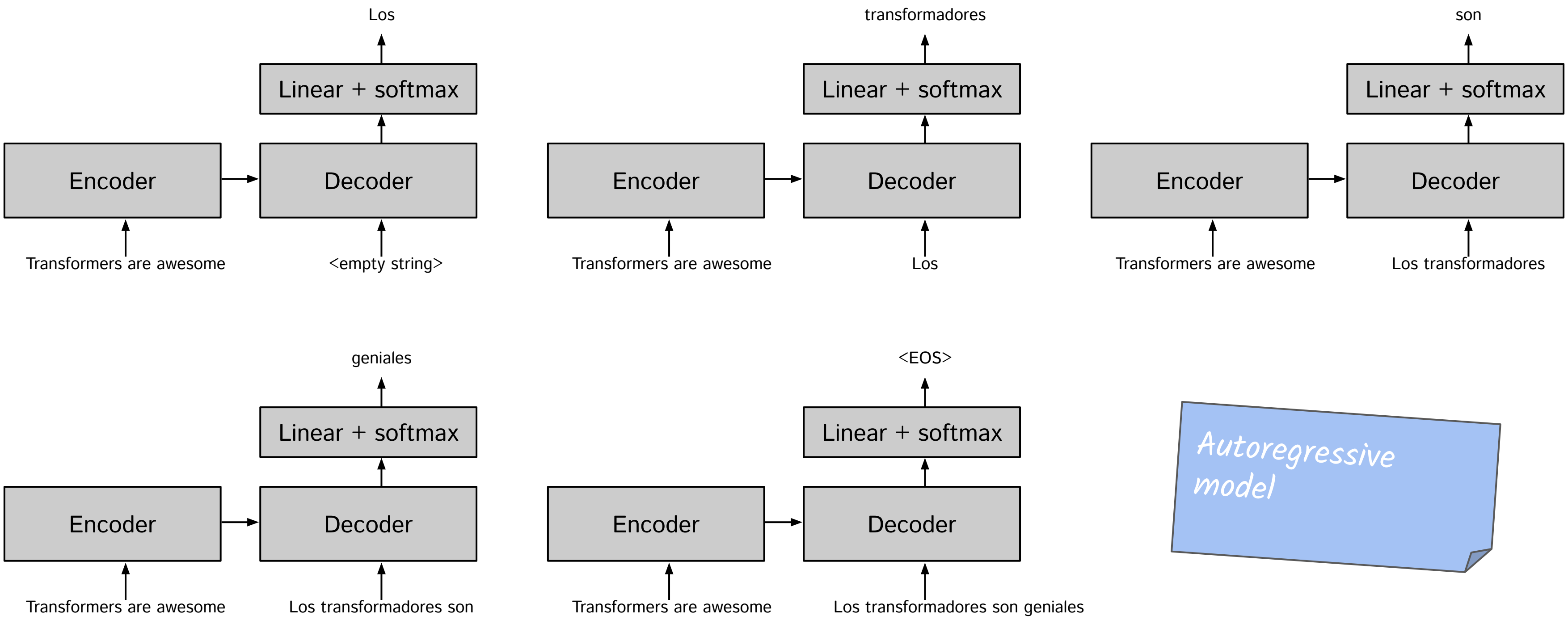
How is a transformer queried?



How is a transformer queried?

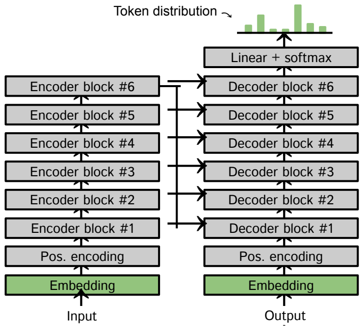


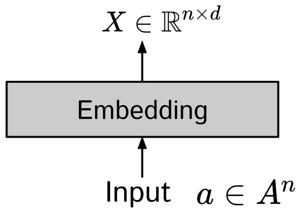
How is a transformer queried?



Embedding

– Purpose: mapping text to numbers.
– Two phases:
  1. text tokenization, tokens in set A (not necessarily w.r.t. words),
  2. token embedding with dimension $d \in \mathbb{N}$.

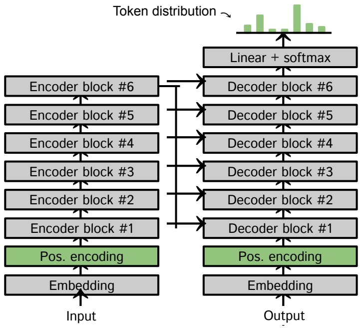


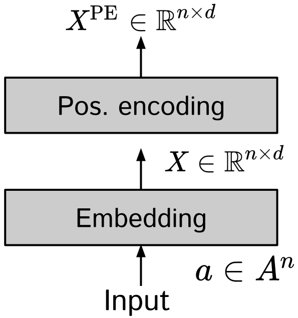


In the original paper, $d = 512$ throughout the model.

Positional encoding

- Note: subsequent processing is insensitive to the order of tokens.
- Purpose: injecting positional information of tokens.

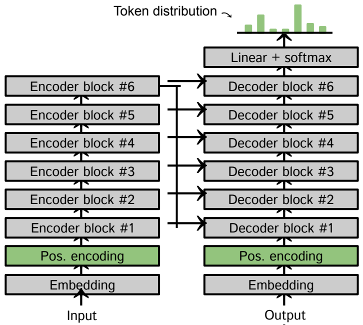




$$
\begin{bmatrix}
x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{16} \\
x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} \\
x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} \\
x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48}
\end{bmatrix}
+
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1
\end{bmatrix}
$$

Position is encoded in terms of frequency of 0 and 1 in token embedding.

Positional encoding

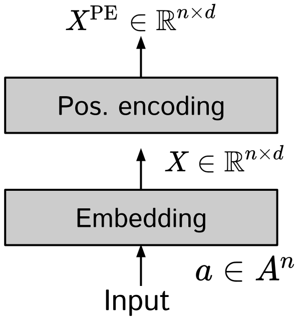


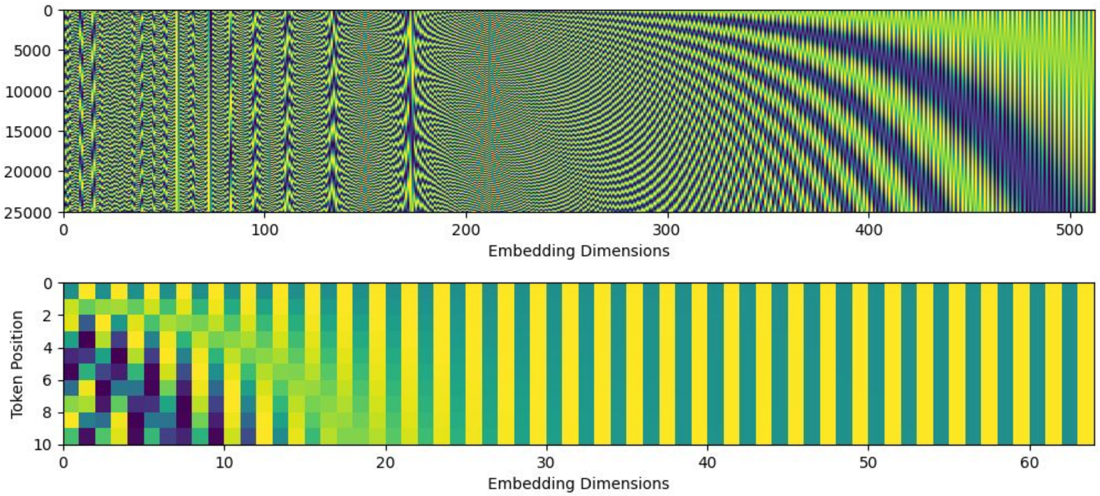
– Note: subsequent processing is insensitive to the order of tokens.

– Purpose: injecting positional information of tokens.

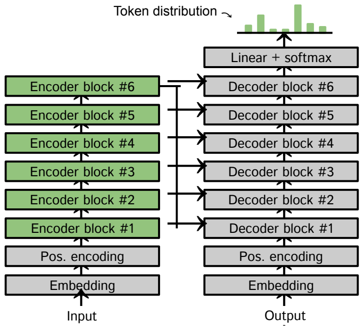
$$\mathrm{PE}(\mathrm{pos}, 2i) = \sin\left(\frac{\mathrm{pos}}{10000^{2i/d}}\right)$$

$$\mathrm{PE}(\mathrm{pos}, 2i+1) = \cos\left(\frac{\mathrm{pos}}{10000^{2i/d}}\right)$$

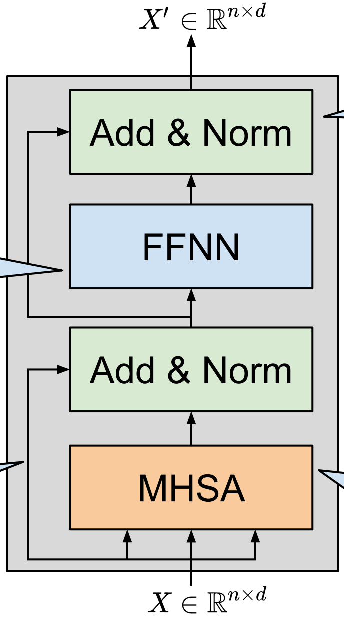




Encoder block



– Purpose: learning/computing abstract representations.



Position-wide feed-forward neural network learns representations.

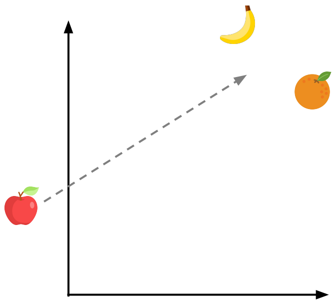
Residual connections counteract vanishing & exploding gradients.

Layer (not batch!) normalization improves convergence.

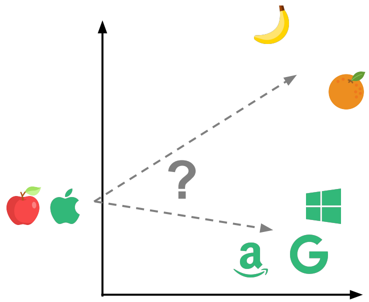
Multi-head self-attention learns long-term dependencies and performs disambiguation.

Self-attention mechanism

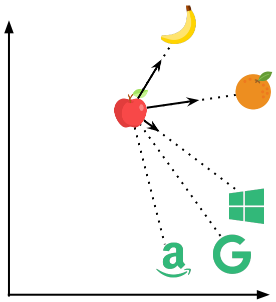
– Purpose: dealing with long-term context and disambiguation.



If there is a single, definite meaning of a word, we rely on the used embedding.



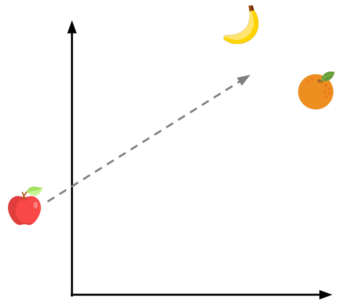
What about the (frequent) case this is not true?



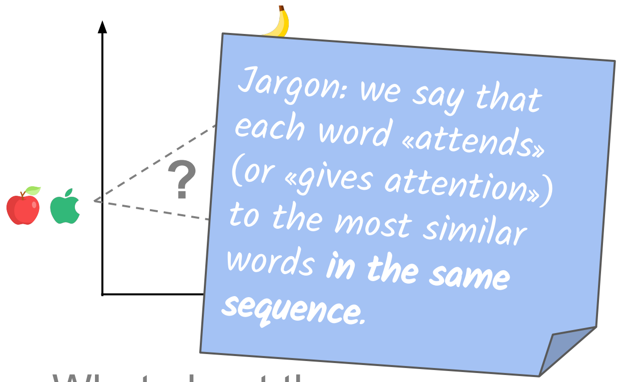
Solution: move towards the more «contextually» similar words.

Self-attention mechanism

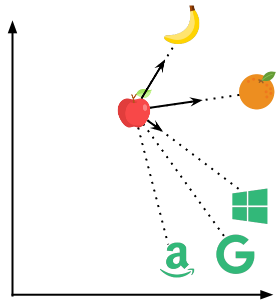
– Purpose: dealing with long-term context and disambiguation.



If there is a single, definite meaning of a word, we rely on the used embedding.



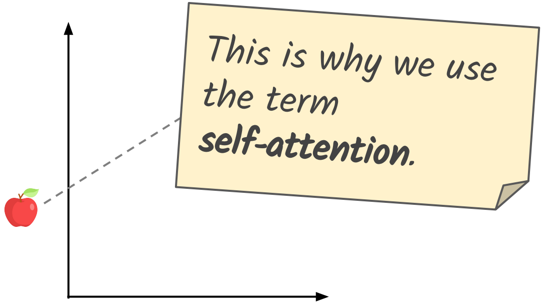
What about the (frequent) case this is not true?



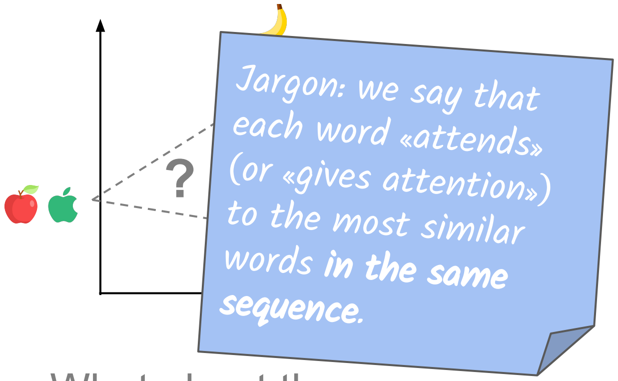
Solution: move towards the more «contextually» similar words.

Self-attention mechanism

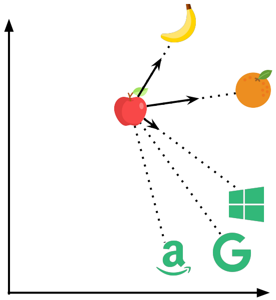
– Purpose: dealing with long-term context and disambiguation.



If there is a single, definite meaning of a word, we rely on the used embedding.



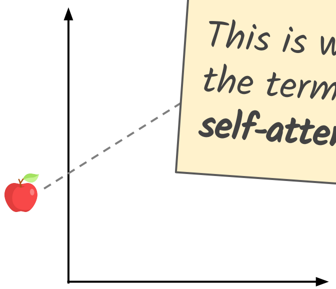
What about the (frequent) case this is not true?



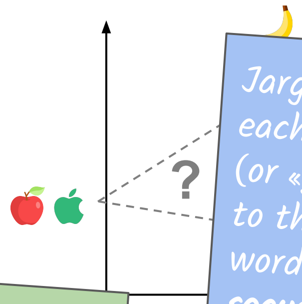
Solution: move towards the more «contextually» similar words.

Self-attention mechanism

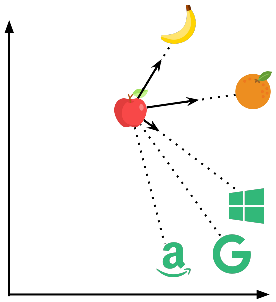
– Purpose: dealing with long-term context and disambiguation.



This is why we use the term ***self-attention.***



Jargon: we say that each word «attends» (or «gives attention») to the most similar words ***in the same sequence.***



Moreover, each word can attend each other word in the sequence, thus learning long-term dependencies.

If there is a singl
definite meaning
a word, we rely o
the used
embedding.

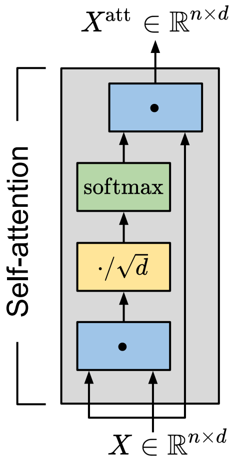
at about the
quent) case this
not true?

Solution: move towards the more «contextually» similar words.

Attention in practice

– Similarity is well caught by inner products, but
  – inner product values tend to increase with d,
  – inner product values can be negative.
– Solution:
  – normalize by dividing by $\sqrt{d}$,
  – subsequently apply softmax.

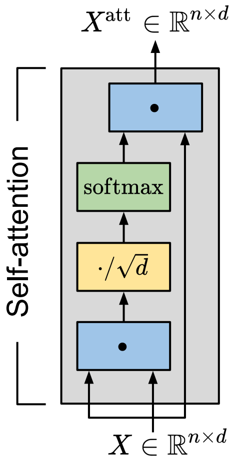
$$X^{\mathrm{att}} = \underbrace{\overbrace{\mathrm{softmax}\left(\frac{X \cdot X^{\top}}{\sqrt{d}}\right)}^{\text{computes scaled similarities between encoded words}} \cdot X}_{\text{moves each encoded word towards more similar ones}}$$



Attention in practice

- Similarity is well caught by inner products, but
  - inner product values tend to increase with d,
  - inner product values can be negative.
- Solution:
  - normalize by dividing by $\sqrt{d}$,
  - subsequently apply softmax.

$$X^{\text{att}} = \underbrace{\overbrace{\text{softmax}\left(\frac{X\cdot X^\top}{\sqrt{d}}\right)}^{\text{computes scaled similarities between encoded words}} \cdot X}_{\text{moves each encoded word towards more similar ones}}$$

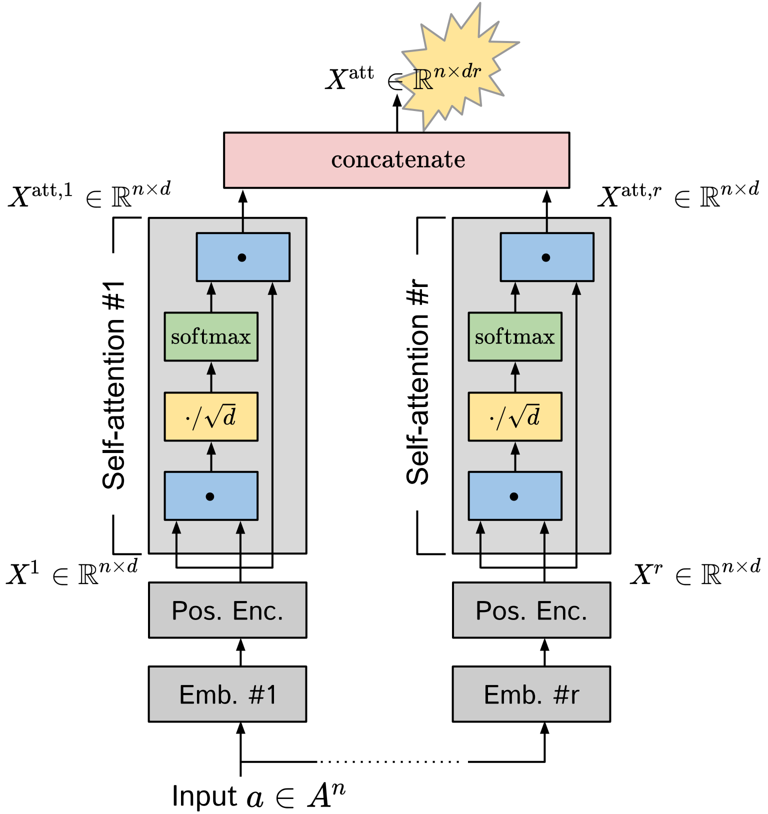


What about the use of a same word with different meanings?

Solution: ***multi-head self-attention.***

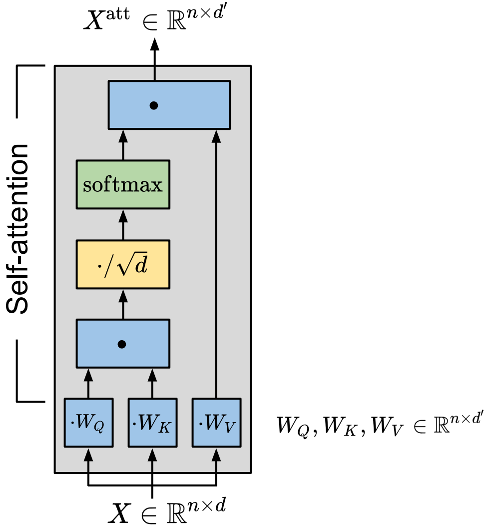
Multi-head self-attention

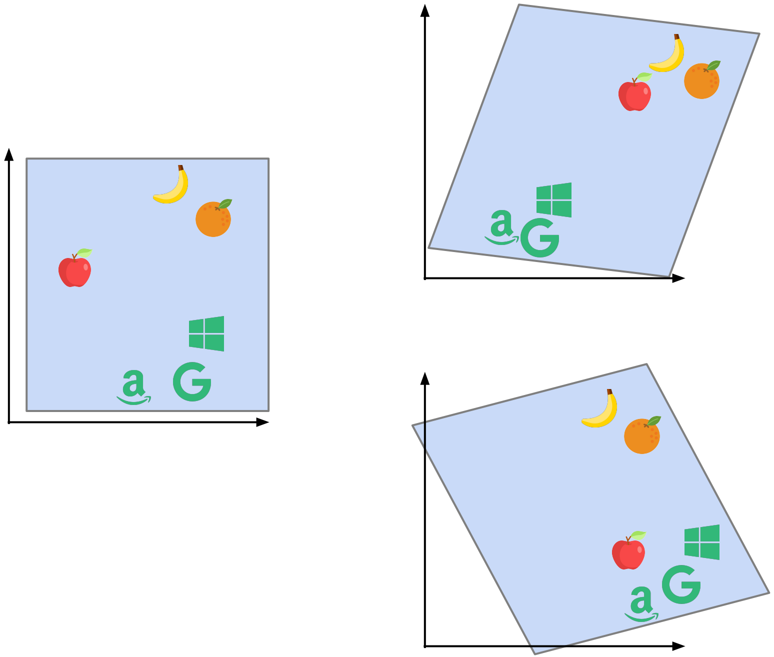
– Embedding conveyes semantics.
– Hint: use several embeddings.
– But good embeddings are hard to find.
– Moreover: dimensionality explosion in output when stacking several encoders.



Linear transformations is the new embedding

- Keep only a valid embedding.
- Use different linear mappings.

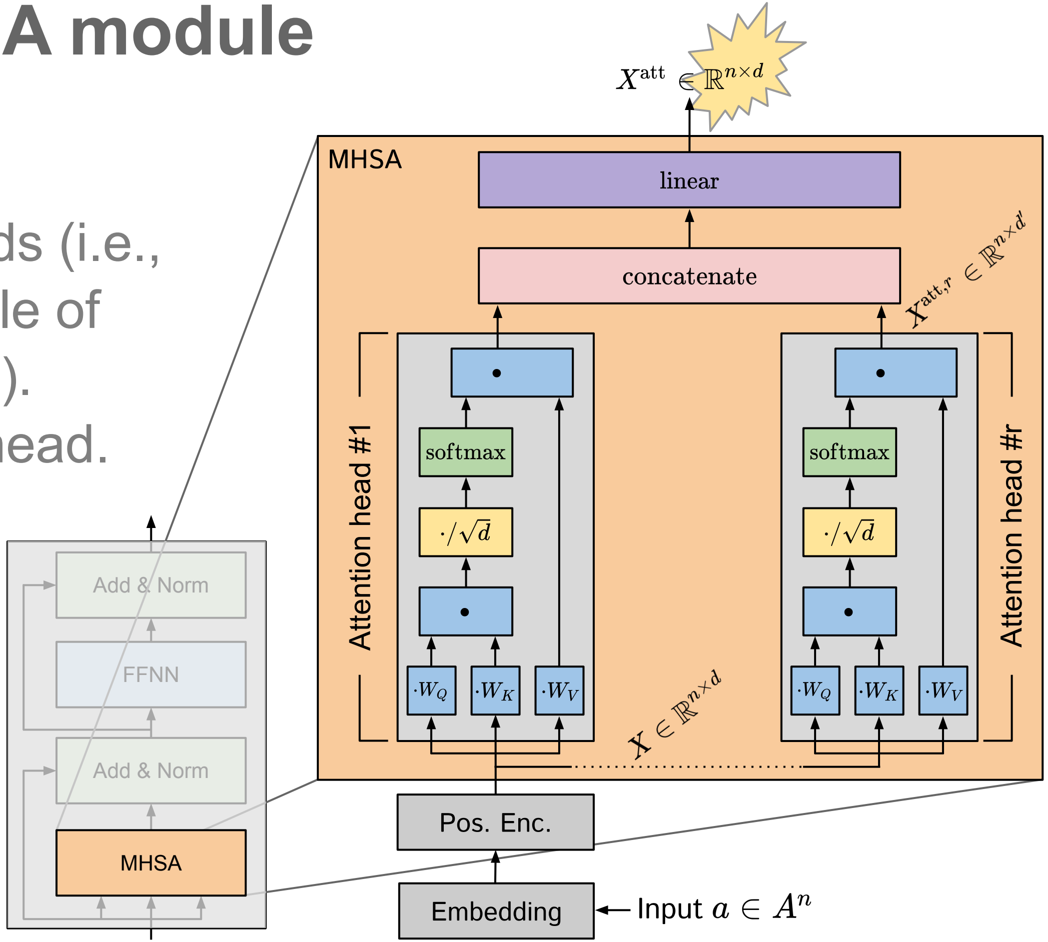




Summing up: MHSA module

- Same encoding.
- r different attention heads (i.e., each has a different triple of transformation matrices).
- Dimension d' for each head.
- If we choose $rd' = d$....
- ...dimensionality is preserved.

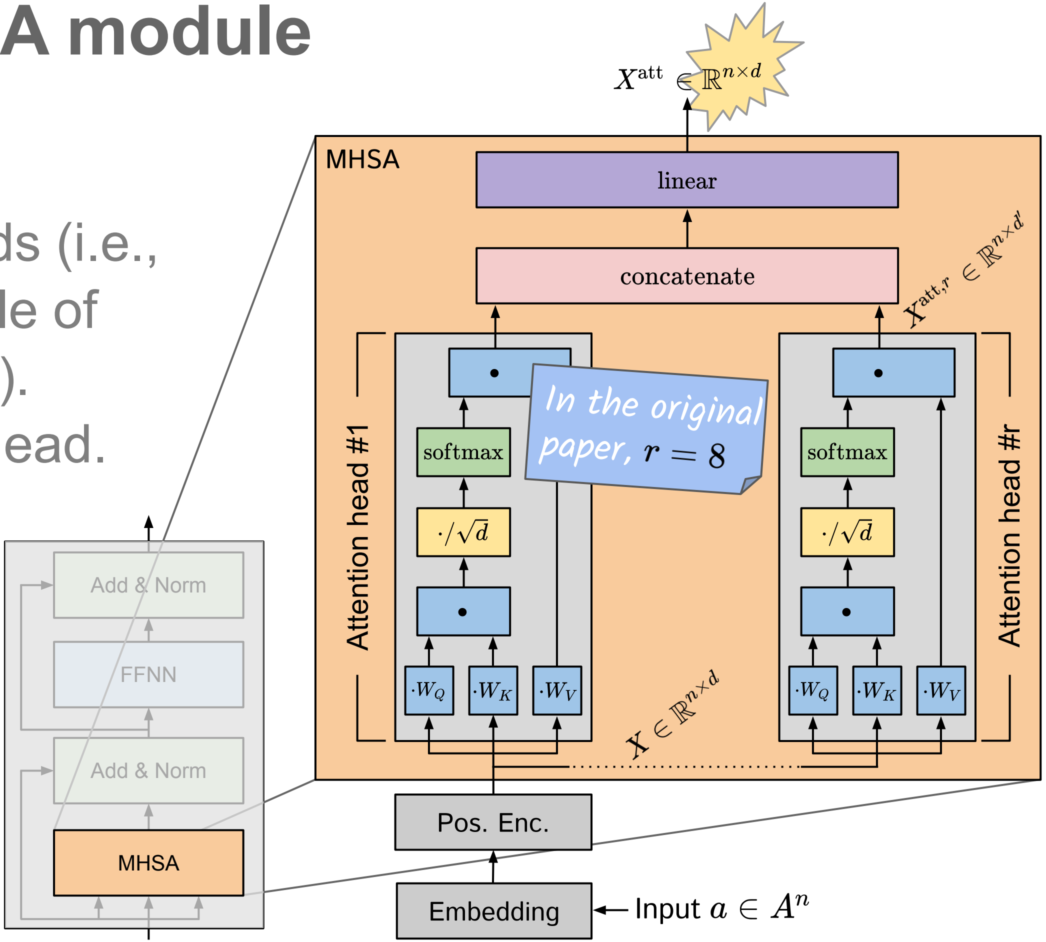
$$X^{\text{att},i} = \text{softmax}\left(\frac{W_Q X \cdot (W_K X)^\top}{\sqrt{d}}\right) \cdot (W_V X)$$



Summing up: MHSA module

- Same encoding.
- r different attention heads (i.e., each has a different triple of transformation matrices).
- Dimension d' for each head.
- If we choose $rd' = d$….
- …dimensionality is preserved.

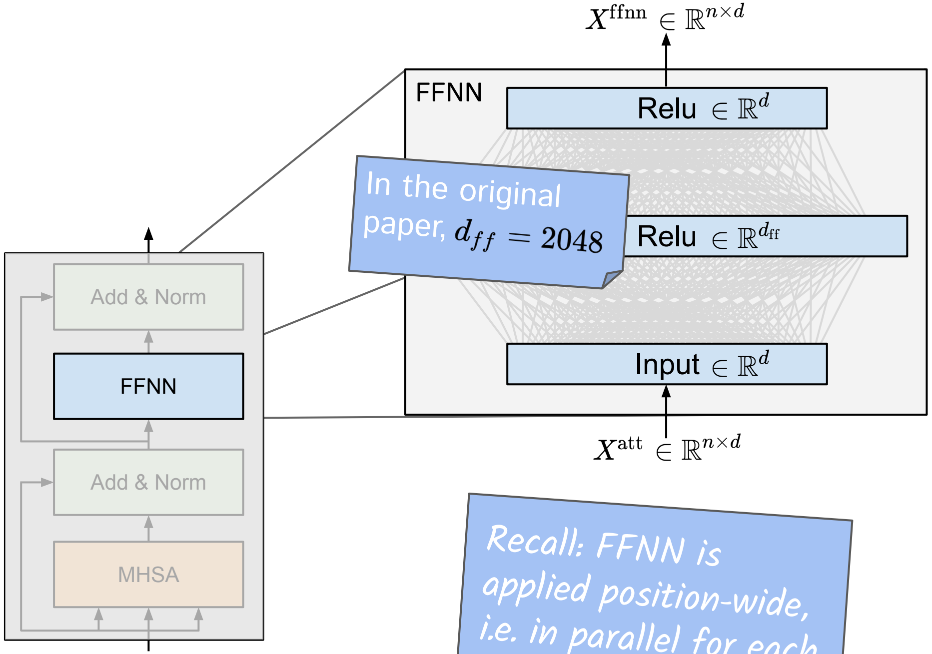
$$X^{\mathrm{att},i} = \mathrm{softmax}\left(\frac{W_Q X \cdot (W_K X)^\top}{\sqrt{d}}\right) \cdot (W_V X)$$



Position-wide feed-forward neural network

- One hidden layer.
- ReLU activation.
- Preserves input dimensionality.
- Applied token-wise.
- Possibly executed in parallel through tokens in a sequence.

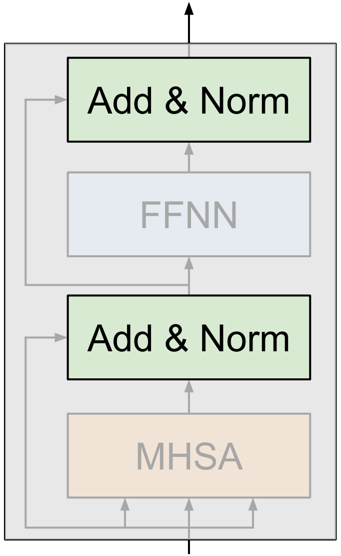
$$X^{\text{ffnn}} = \max(0, X^{\text{att}} \cdot W_1 + b_1) \cdot W_2 + b_2$$

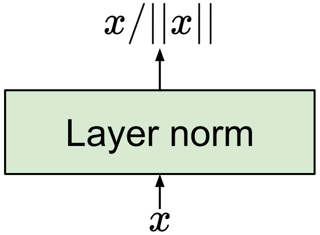


Recall: FFNN is applied position-wide, i.e. in parallel for each token

Add & Norm

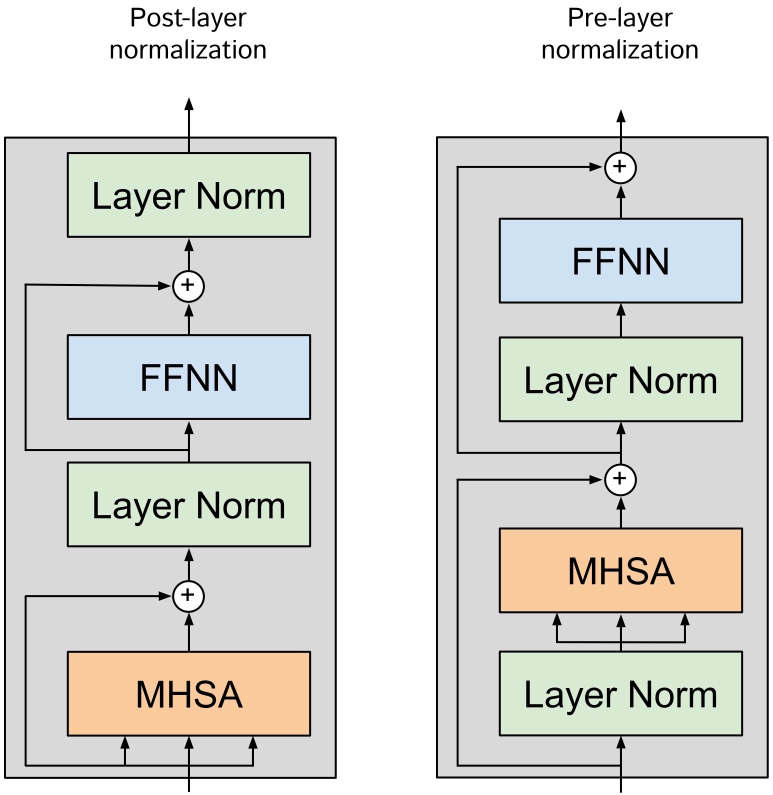
– Add: residual connection operation.
– Norm: layer normalization (possibly in parallel).





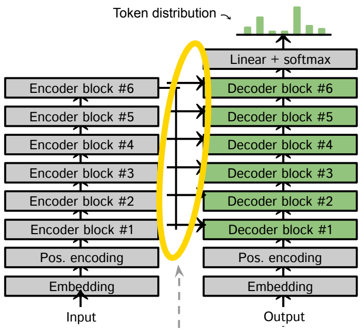
Add & Norm

- Note that sometimes it's add & norm…
- … but sometimes it's norm & add!



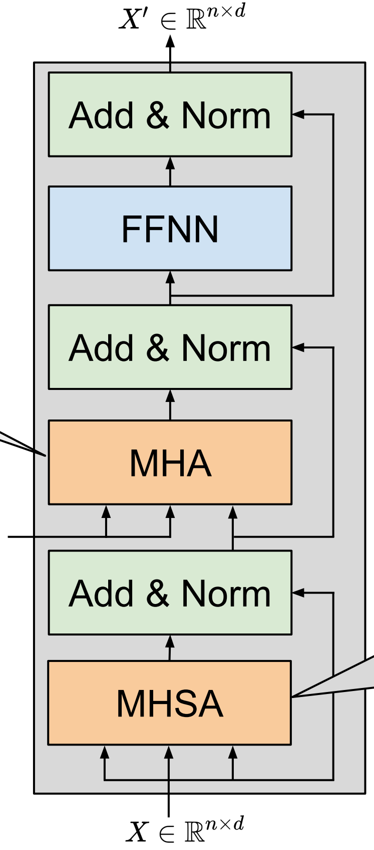
Decoder block

– Purpose: learning/computing refined views of the data.



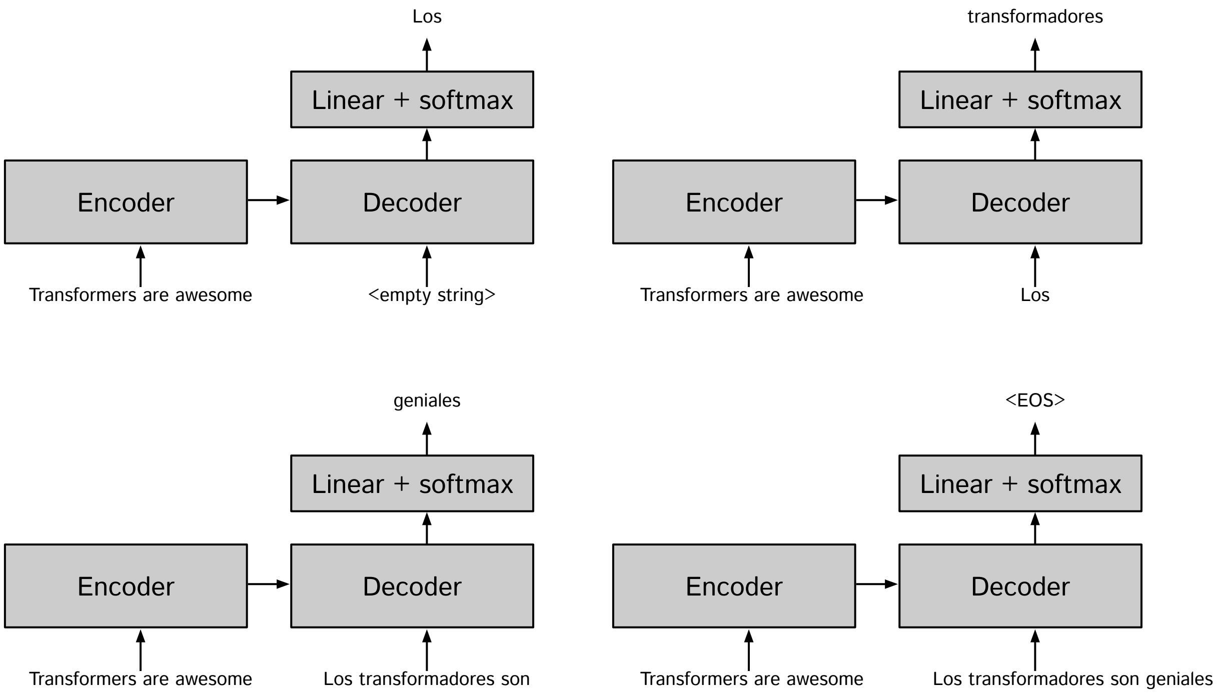
Attention rather than self-attention, …

… as this info comes from outside the module



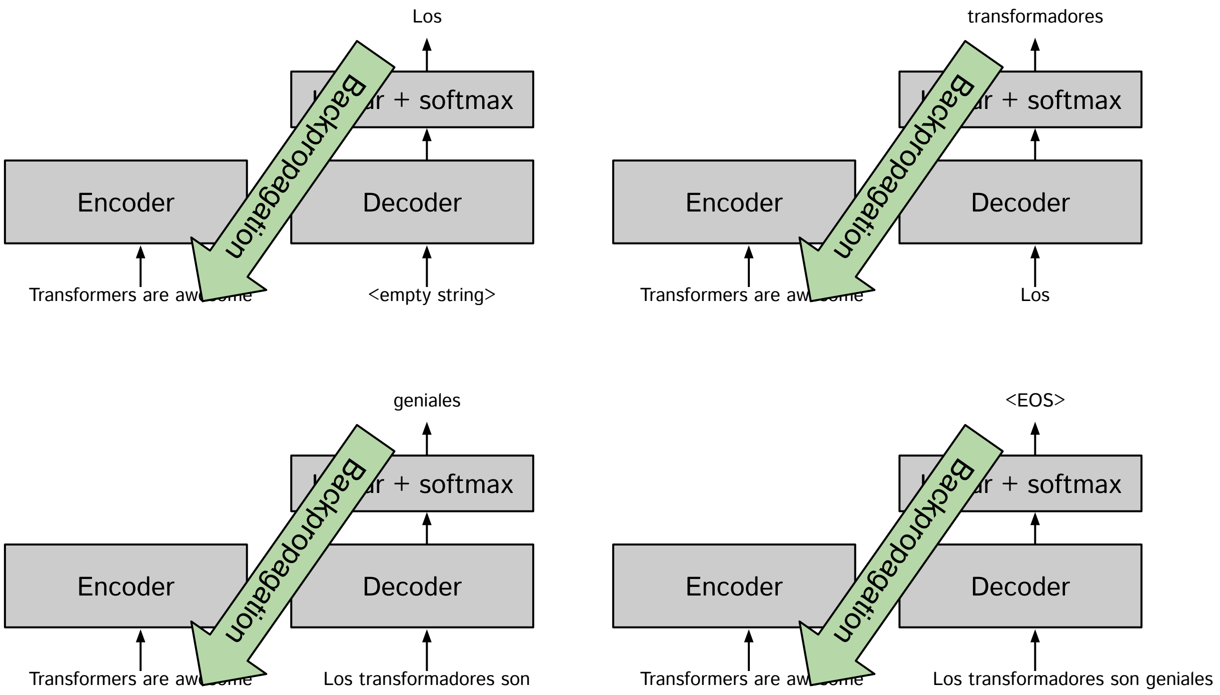
Here we need to mask the right-part of the output, as we cannot learn from what we have not seen yet!

How is a transformer queried?



Autoregressive model:
- *ML/greedy search*
- *top-k probability simulation*
- *beam search*

How is a transformer trained?



In the decoder self-attention modules, tokens cannot attend to subsequent tokens in the output!

Solution: masked self-attention!

Masked attention

For a given token, say the i-th,

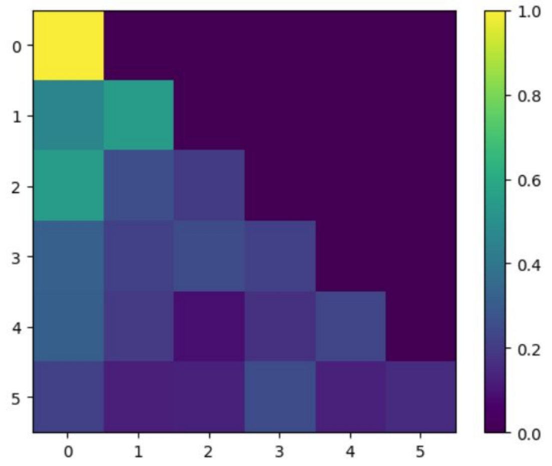
– self-attention output should be nullified in correspondence of all subsequent tokens,

– this is accomplished by forcing the softmax argument to be -∞…

– …so that scores nullify.

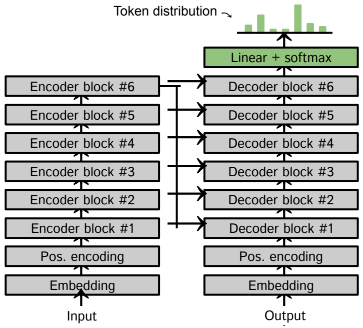
$$X^{\text{att},k} = \text{softmax}\left(\frac{W_Q X \cdot (W_K X)^\top}{\sqrt{d}} + M\right) \cdot (W_V X)$$

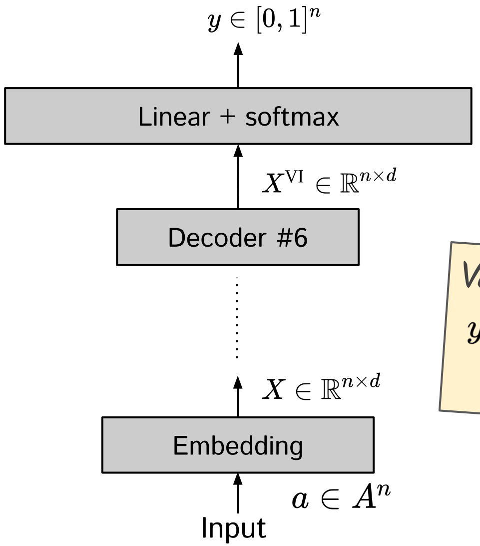
$$m_{ij} = \begin{cases} 0 & \text{if } i \leq j \\ -\infty & \text{otherwise} \end{cases}$$



Output blocks

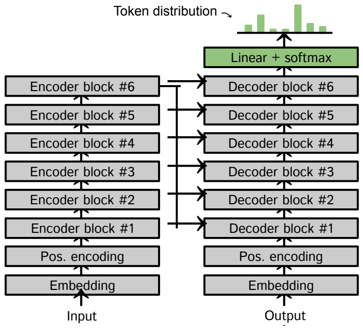
– Purpose: predicting the probability of next tokens.
– A single linear layer projecting the output of last decoder to a (much bigger) vector, having one position for each possible output token…
– …with softmax activation.





Variation:
$y \in [0,1]^m, m \neq n$

Generating outputs



- The final softmax layer output can be interpreted as a probability distribution over tokens.
- Easiest way to go: ML output (i.e., token with highest probability).
- Alternative: sample from the distribution (effect: nondeterministic output).

Final remark: all components only use linear algebra operations, possibly followed by nonlinear functions. Thus transformers are, in the end, highly-structured neural networks.

Training of transformer-based models

- Backpropagation (cross-entropy loss), possibly with GPUs/TPUs.
- Hyperparameter selection: optimizer, learning rate, dropout, and so on.
- Self-supervised training, using, e.g., Wikipedia, or other (big) corpora.
- The Web is full of textual sequences that a transformer can learn to reproduce.
- Refined technique
  1. fix beam size s, run the model for tokens with top-s probability,
  2. select the output with smallest cross-entropy and proceed with training.

Training of transformer-based models

- Fine-tuning (aka few-shots learning):
  1. train on large corpora via self-supervision,
  2. train on a small dataset with «classical» supervision (transfer learning to the specific problem under study).
- Note that zero-shot learning, aka «prompting», is sometimes also possible.

You

Complete this sentence: "Transformers are..."

ChatGPT

Transformers are a type of deep learning model architecture that revolutionized natural language processing and various other tasks by introducing the innovative self-attention mechanism, allowing for effective capturing of long-range dependencies in sequential data.

The art of prompting

Table 3 (excerpt) from Liu et al., 2023 [8]

Type	Task	Input ([X])	Template	Answer ([Z])
Text CLS	Sentiment	I love this movie.	[X] The movie is [Z].	great fantastic ...
	Topics	He prompted the LM.	[X] The text is about [Z].	sports science ...
	Intention	What is taxi fare to Denver?	[X] The question is about [Z].	quantity city ...
Text-span CLS	Aspect Sentiment	Poor service but good food.	[X] What about service? [Z].	Bad Terrible ...
Text-pair CLS	NLI	[X1]: An old man with ... [X2]: A man walks ...	[X1]? [Z], [X2]	Yes No ...
Tagging	NER	[X1]: Mike went to Paris. [X2]: Paris	[X1] [X2] is a [Z] entity.	organization location ...
Text Generation	Summarization	Las Vegas police ...	[X] TL;DR: [Z]	The victim ... A woman ... ...
	Translation	Je vous aime.	French: [X] English: [Z]	I love you. I fancy you. ...

Example

– English-to-German and English-to-French sentence translation.
– Dataset: WMT 2014 (4.5M (EN-DE) and 36M (EN-FR) pairs).
– Hw: 8 GPUs (NVIDIA P100).
– Train time: 12 hours (base model), 3,5 days (big model).

BLEU (BiLingual Evaluation Understudy) is a common metric used for assessing performances in automated translation.

Table 2 from Vaswani et al., 2017 [1]

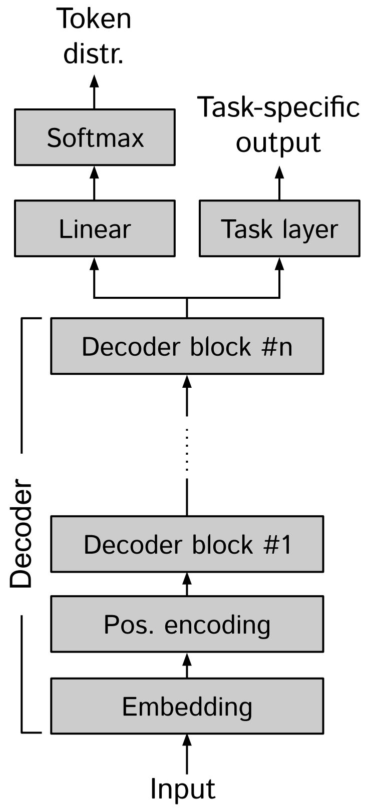
Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Decoder-only architectures

- Based on decoder blocks.
- Proved effective in text generation tasks (e.g., chatbots).
- Task-specific additional output layer (e.g., classification).
- Zero-shot learning (aka prompting) is also an option.

You
"Transformers are awesome." Classify this sentence as "positive", "neutral", or "negative".

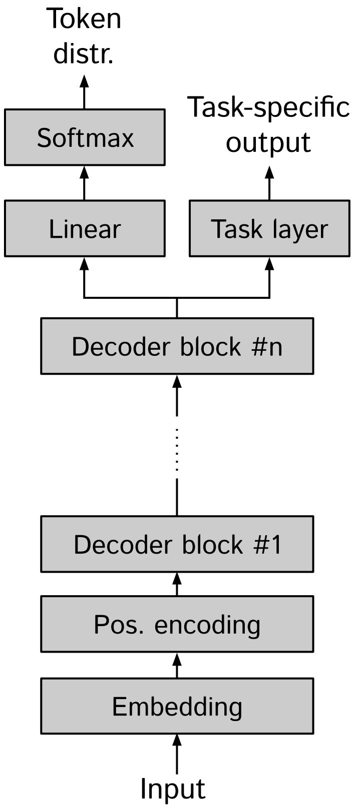
ChatGPT
"Transformers are awesome." is classified as positive.



Most known example: GPT

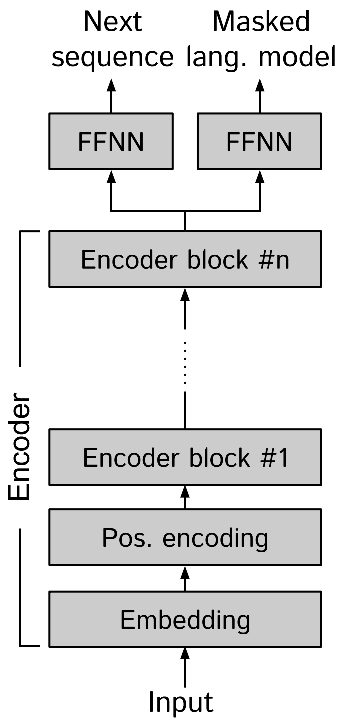
GPT training

– Autoregressive behaviour: given the initial part of a sentence, predict next token.
– Possibly coupled with specific learning tasks (e.g., text classification, similarity, question answering, …).



Encoder-like architectures

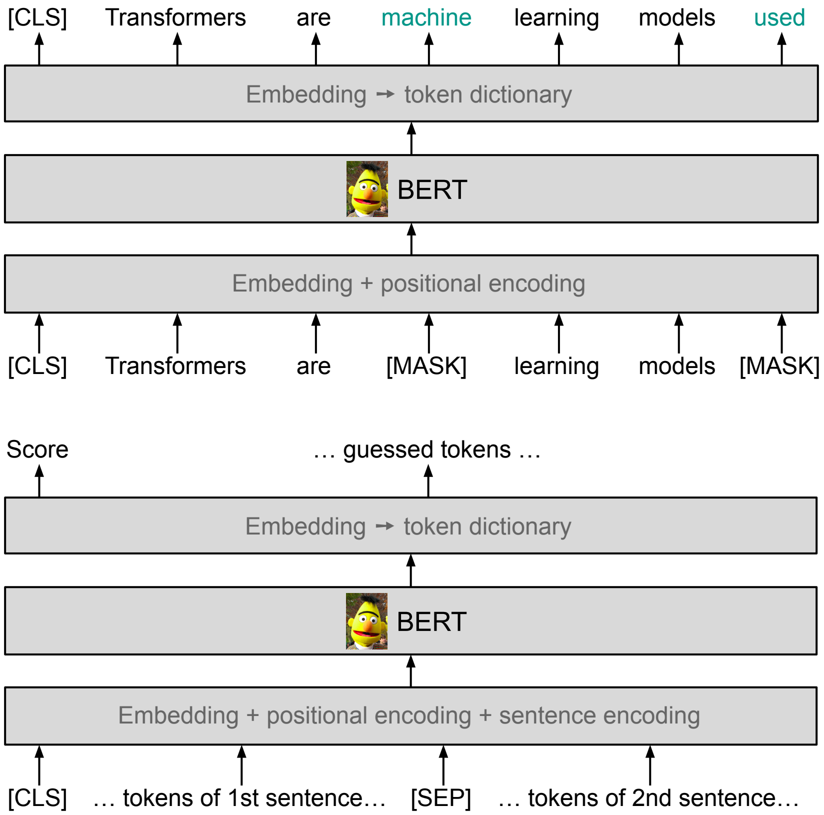
- Based only on encoder blocks.
- Proved effective in learning latent representations (to be used for classification, clustering, and so on).
- Training using the whole sequence.
- Note the completely different output layer.



Most known example: Bert

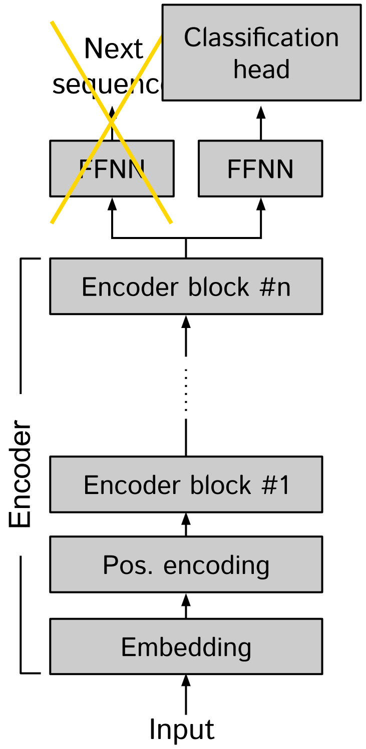
BERT training: MLM & NSP

– Predict tokens masked at random in a sentence (Masked Language Model, MLM).
– Predict if one sentence follows another one (Next Sequence Prediction, NSP).
– NSP subsequently removed in RoBERTa.

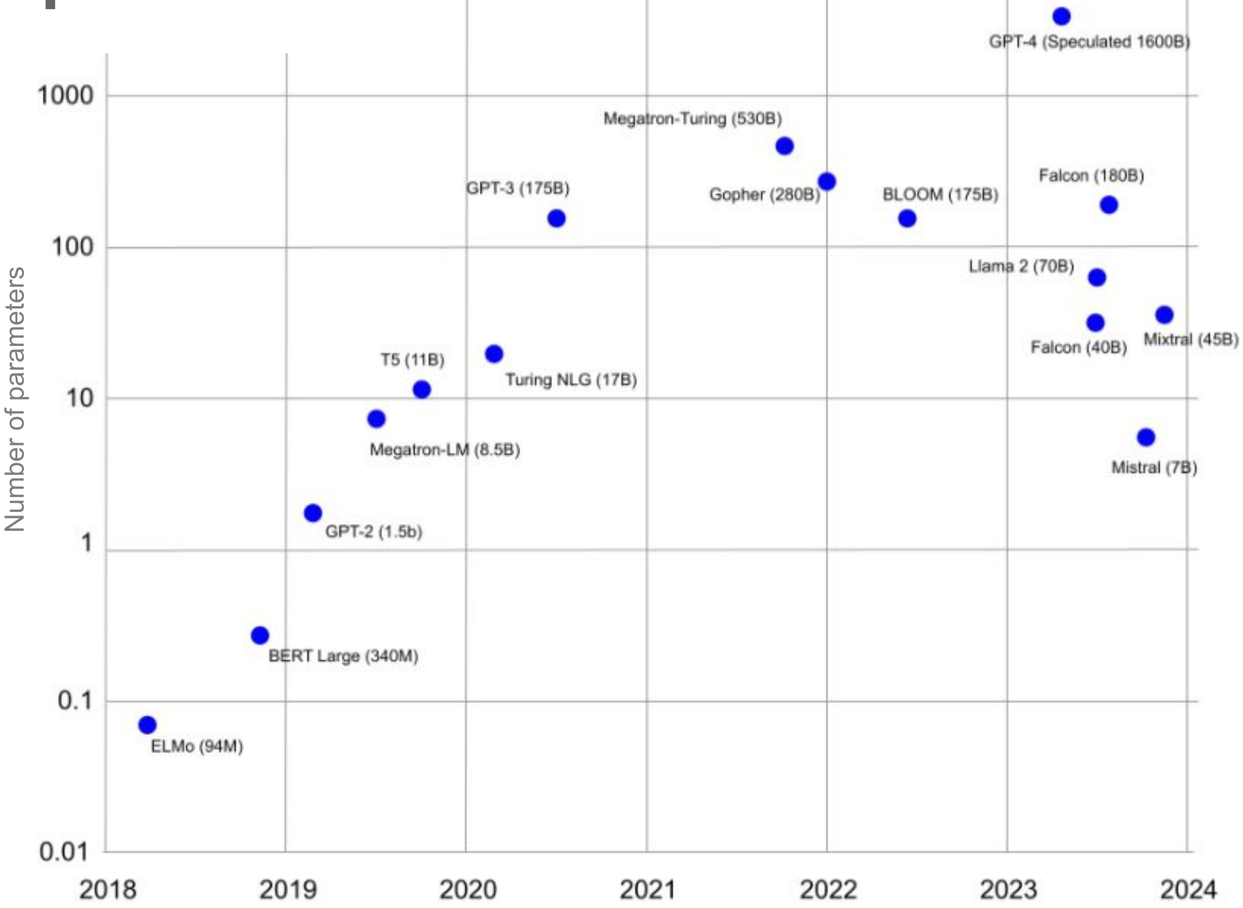


Fine-tuning

- Adapting a pre-trained model to new tasks:
  - few-shots learning,
  - zero-shot learning.
- Head adaptation: adding/modifying the final part of a trained transformer to tackle a different problem:
  - using latent representations as features,
  - through retraining.

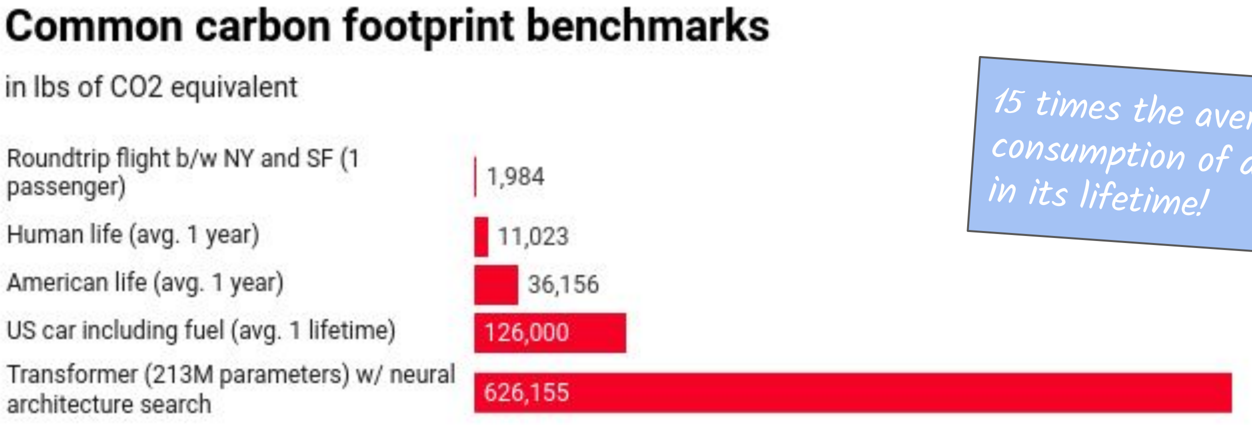


Critical issues: space



Source: Formation FIDLE [5]

Critical issues: energy consumption

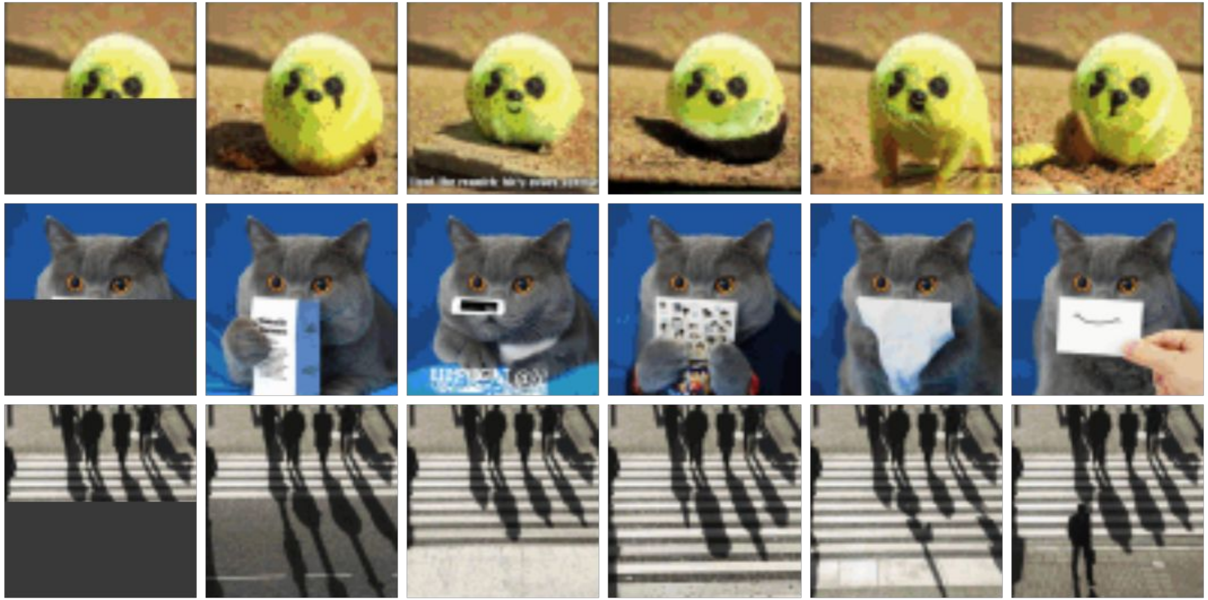


15 times the average consumption of a car in its lifetime!

Source: Strubell et al., 2019 [6]

Transformers for images: ImageGPT

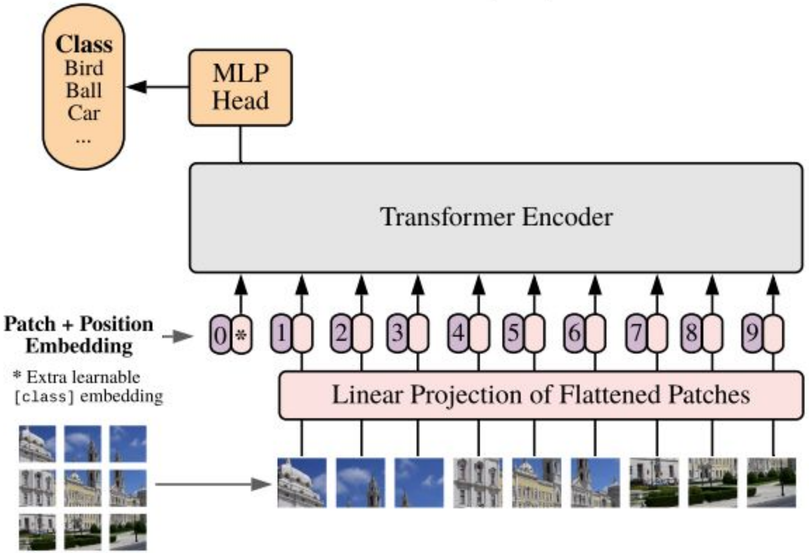
The simplest option: just flatten image as a 1D sequence!



Source: https://openai.com/research/image-gpt

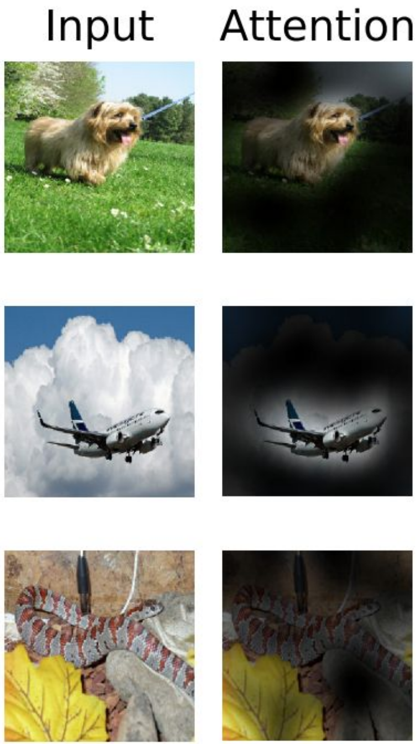
Transformers for images: Visual Transformer

Slightly harder: divide in patches, and flatten them!



Source: Dosovitisky et al, 2021 [7]

Additional feature: explainability via superposition of attention scores on the input image!



From LLMs to Foundation Models

- Transformers have been adapted to a wide range of realms
  - Code production (copilot)
  - Protein generation (proGEN) and structure prediction (Alphafold 2)
  - Chemical reaction prediction
- Foundation models: learn foundational knowledge in a field (e.g., natural language) and adapt in order to solve specific problems (e.g., sentiment analysis)

Materials

[1] A. Vaswani et al., Attention is all you need, "Attention is all you need." Advances in neural information processing systems 30 (2017). https://arxiv.org/abs/1706.03762

[2] E, Muñoz, Attention is all you need: discovering the transformer paper. https://towardsdatascience.com/attention-is-all-you-need-discovering-the-transformer-paper-73e5ff5e0634

[3] P. Bloem, Transformers from scratch. https://peterbloem.nl/blog/transformers

[4] J. Alamar, The Illustrated Transformer. https://jalammar.github.io/illustrated-transformer/

[5] Formation FIDLE, Transformers. https://www.youtube.com/watch?v=L3DGgzIbKz4

[6] Emma Strubell and Ananya Ganesh and Andrew McCallum, Energy and Policy Considerations for Deep Learning in NLP, 2019, https://arxiv.org/abs/1906.02243

[7] A. Dosovitskiy et al., An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, 2021, https://arxiv.org/abs/2010.11929

[8] P. Liu et al., Pre-train, Prompt and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing, 2023, https://doi.org/10.1145/3560815

The lab

- Retrieving pre-trained transformers via 🤗 **Hugging Face** and using them
- Building the basic components of a transformer
- Assemble a transformer
- Query a transformer
- Train a transformer

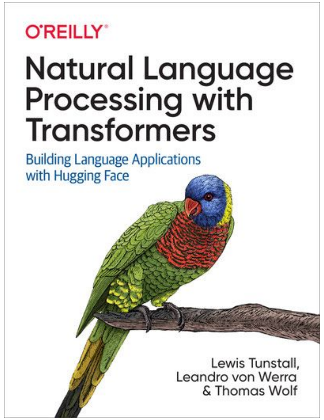
Want to learn more?

SERRANO.ACADEMY
the art of understanding

https://serrano.academy/

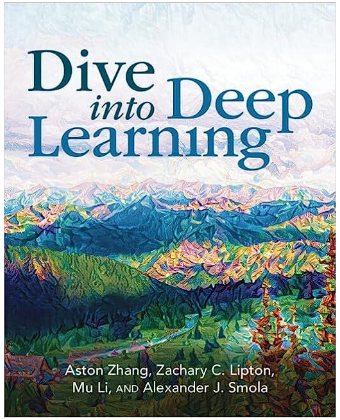
Hugging Face

https://huggingface.co/



https://transformersbook.com/

https://d2l.ai/



Thanks!

dario.malchiodi@unimi.it https://malchiodi.di.unimi.it

Assets:

- Banana juice icons created by Freepik - Flaticon, https://www.flaticon.com/free-icons/banana-juice
- Google fonts and Material icons, https://fonts.google.com
- Font awesome, https://fontawesome.com