Probabilistic Graphical Models & Probabilistic AI

Ben Lengerich

Lecture 19: Attention and Transformers

April 10, 2025

Reading: See course homepage

Today

- Attention
- Self-attention
- Transformers

The Attention Mechansism

Why Attention?

- Consider machine translation:
  - Do we really need the whole sequence to translate each word?
  - Where is **the** library? →
    - Donde esta **la** biblioteca?
  - Where is **the** huge public library? →
    - Donde esta **la** enorme biblioteca publica?
- Problem: RNNs compress all information into a fixed-length vector. Long-range dependencies are tricky.

Hard attention?

- Make a zero-one decision about where to attend.
- Problem: Hard to train. Requires methods such as reinforcement learning

Benefits: Interpretable?

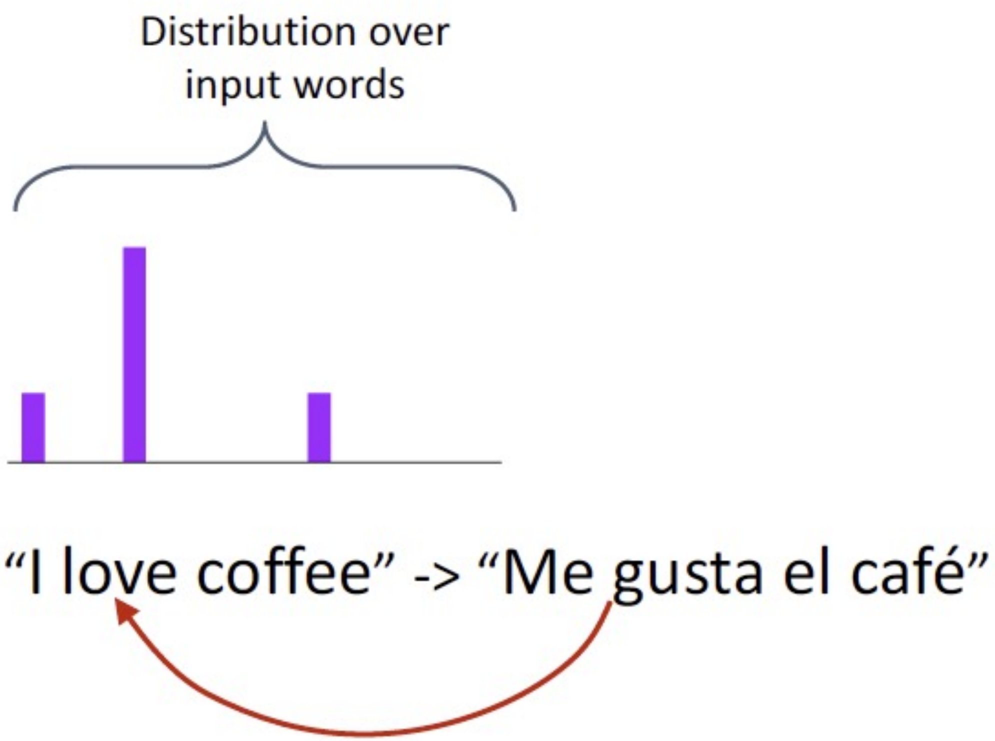
Review

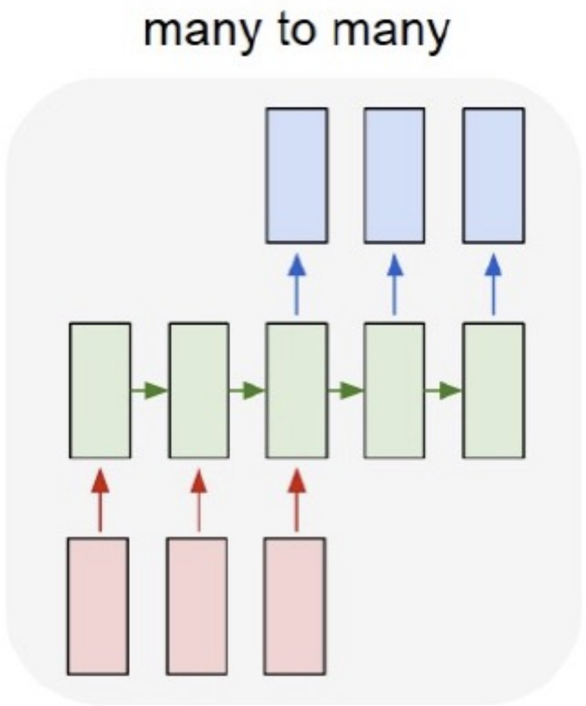
the beer was n't what i expected, and i'm not sure it's "true to style", but i thought it was delicious. **a very pleasant ruby red-amber color** with a relatively brilliant finish, but a limited amount of carbonation, from the look of it. aroma is what i think an amber ale should be - a nice blend of caramel and happiness bound together.

Ratings *Look:* 5 stars *Smell:* 4 stars

Lei et al 2016

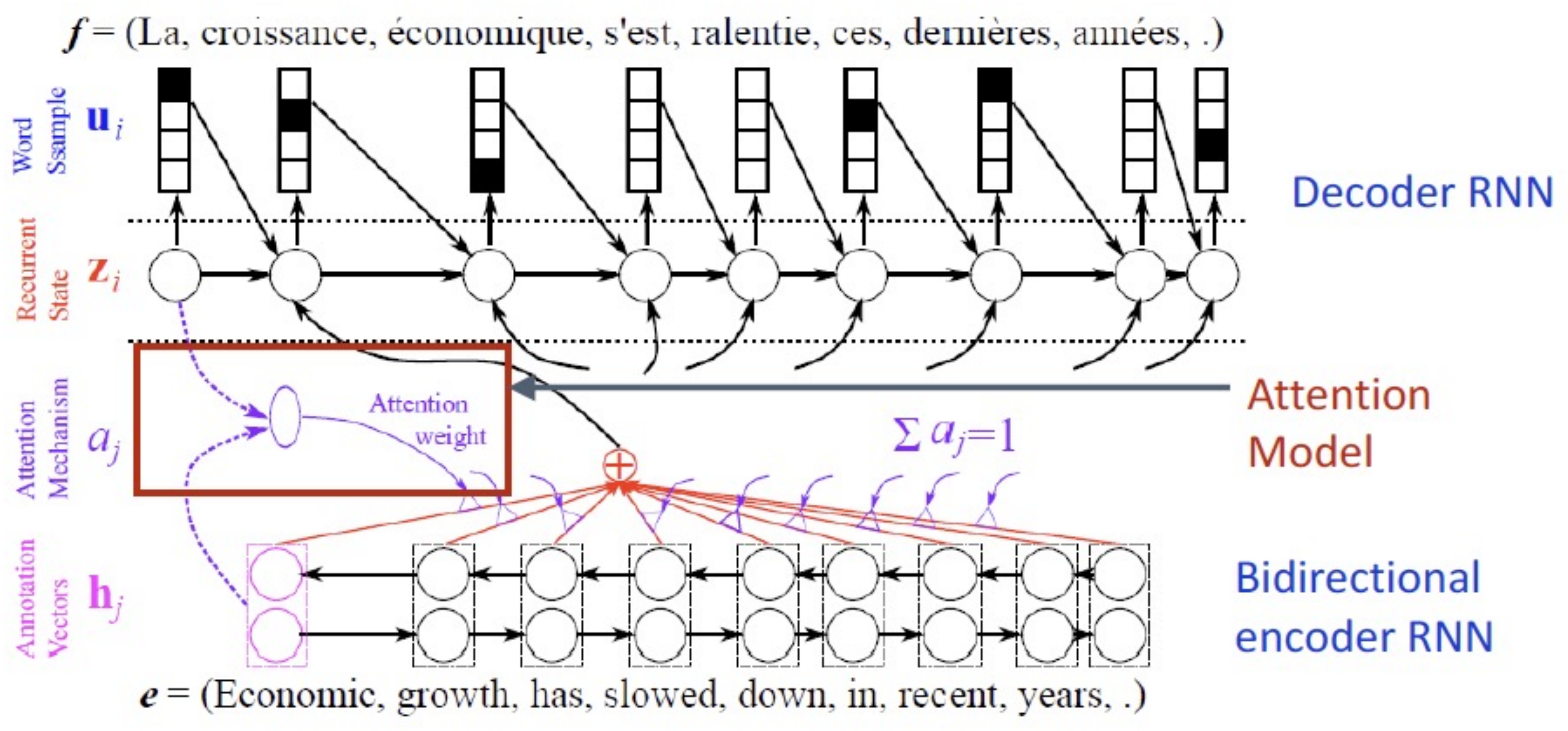
Soft attention





Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Soft attention



From Y. Bengio CVPR 2015 Tutorial

Soft attention

Context vector (input to decoder):

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

Mixture weights:

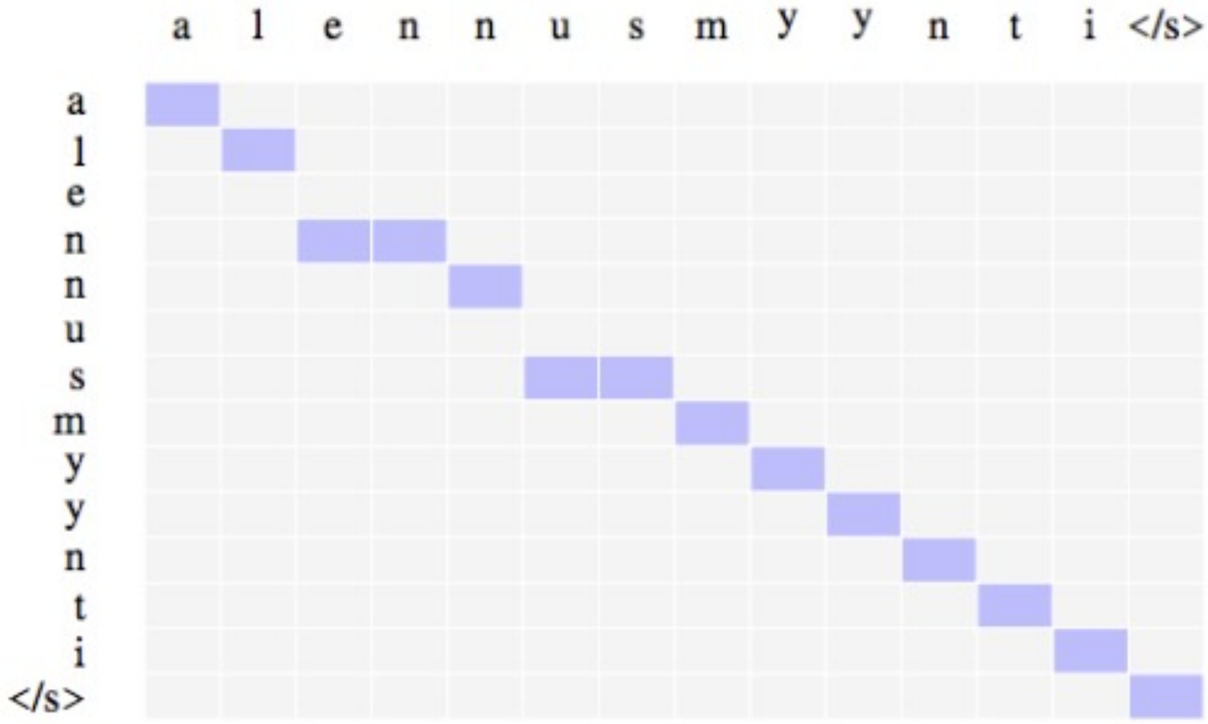
$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

Alignment score (how well do input words near j match output words at position i):

$$e_{ij} = a(s_{i-1}, h_j)$$

Monotonic attention

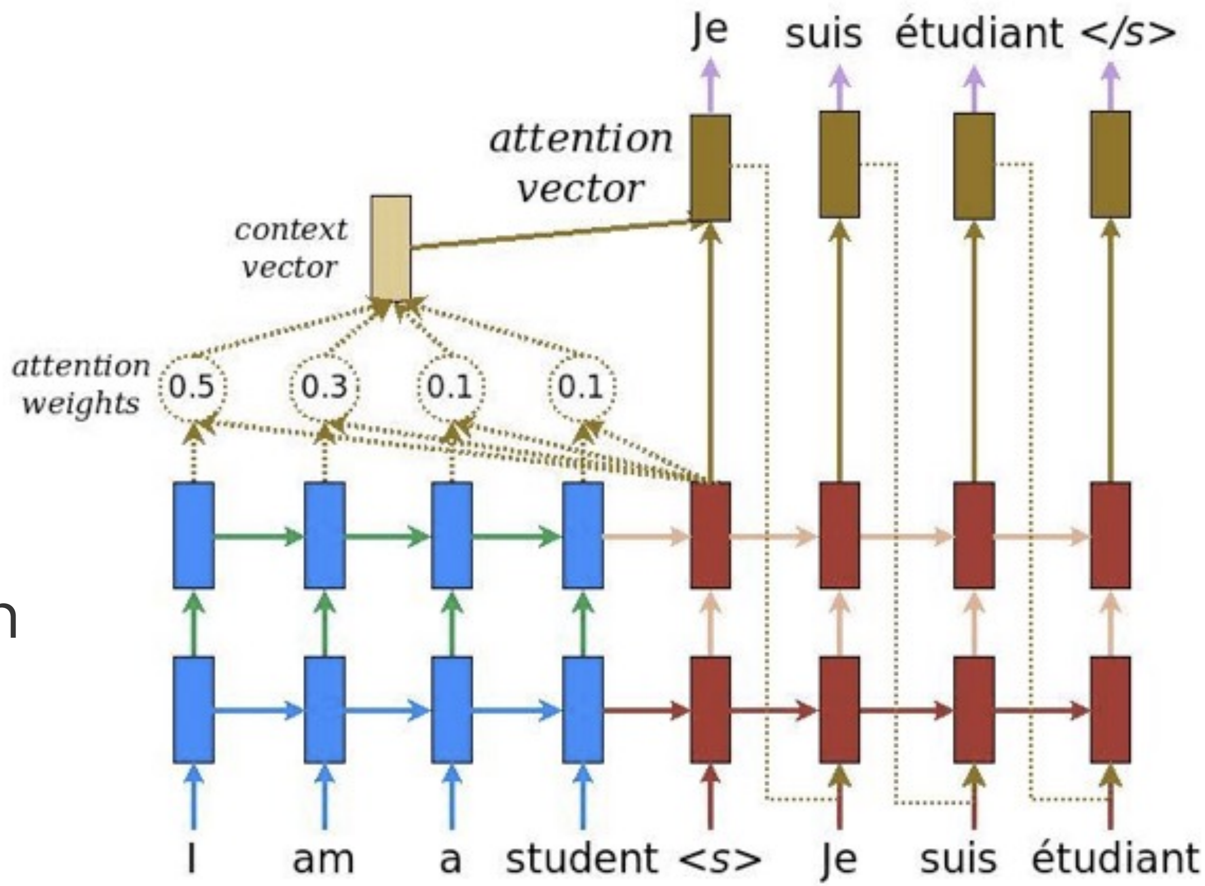
- In some cases, we might know the output will be the same order as the input
  - Speech recognition, incremental translation, morphological inflection (?), summarization (?)



- **Basic idea:** hard decisions about whether to read more

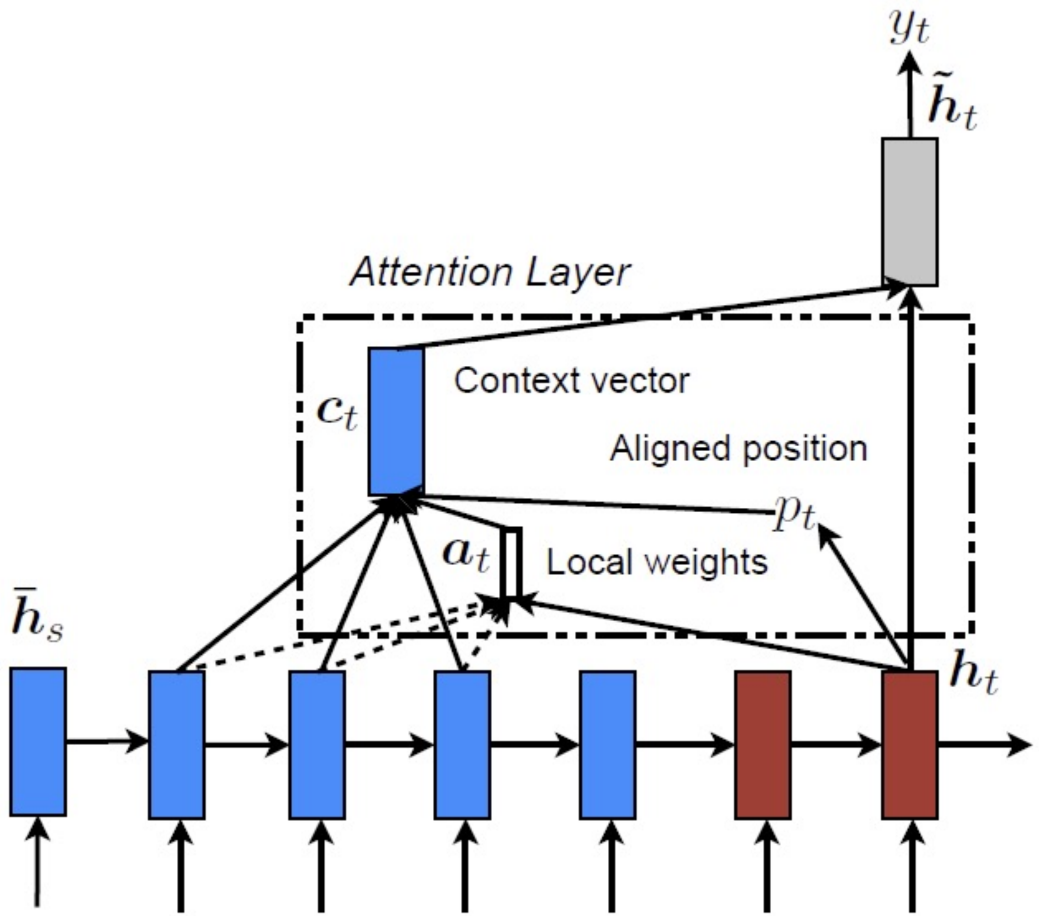
Global Attention

- Attend to a context vector
- Decoder captures global information, not just the information from one hidden state.
- Context vector takes all cells' outputs as input and computes a probability distribution for each token the decoder wants to generate.

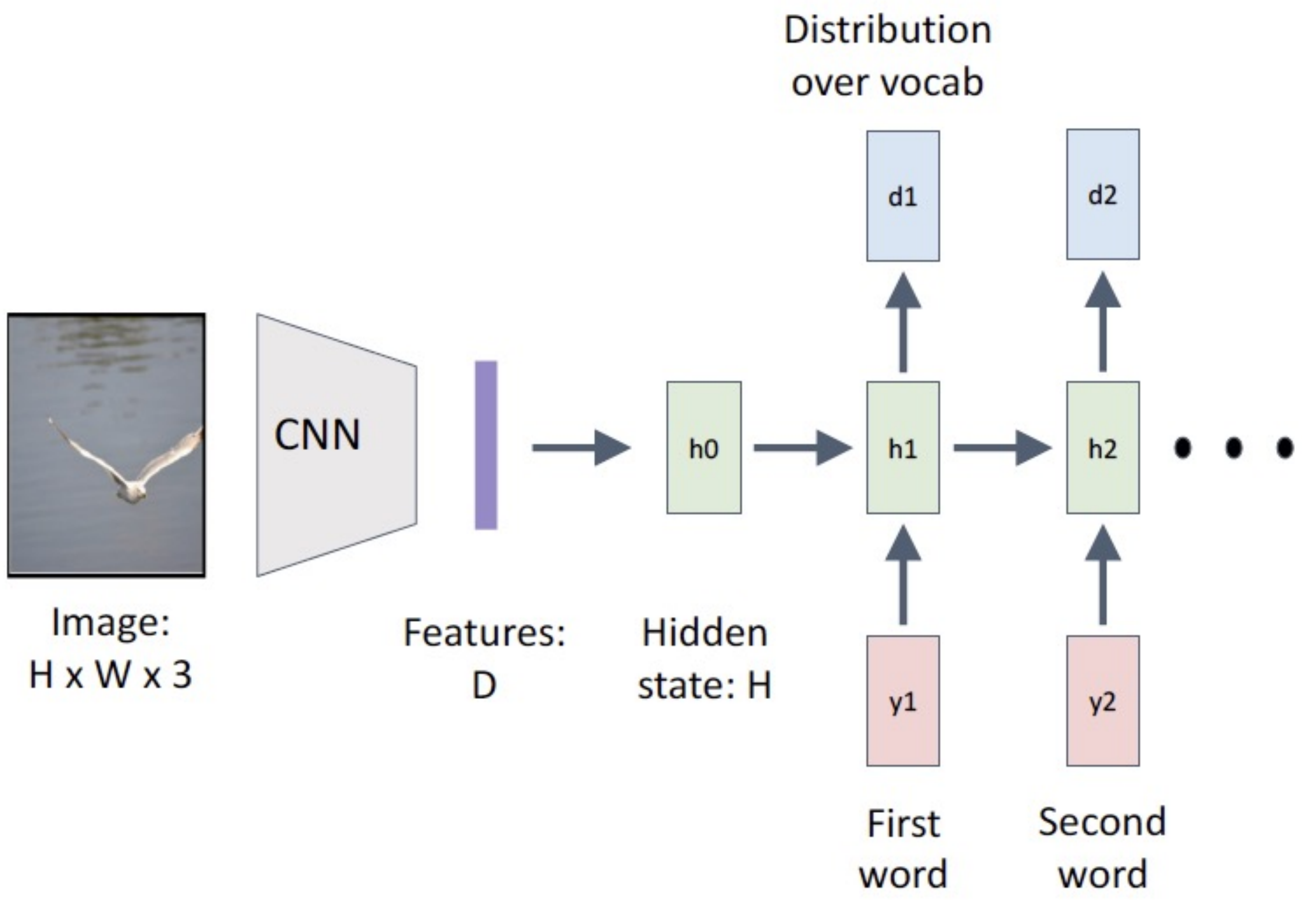


Local Attention

- Compute a best aligned position first
- Then compute a context vector centered at that position

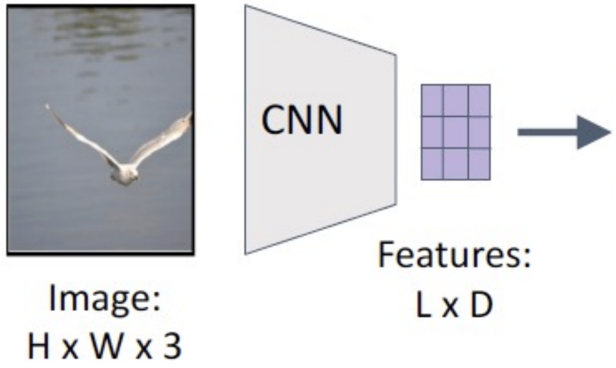


Example: RNN for Image Captioning



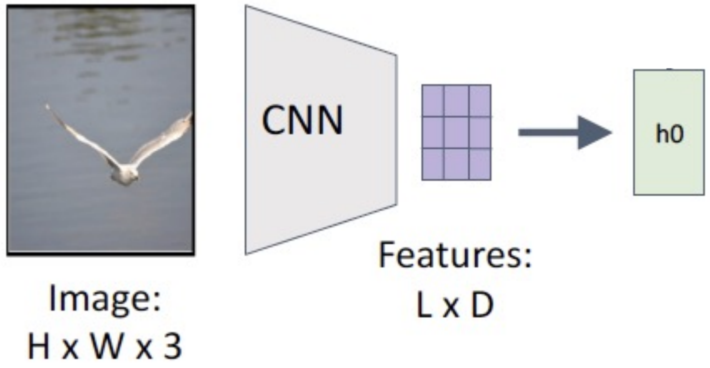
RNN only looks at whole image once…but different parts of the image are important for different parts of the caption.

Example: Soft Attention for Image Captioning



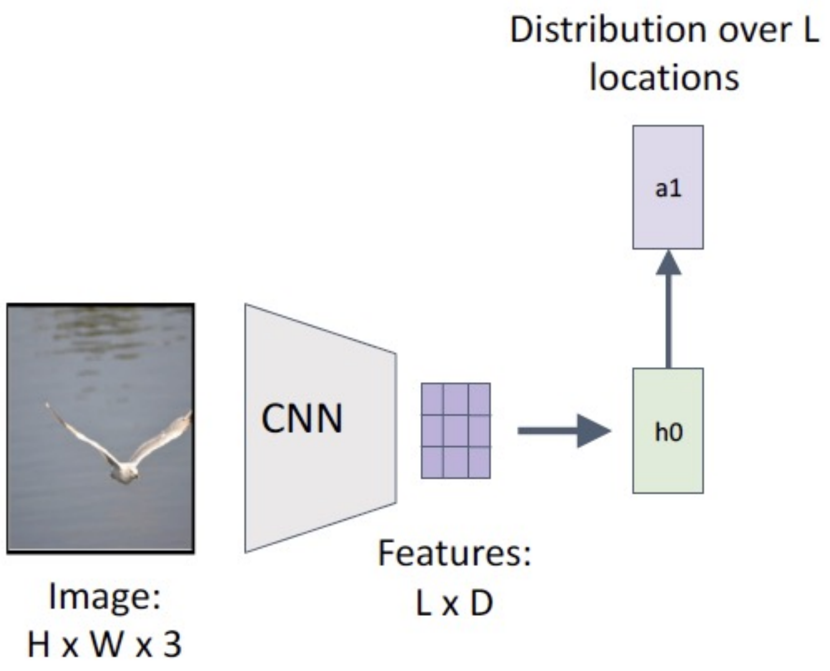
Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Example: Soft Attention for Image Captioning



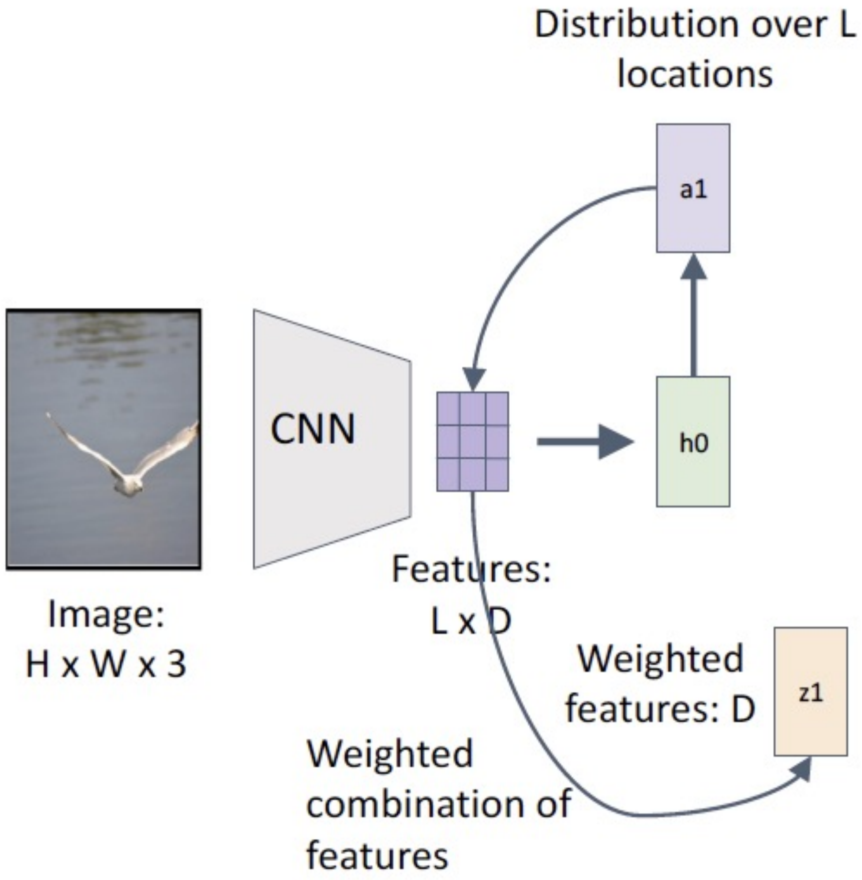
Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Example: Soft Attention for Image Captioning

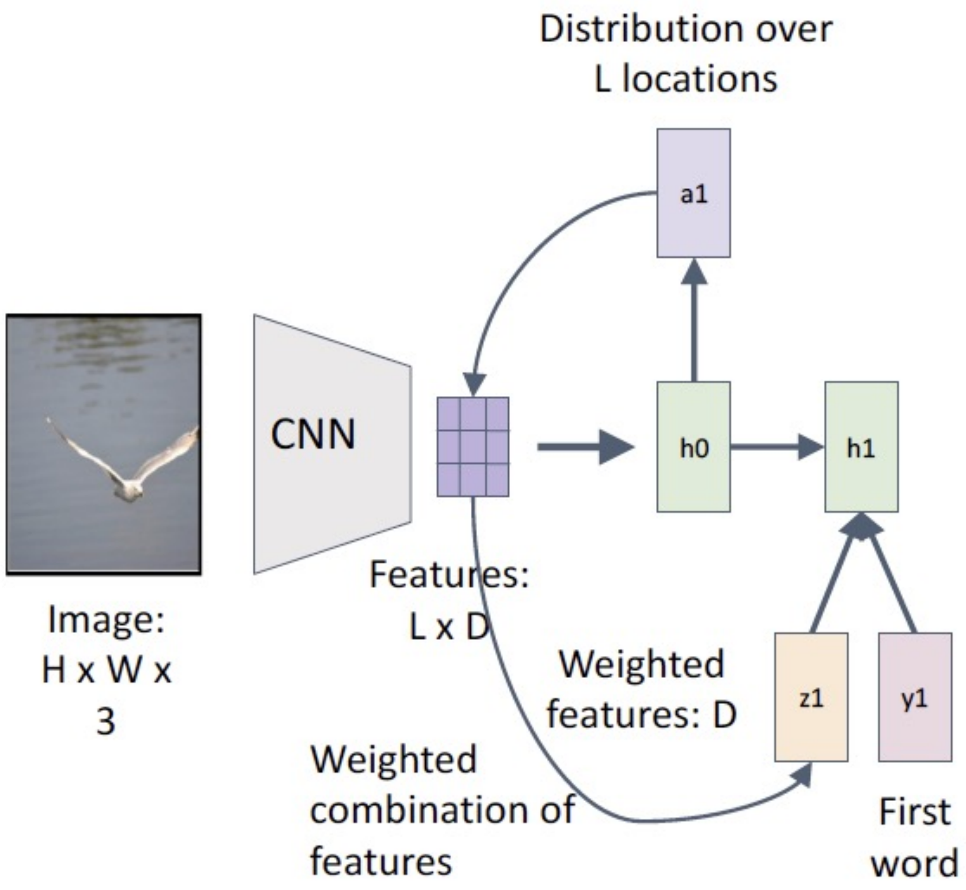


Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

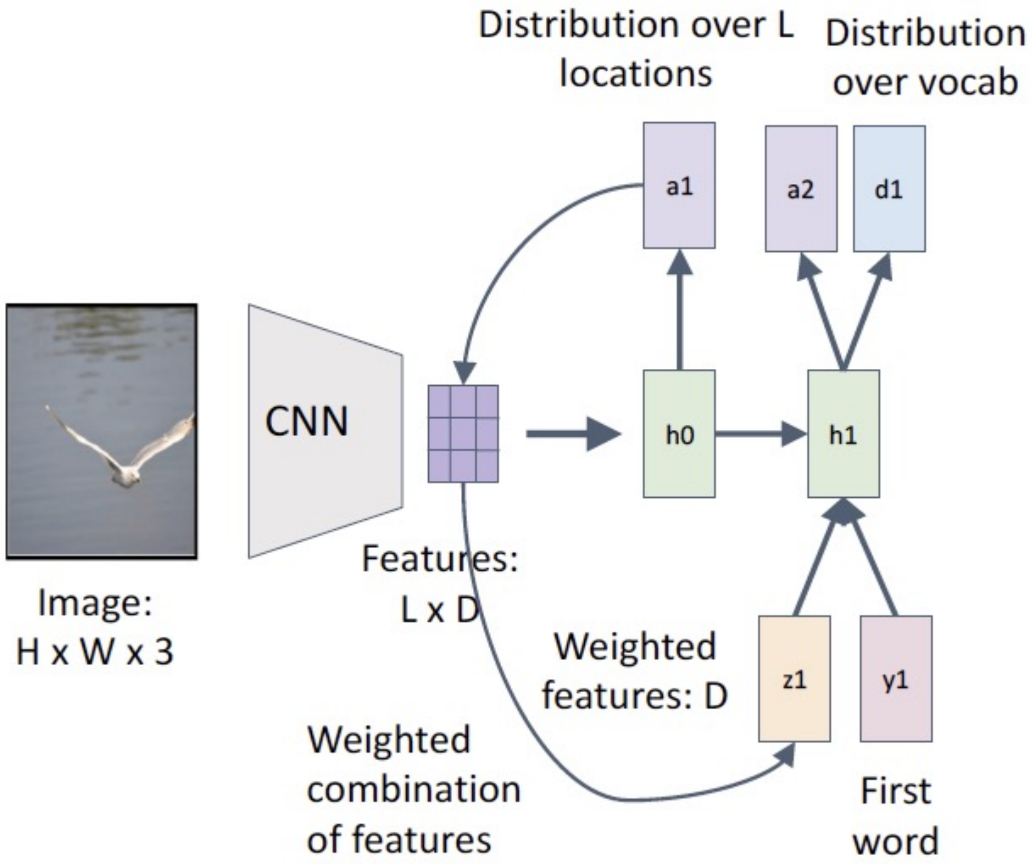
Example: Soft Attention for Image Captioning



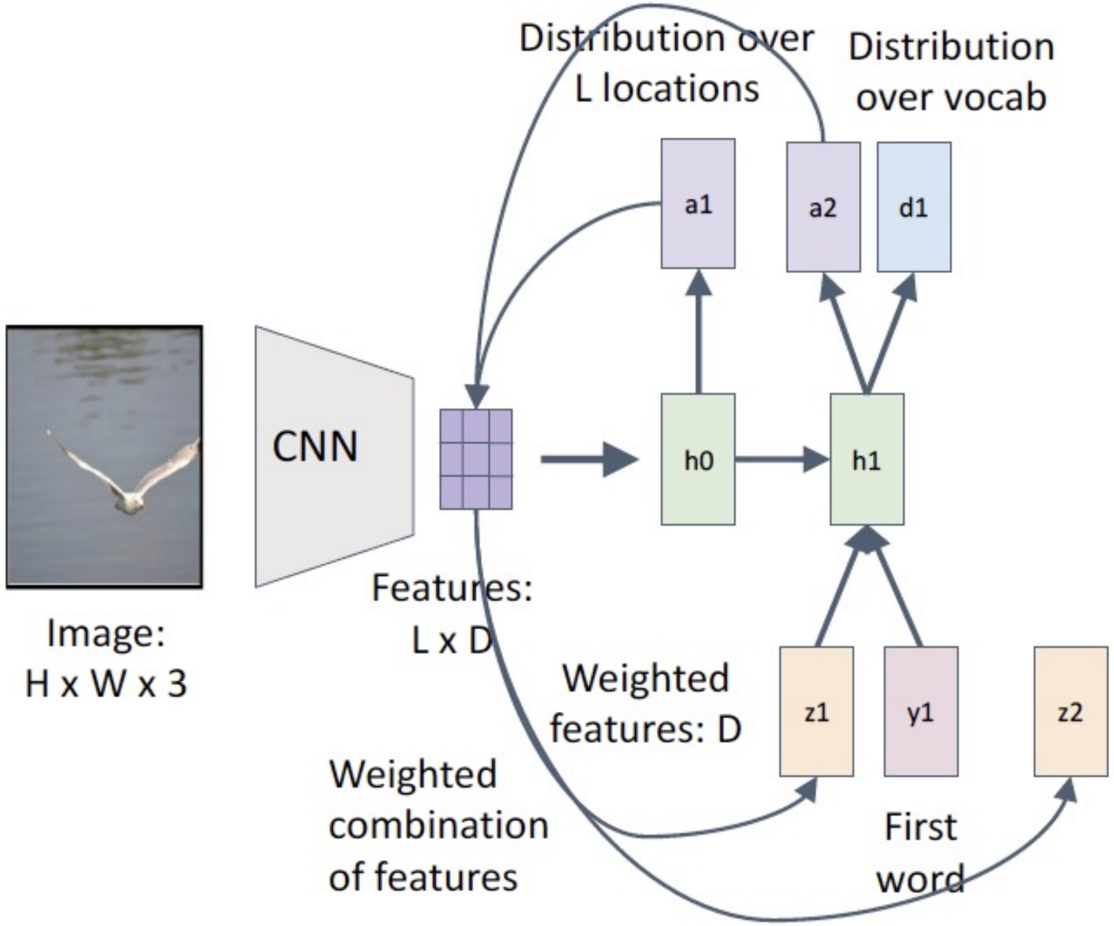
Example: Soft Attention for Image Captioning



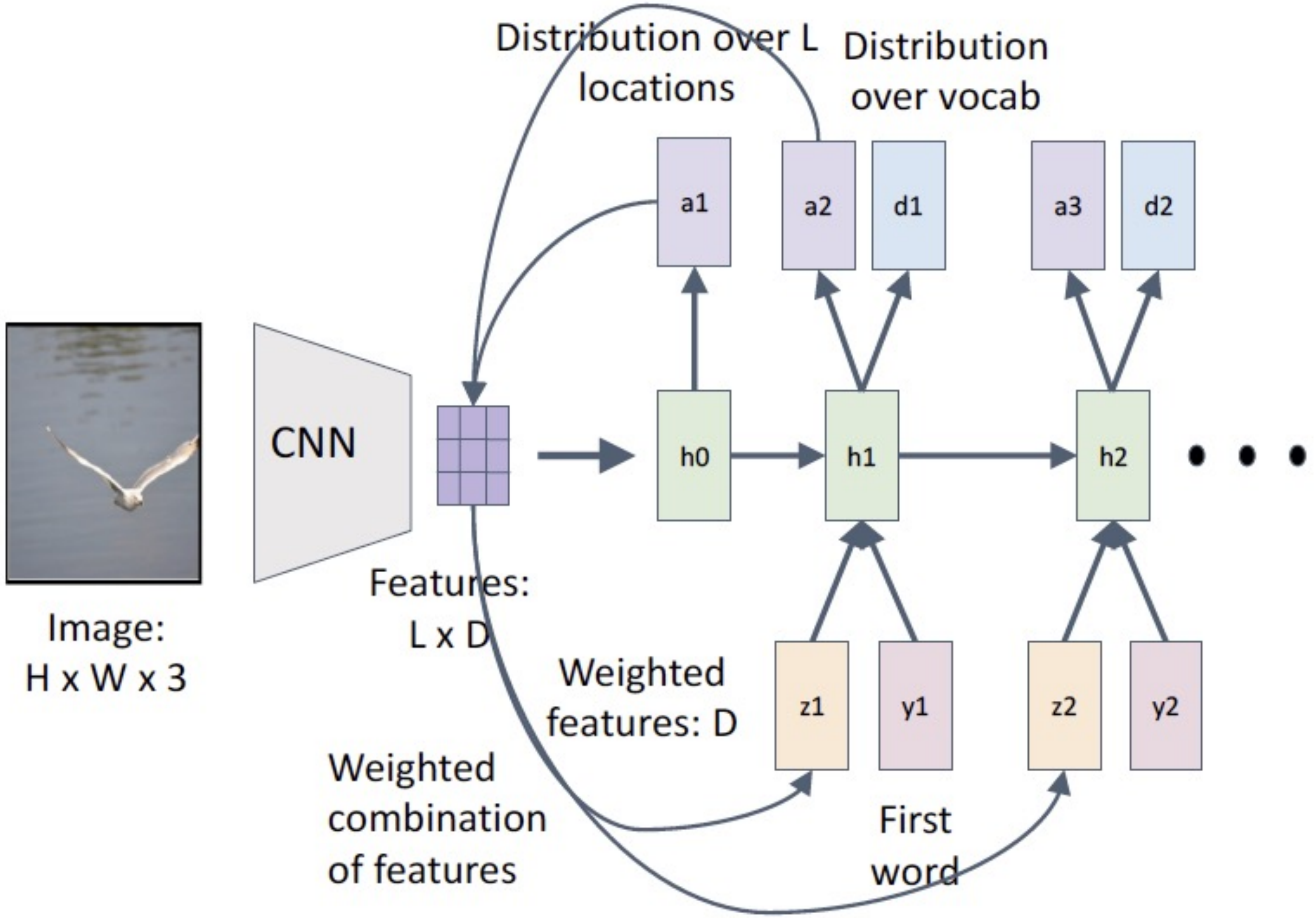
Example: Soft Attention for Image Captioning



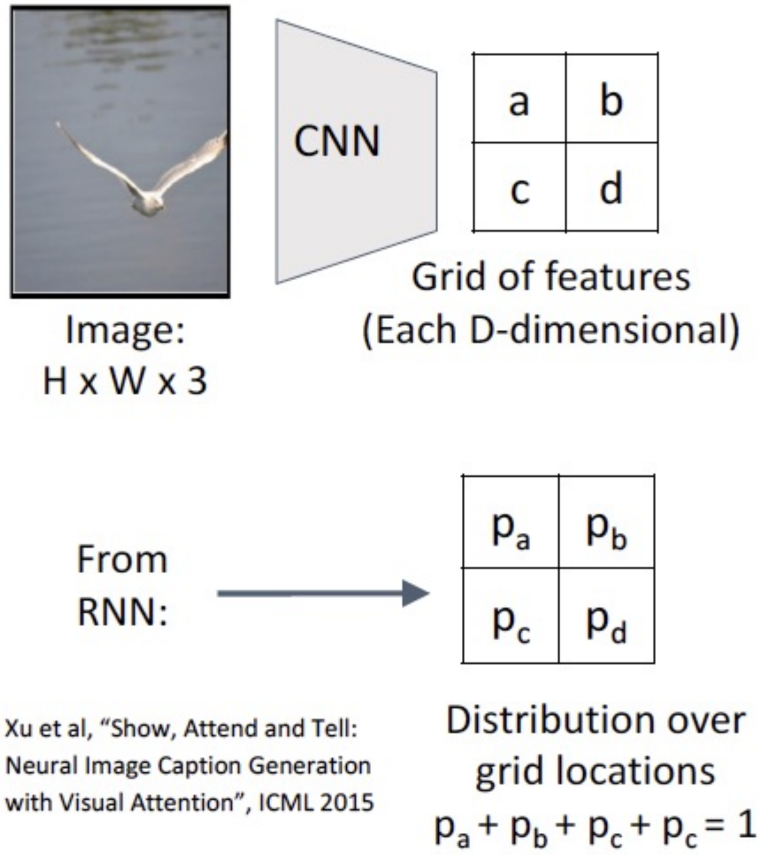
Example: Soft Attention for Image Captioning



Example: Soft Attention for Image Captioning

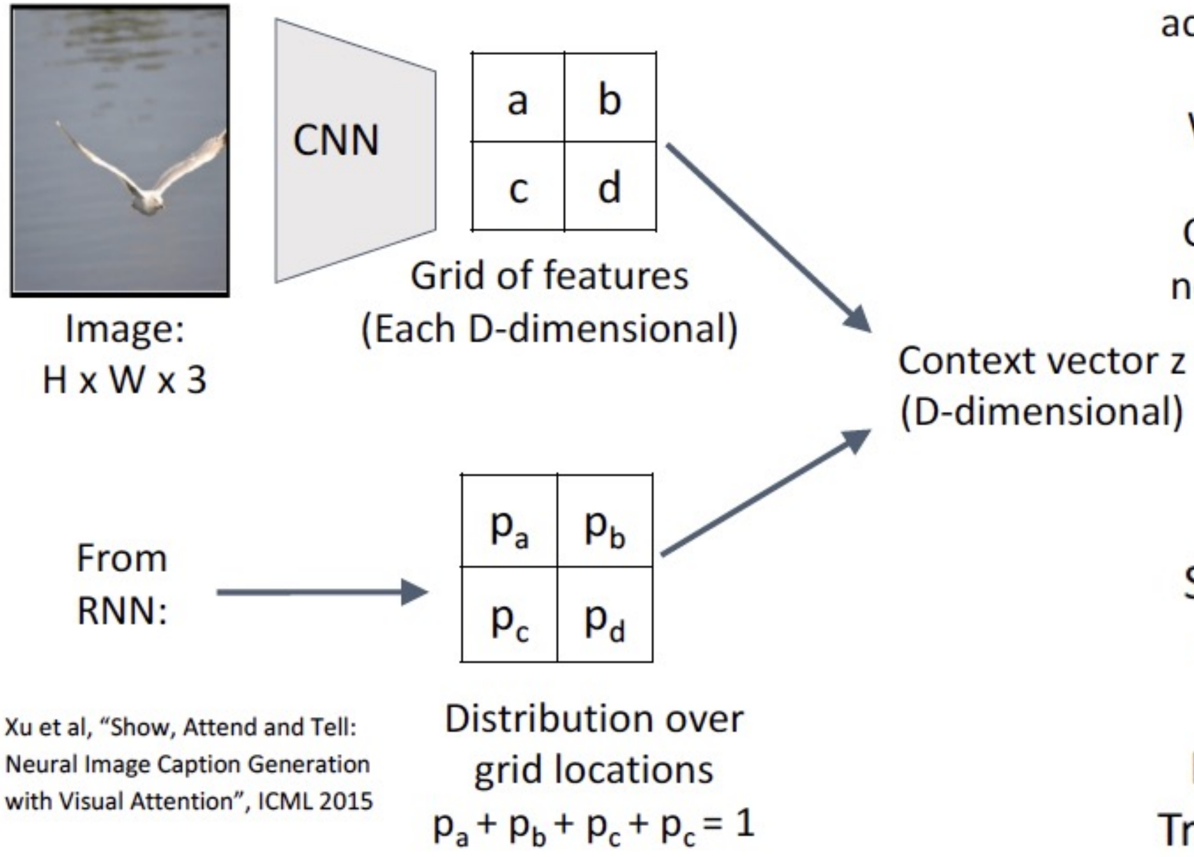


CNNs were an example of hard attention



Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

CNNs were an example of hard attention



Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Hard attention:
Sample ONE location
according to p, z = that vector

With argmax, dz/dp is zero
almost everywhere …
Can't use gradient descent;
need reinforcement learning

Soft attention:
Summarize ALL locations
$z = p_a a + p_b b + p_c c + p_d d$

Derivative dz/dp is nice!
Train with gradient descent

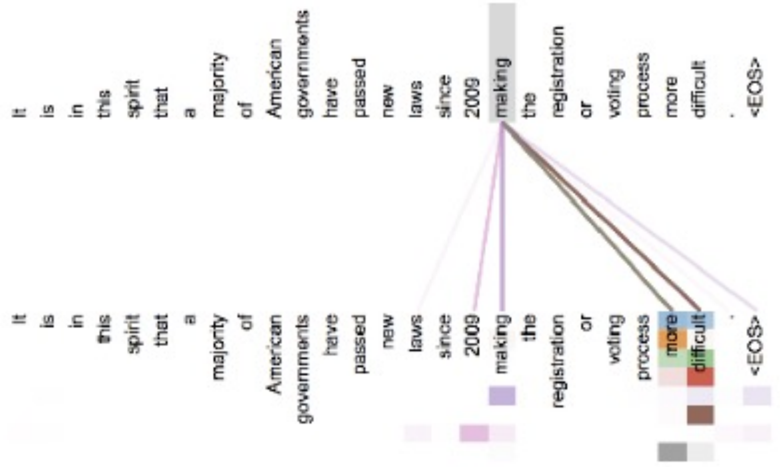
Multi-headed Attention

- **Idea:** multiple attention "heads" focus on different parts of the sentence

- e.g. Different heads for "copy" vs regular (Allamanis et al. 2016)

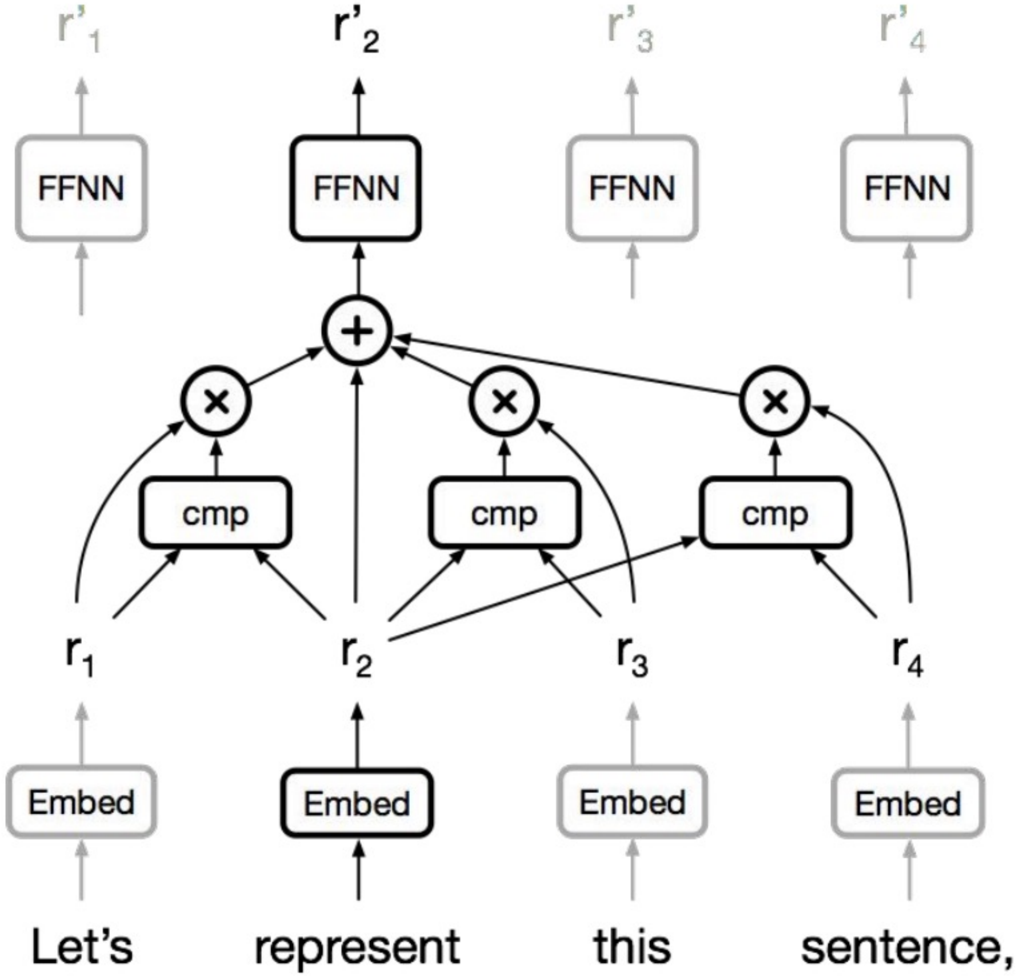
Target			Attention Vectors	λ
m_1	set	α =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.012
		κ =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_2	use	α =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.974
		κ =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_3	browser	α =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.969
		κ =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_4	cache	α =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.583
		κ =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_5	END	α =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.066
		κ =	<s> { this . use Browser Cache = use Browser Cache ; } </s>	

- Or multiple independently learned heads (Vaswani et al. 2017)



Self-Attention

Self-Attention



Vasvani "Self-Attention for Generative Models"

Self-Attention

Constant ‘path length’ between any two positions.

Gating/multiplicative interactions.

Trivial to parallelize (per layer).

Can replace sequential computation entirely?

Vasvani [“Self-Attention for Generative Models”]

Attention is cheap

	FLOPs
Self-Attention	$O(\text{length}^2 \cdot \text{dim})$
RNN (LSTM)	$O(\text{length} \cdot \text{dim}^2)$
Convolution	$O(\text{length} \cdot \text{dim}^2 \cdot \text{kernel\_width})$

Vasvani "Self-Attention for Generative Models"

The Transformer

The "Transformer"

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

Attention is **all you need**

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

… to attend to **all** positions in the decoder up to and including that position. **We need** to prevent … **We** implement this inside of scaled dot-product **attention** by masking out (setting to −∞) …

☆ Save 99 Cite Cited by 174852 Related articles All 73 versions »

The "Transformer"

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

Attention is **all you need**
A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc
… to attend to **all** positions in the decoder up to and including that position. **We need** to prevent … **We** implement this inside of scaled dot-product **attention** by masking out (setting to −∞) …
☆ Save 99 Cite Cited by 174852 Related articles All 73 versions

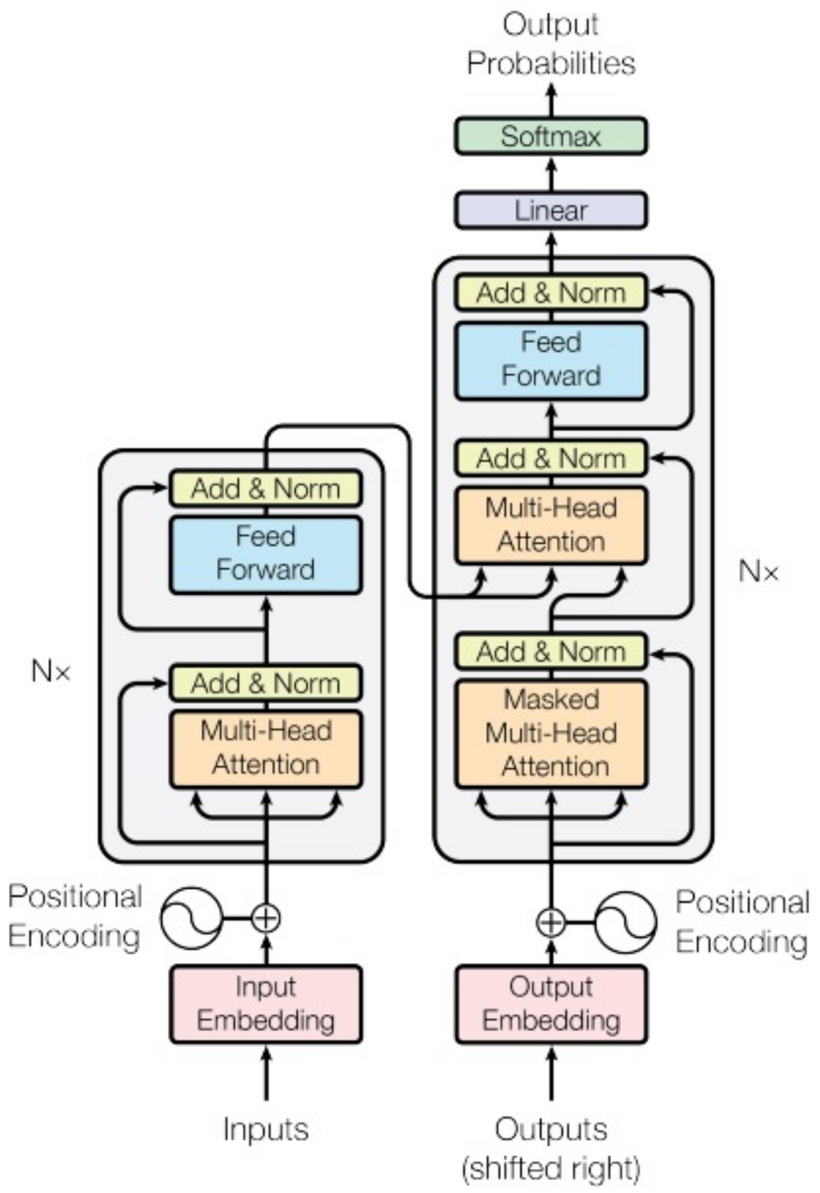
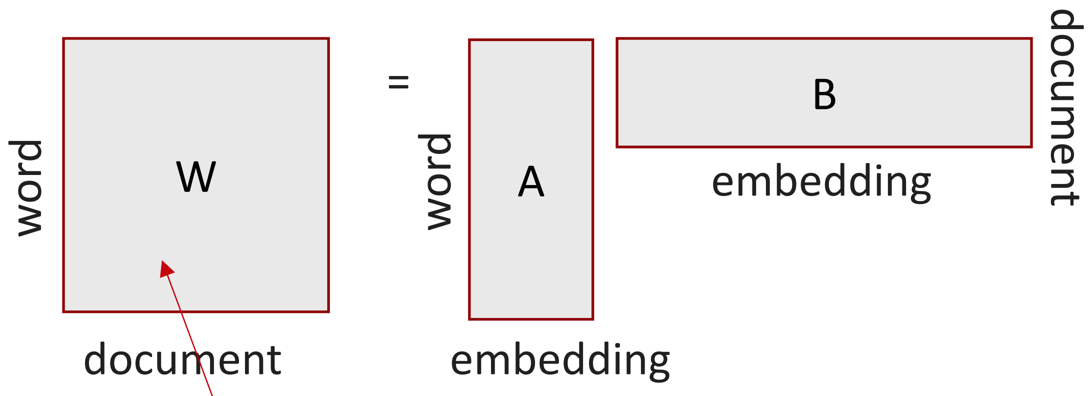


Figure 1: The Transformer - model architecture.

Start with word embeddings…

- Lookup table that translates words (or more formally "tokens") into continuous-valued "embeddings"
- Simplest form: random embeddings
- Slightly better: TF-IDF embeddings
- Many ways to improve pre-trained embeddings

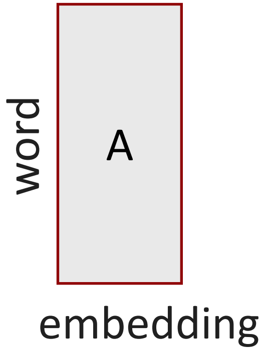


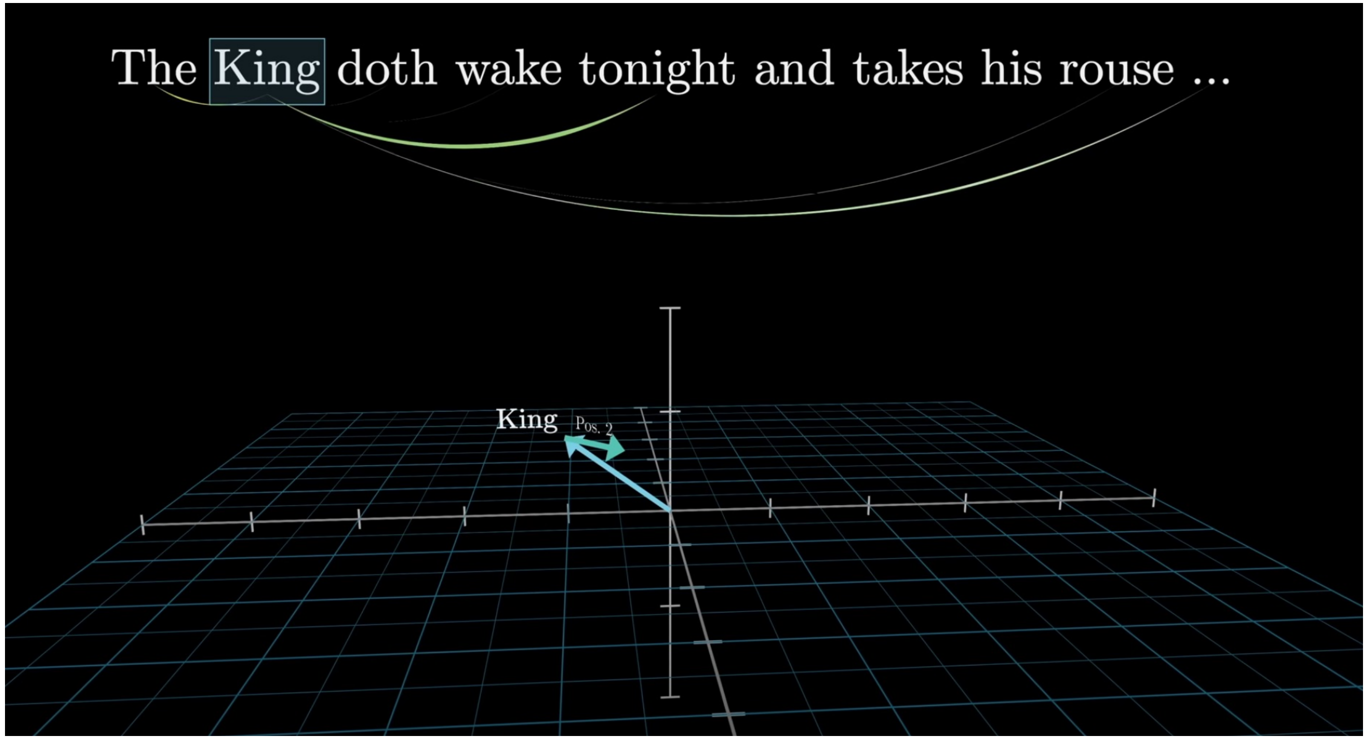
$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF
Term x within document y

$tf_{x,y}$ = frequency of x in y
df_x = number of documents containing x
N = total number of documents

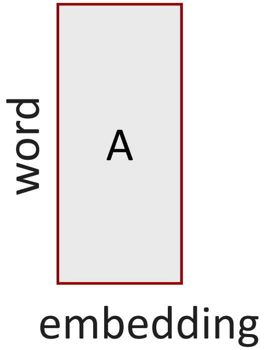
Start with word embeddings...

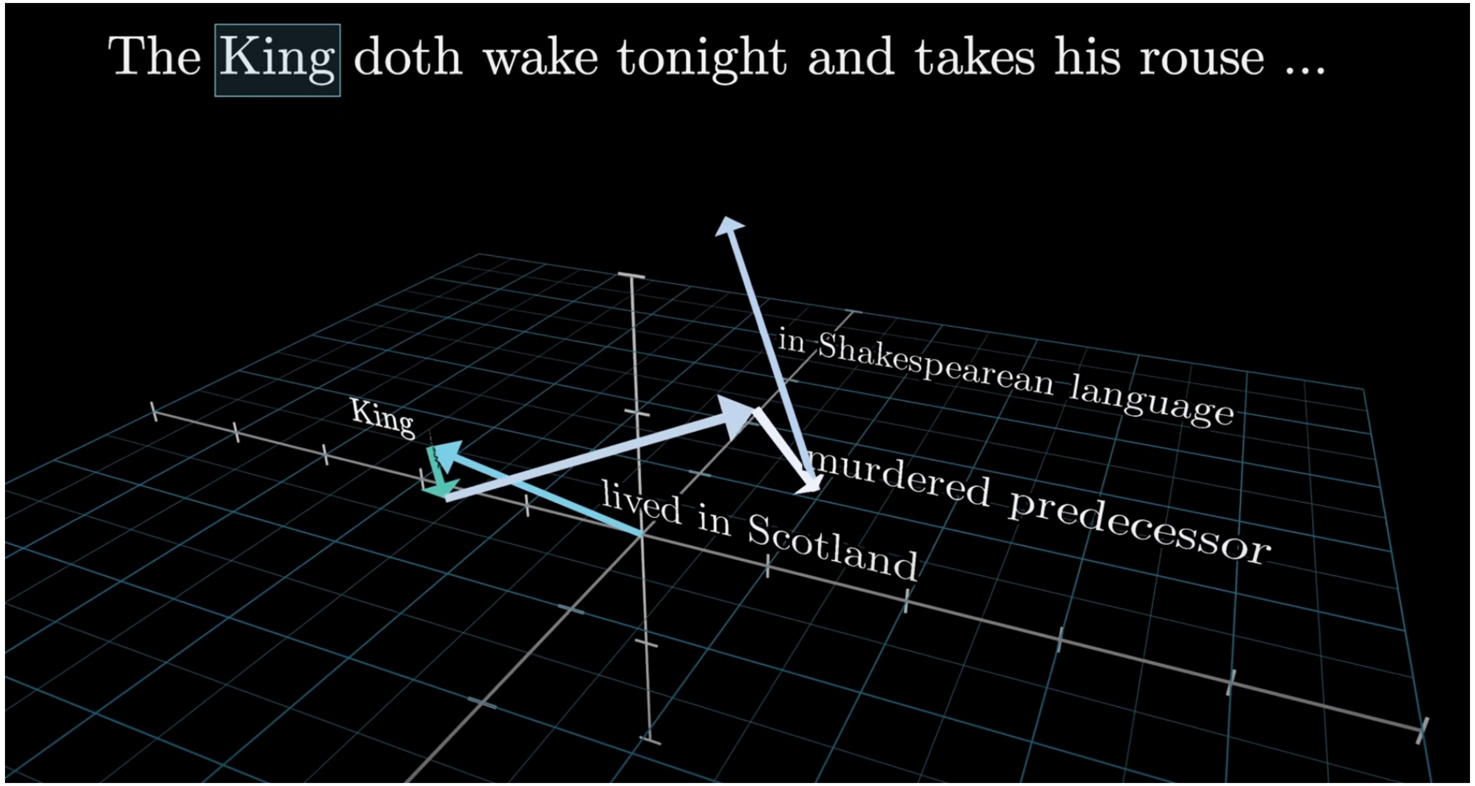




3Blue1Brown [“Attention in Transformers”]

Update embeddings by context

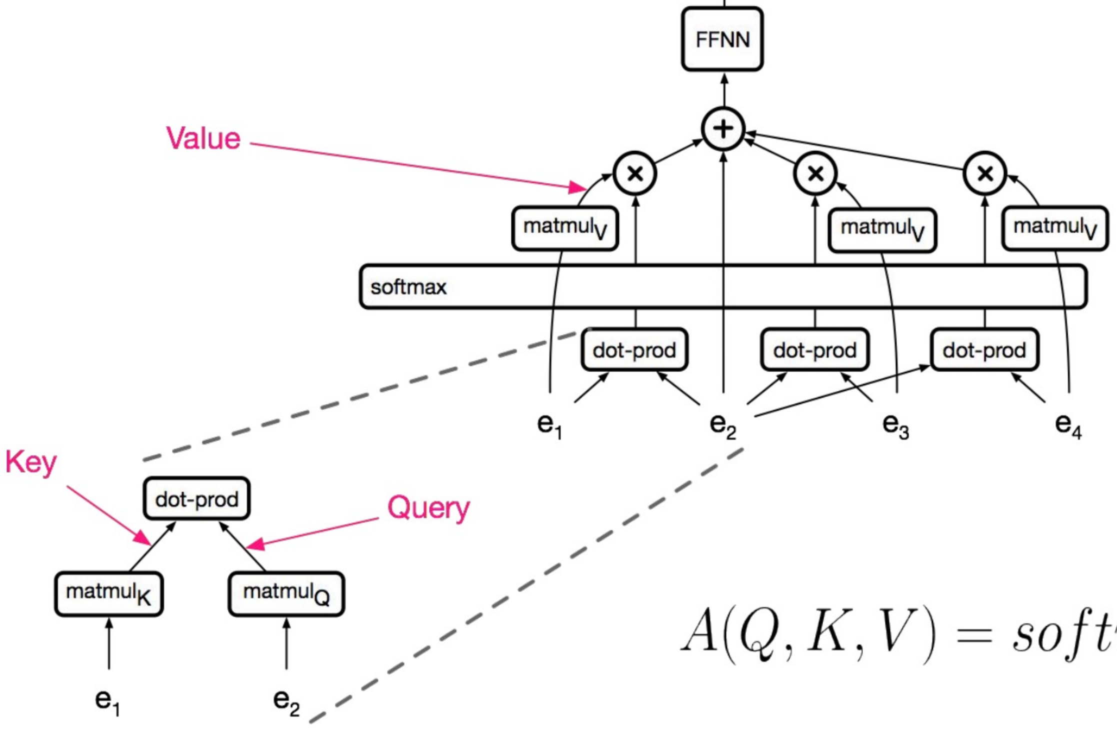




3Blue1Brown [“Attention in Transformers”]

The "Transformer": Encoder

Encoder Self-Attention



$$A(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Vasvani ["Self-Attention for Generative Models"]

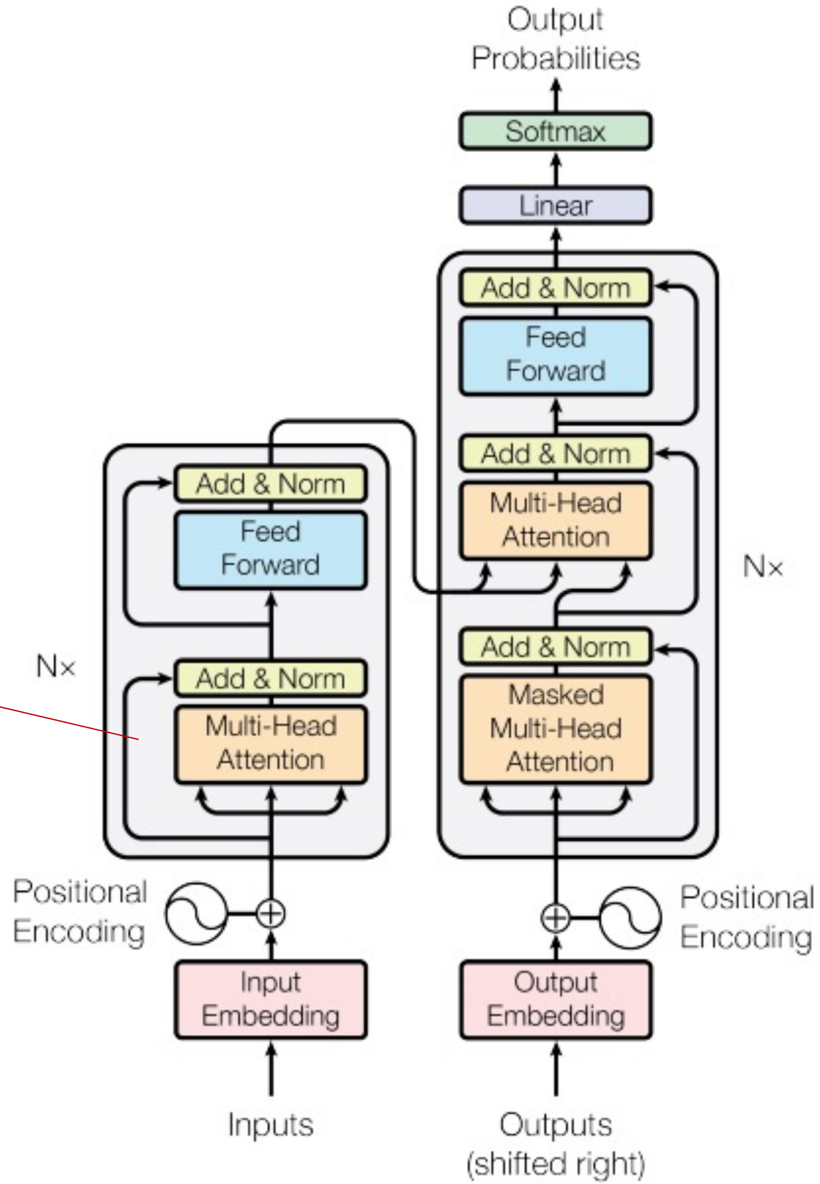
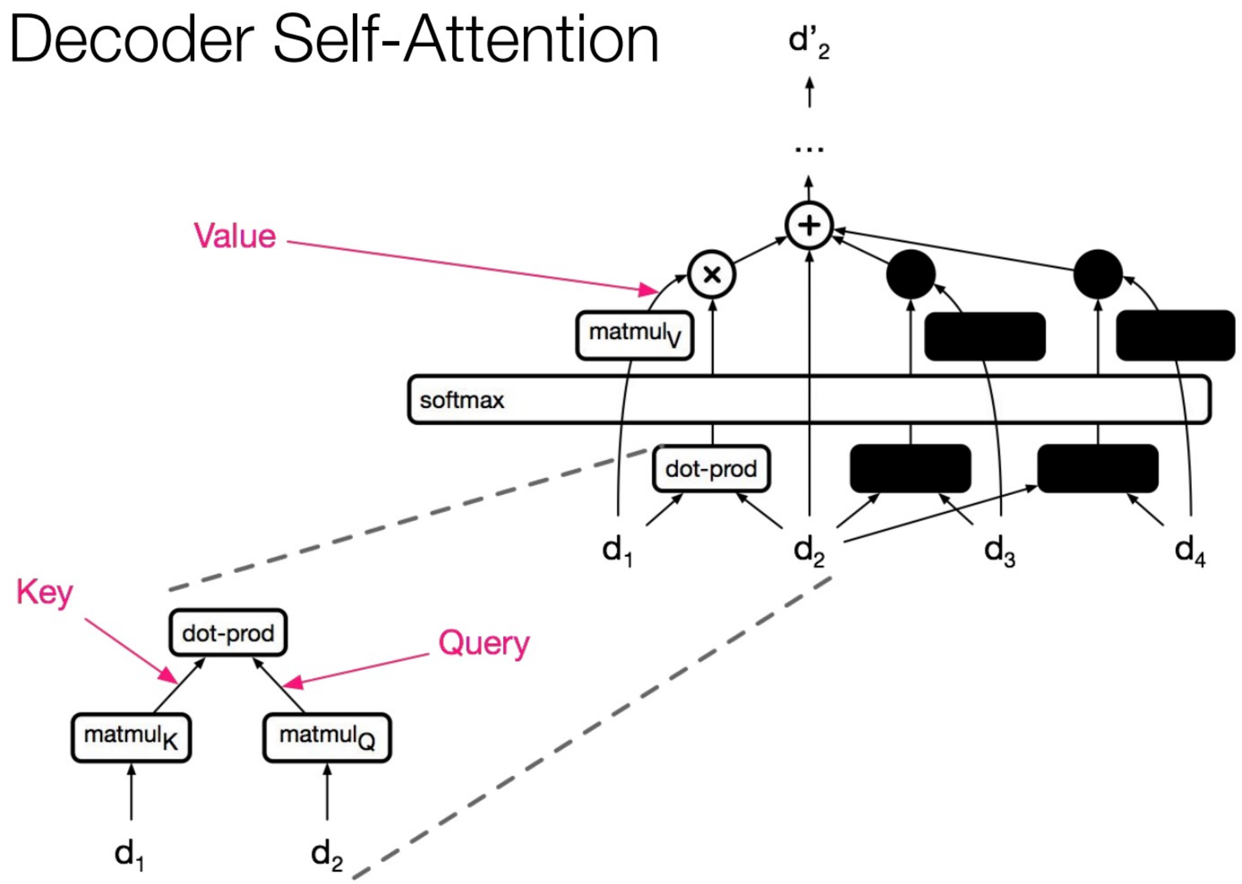


Figure 1: The Transformer - model architecture.

The "Transformer": Decoder



Vasvani ["Self-Attention for Generative Models"]

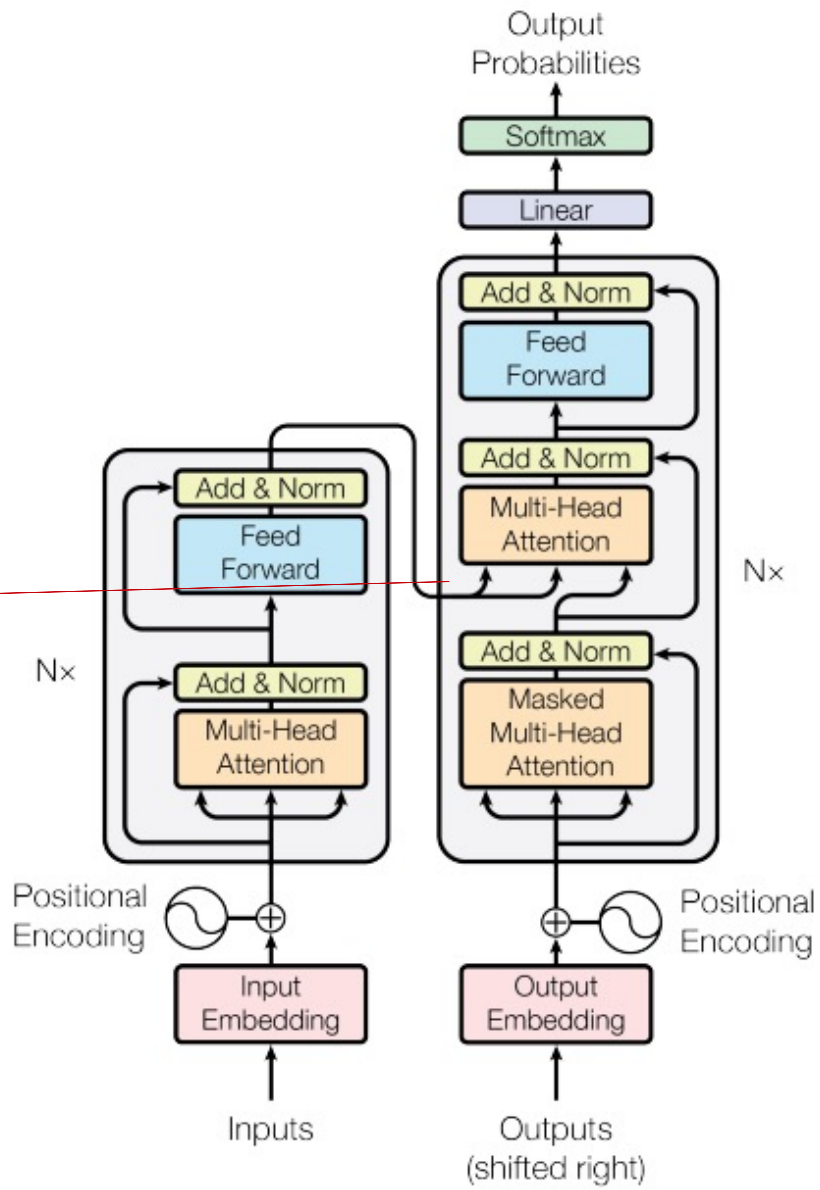


Figure 1: The Transformer - model architecture.

Transformer Tricks

- **Self Attention:** Each layer combines words with others
- **Multi-headed Attention:** 8 attention heads learned independently
- **Normalized Dot-product Attention:** Remove bias in dot product when using large networks
- **Positional Encodings:** Make sure that even if we don't have RNN, can still distinguish positions

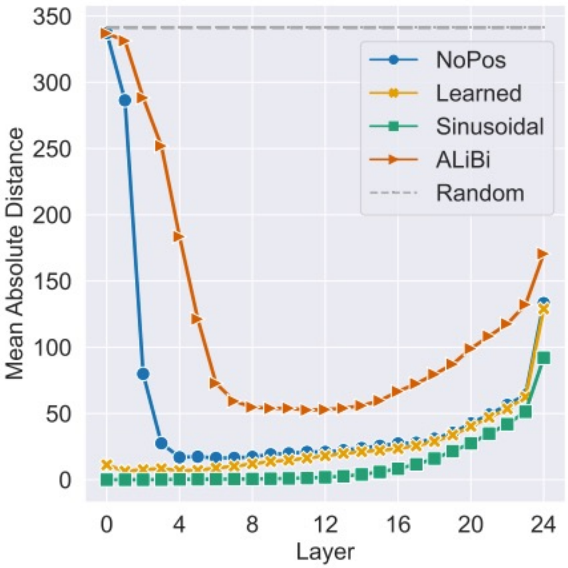
But...

Transformer Language Models without Positional Encodings Still Learn Positional Information

Adi Haviv[τ] Ori Ram[τ] Ofir Press[ω] Peter Izsak[ι] Omer Levy[τμ]

[τ]Tel Aviv University [ω]University of Washington [ι]Intel Labs [μ]Meta AI

{adi.haviv, ori.ram, levyomer}@cs.tau.ac.il, ofirp@cs.washington.edu, peter.izsak@intel.com



Questions?

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON