CMPT 983

Grounded Natural Language Understanding

January 286 2022

Attention for visual grounding

Today

- Attention
  - Review of attention mechanism
  - Attention for visual and language
  - Review of transformers

Attention

Need for "attention"

- Uses encoding of entire input when generating each output token
- Maybe would be useful to focus on a part of the input as the output tokens are generated

Translation

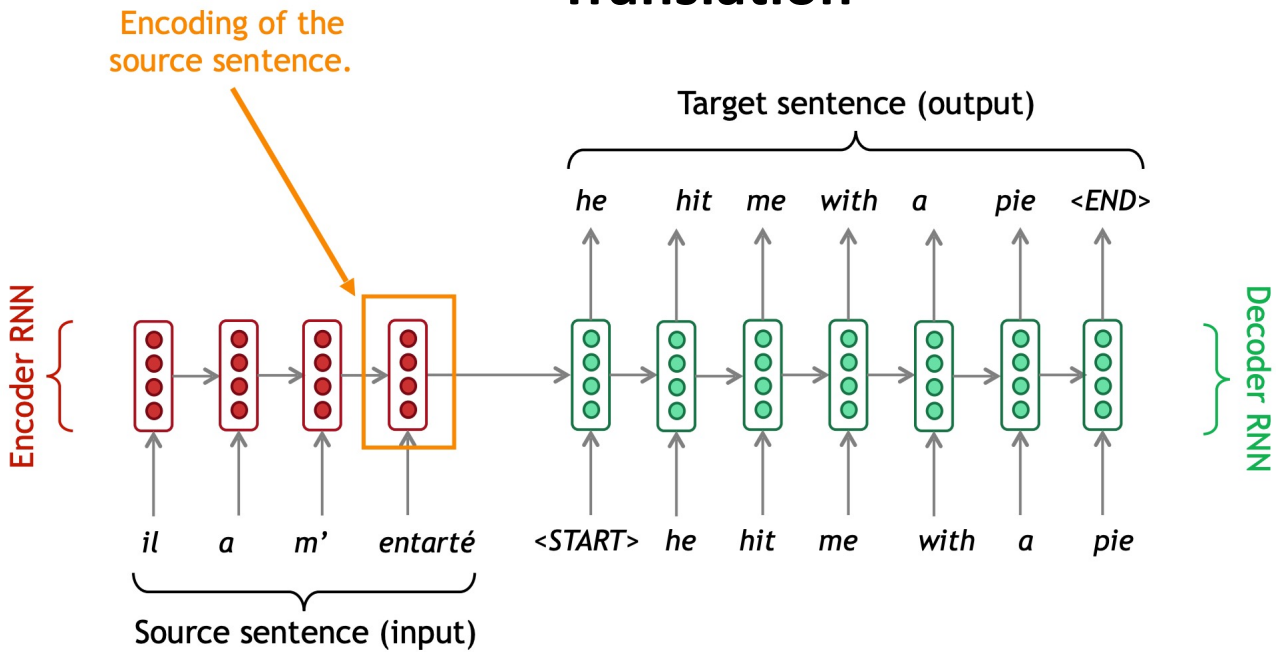


Image credit: Abigail See

Captioning

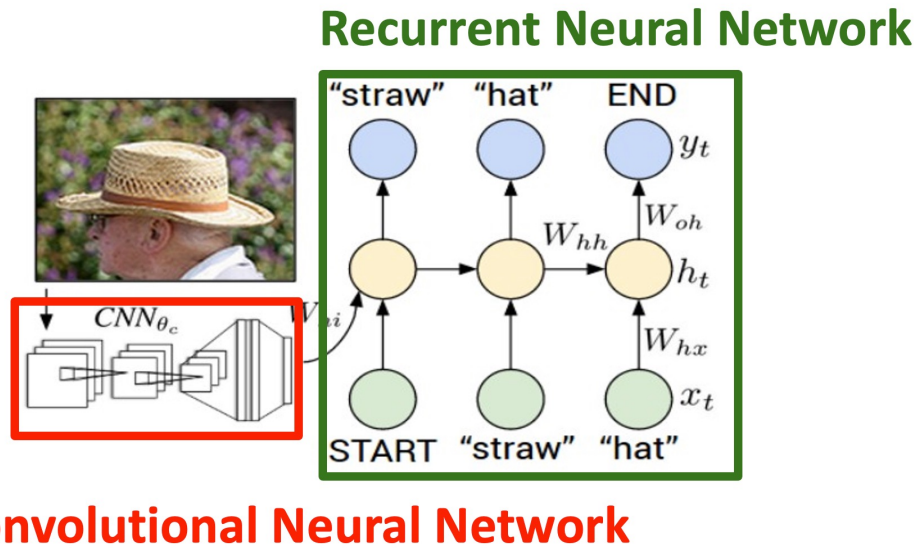


Image credit: Andrej Karpathy

Attention mechanisms - summary

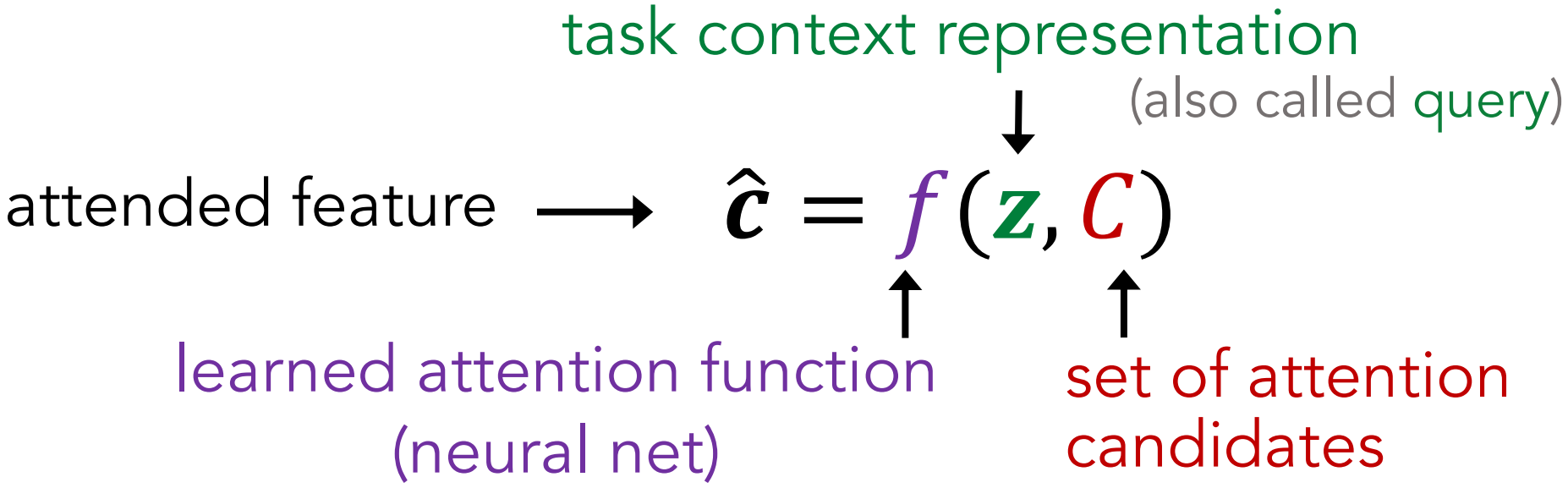
- Attention is probably one of the simplest and most effective ideas in deep learning – proven across many different domains
- Practically: focus on part of input by taking a weighted sum of different input parts
- With sufficient data, attention mechanisms can learn to ground language in visual content from ‘distant’ supervision
- Given the complexity of biological attention systems, assume there is still much to explore… particularly temporal aspects in context of LV&A

Slides Credit: Peter Anderson

Attention

Attention in Neural Networks:

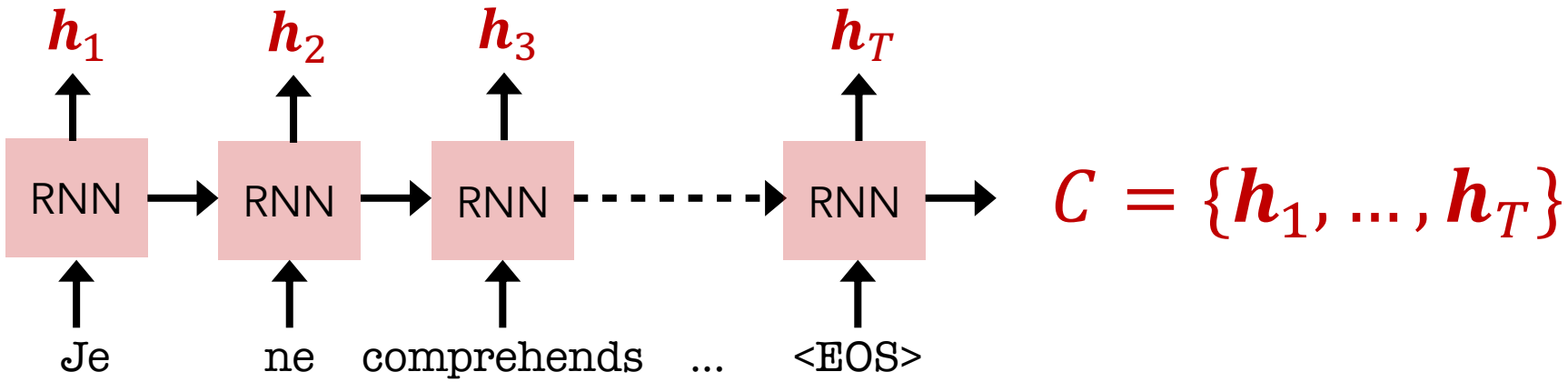
A learned mechanism that **learns to focus** on a subset of the **input** that is most **relevant to the current task**.



Slides Credit: Peter Anderson

Attention over Text

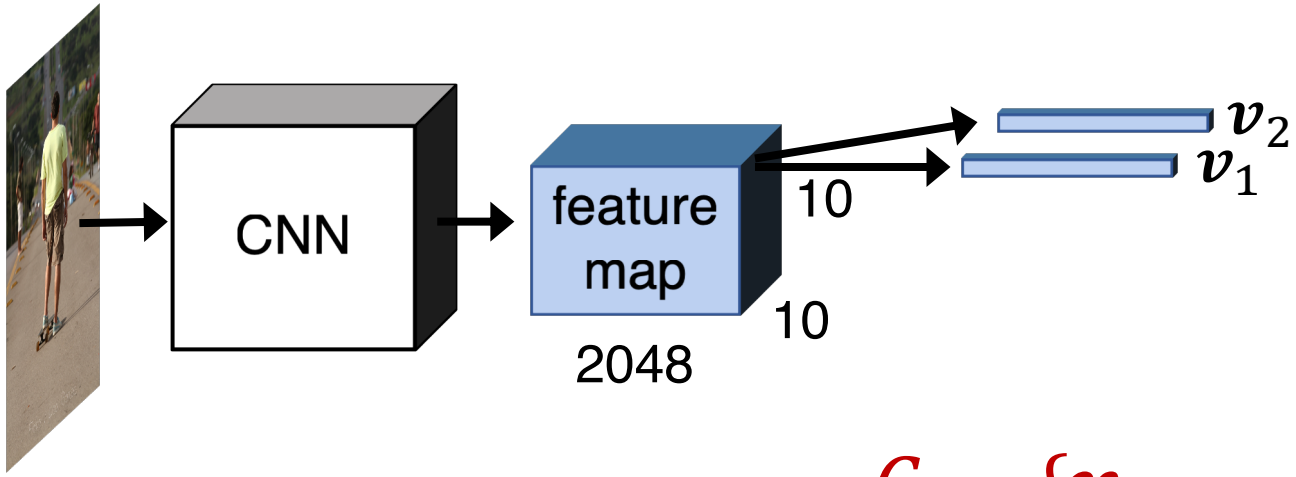
Attention candidates, C typically defined by the hidden states of an encoder (e.g. one feature vector for each word in the input text)

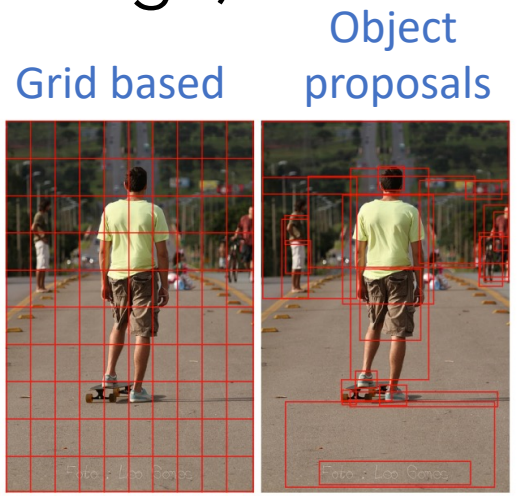


Slides Credit: Peter Anderson

Attention over Visual Features

- Attention candidates, $\boldsymbol{C}$ typically defined by the spatial output of a CNN (feature vectors for different parts of the image)





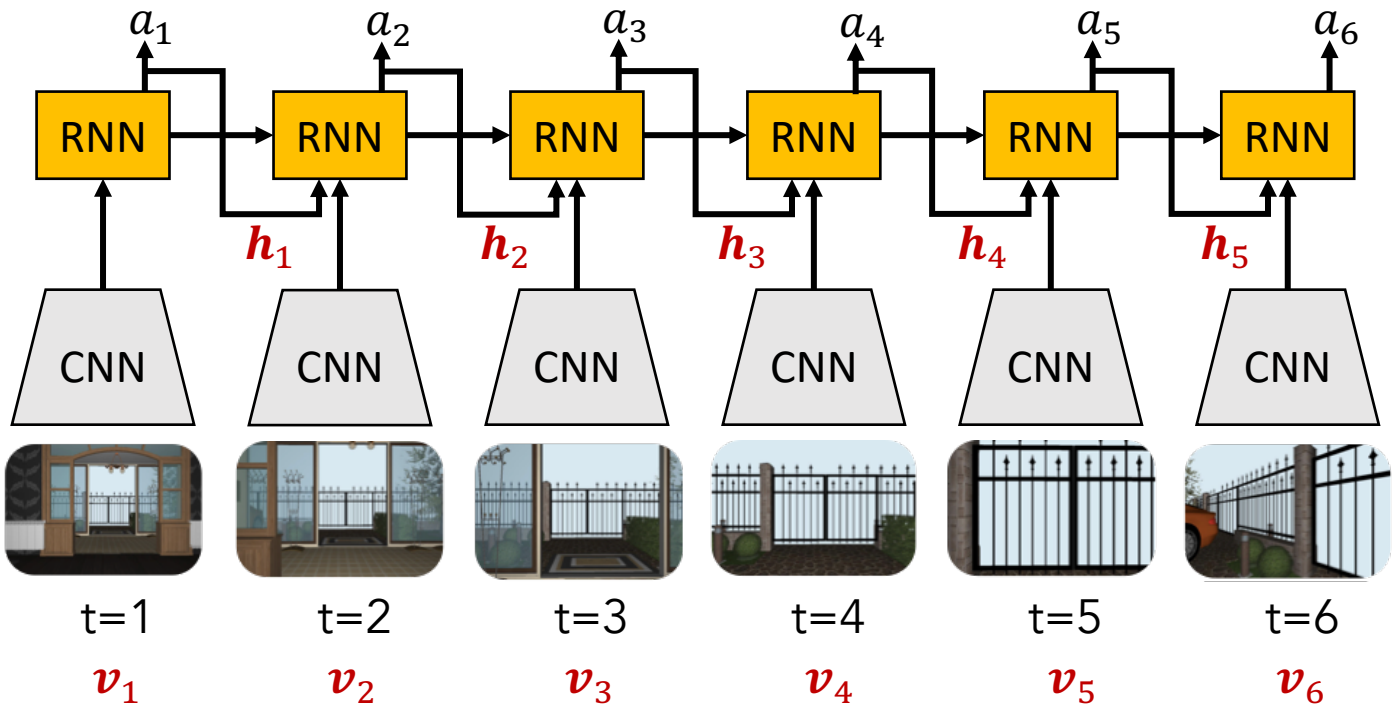
$$C = \{v_1, \ldots, v_{100}\}$$

Slides Credit: Peter Anderson

Attention over Agent Experience

Embodied AI (visual language navigation)

- Attention candidates, $\boldsymbol{C}$ as agent hidden state or visual vectors



$$C = \{h_1, \dots, h_5\}$$

or

$$C = \{v_1, \dots, v_6\}$$

Computing attention

Attention function, f

$$a_i = g(\boldsymbol{c}_i, \boldsymbol{z})$$
$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{a})$$
$$\hat{\boldsymbol{c}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{c}_i$$

Attention scores: $\boldsymbol{a}$ (unnormalized)

Attention weights: $\boldsymbol{\alpha}$ (normalized)

Final attention output

Weighted sum of context features (or values)

Attention score $a_i = g(\boldsymbol{c}_i, \boldsymbol{z})$
how well does the attention candidate $\boldsymbol{c}_i$ match the query $\boldsymbol{z}$

- Dot-product attention:

$$g(c_i, z) = z^\top c_i$$

- Neural network

$$g(c_i, z) = v^\top \tanh\left(W_1 c_i + W_2 z\right)$$

Query-key-value view of attention

Attention function, f

$$a_i = g(\boldsymbol{c}_i, \boldsymbol{z})$$
$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{a})$$
$$\hat{\boldsymbol{c}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{c}_i$$

→

Attention function, f

$$a_i = g(\boldsymbol{k}_i, \boldsymbol{q})$$
$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{a})$$
$$\hat{\boldsymbol{c}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{v}_i$$

Projected query,key,value →

$$\boldsymbol{q} = W_Q \, \boldsymbol{z}$$
$$\boldsymbol{k}_i = W_K \, \boldsymbol{c}_i$$
$$\boldsymbol{v}_i = W_V \, \boldsymbol{c}_i$$

→ Matrix form

$$\boldsymbol{q} = W_Q \, \boldsymbol{z}$$
$$K = W_K \, C^T$$
$$V = W_V \, C^T$$

$$C \in \mathbb{R}^{N \times d_C}$$

Attention is a general concept

Given query q and a set of key-value pairs (K,V)

- **Attention** is a way to compute a ***weighted sum*** of the values dependent on the query and the corresponding keys.

- The query determines what values to focus on
  - We say that the query "attends" to the values
    - In NMT, each decoder hidden state (query) attends to all the encoder hidden states (values)
    - In image captioning, each decoder hidden state (query) attends to image features (values) corresponding to regions of the image

- All of these (key value query) are represented using ***vectors***
  - These vectors are created by multiplying the input embedding by trained weight matrices.

Attention is a general concept

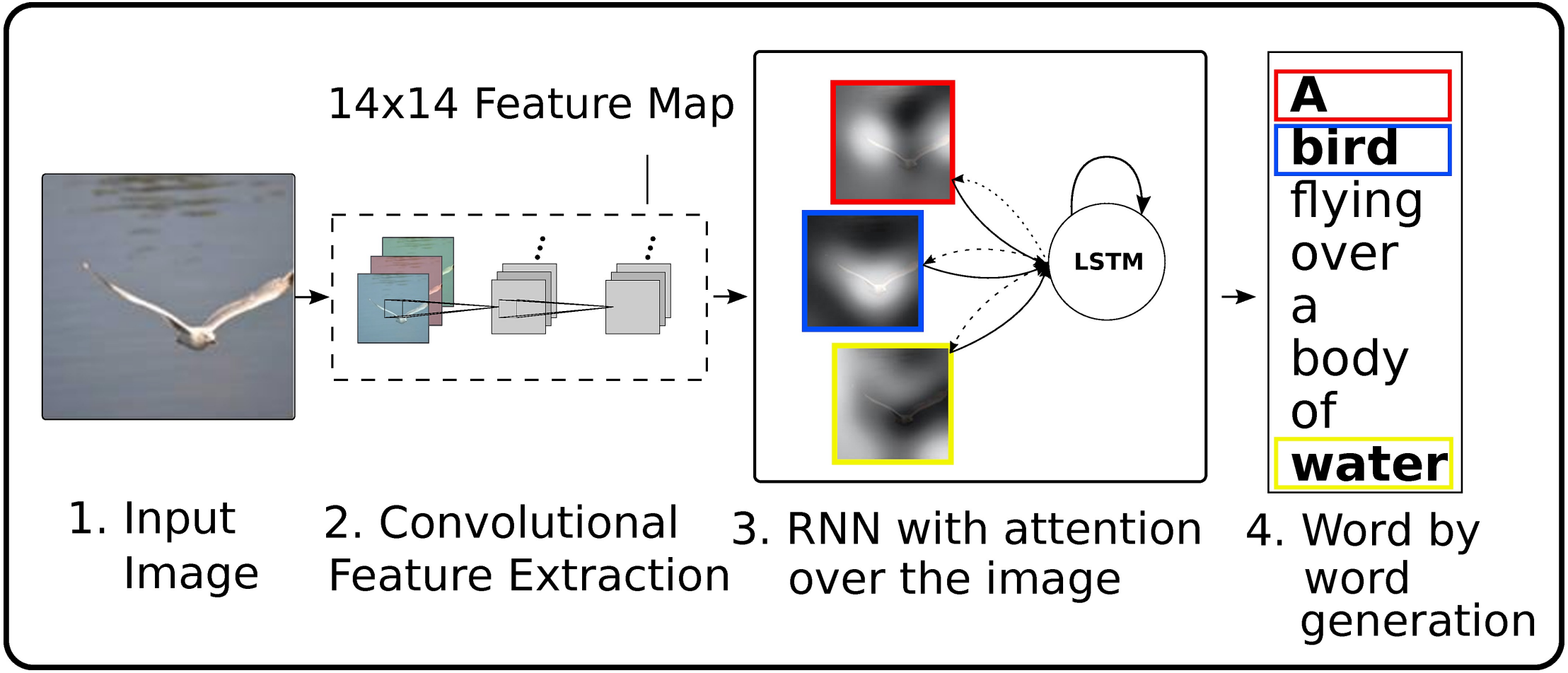
Attention is a way to compute a ***weighted sum*** of the values dependent on the query and the corresponding keys.

Intuition

- The weighted sum is a **selective summary** of the information found in the values
- It is a way to obtain a **fixed-size representation** of an arbitrary set of representations (values) based on some other representation (the query)
- The query and key are used for addressing – contains partial information. While the values provide more complete information.

Attention for grounding

Image captioning example

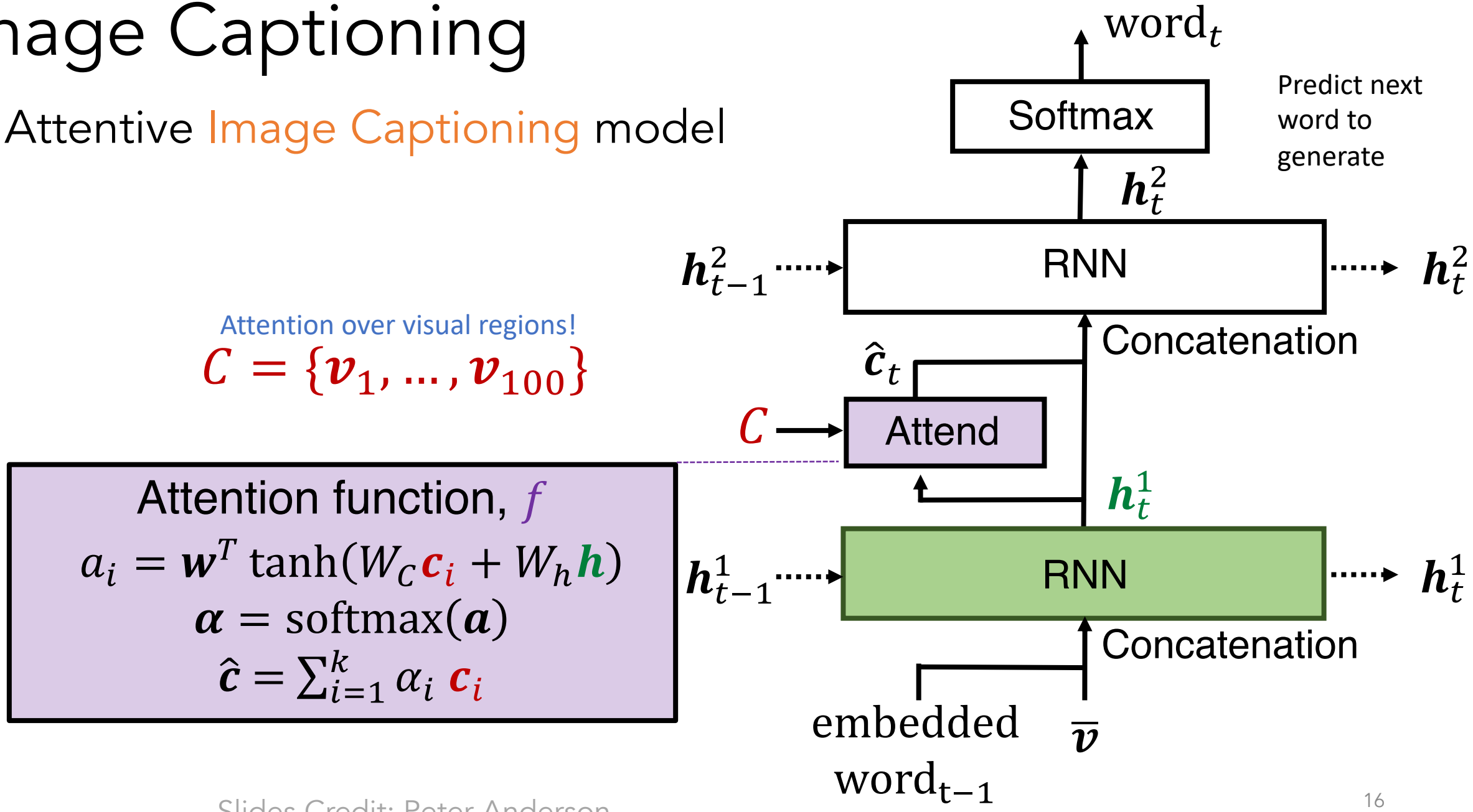


Show, Attend and Tell: Neural Image Caption Generation with Visual Attention [Xu et al. ICML 2015]

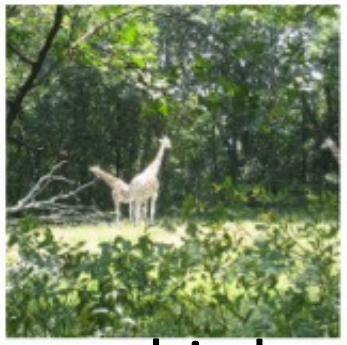
Slides Credit: Peter Anderson

Image Captioning

- Attentive Image Captioning model



Slides Credit: Peter Anderson



A

large

white

bird

standing

in

a

forest

.

[Xu et al. ICML 2015]

‘Soft’ vs ‘hard’ attention

- Soft: Each attention candidate is weighted by α_i

$$\hat{\boldsymbol{v}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{v}_i$$

  - Easy to train (smooth and differentiable)
  - But can be expensive over large input

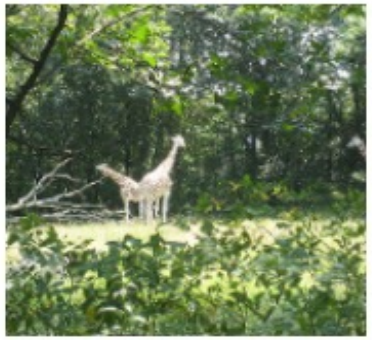
- Hard: Use α_i as a sample probability to pick *one* attention candidate as input to subsequent layers
  - Trainable with REINFORCE approaches (Xu et al. ICML 2015), or Gumbel-Softmax (Jang et al. ICLR 2017)

Soft

Hard

bird

Xu et al. ICML 2015



A

giraffe

standing

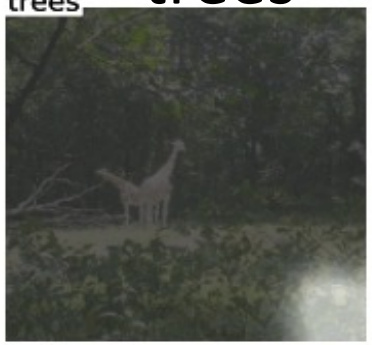
in

a

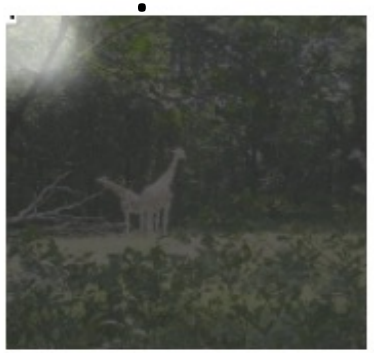
field

with

trees



.



[Xu et al. ICML 2015]

Comparison of attention for image captioning

		BLEU				
Dataset	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
Flickr8k	Google NIC(Vinyals et al., 2014)$^{\dagger\Sigma}$	63	41	27	—	—
	Log Bilinear (Kiros et al., 2014a)$^{\circ}$	65.6	42.4	27.7	17.7	17.31
	Soft-Attention	**67**	44.8	29.9	19.5	18.93
	Hard-Attention	**67**	**45.7**	**31.4**	**21.3**	**20.30**
Flickr30k	Google NIC$^{\dagger\circ\Sigma}$	66.3	42.3	27.7	18.3	—
	Log Bilinear	60.0	38	25.4	17.1	16.88
	Soft-Attention	66.7	43.4	28.8	19.1	**18.49**
	Hard-Attention	**66.9**	**43.9**	**29.6**	**19.9**	18.46
COCO	CMU/MS Research (Chen & Zitnick, 2014)^{a}	—	—	—	—	20.41
	MS Research (Fang et al., 2014)$^{\dagger a}$	—	—	—	—	20.71
	BRNN (Karpathy & Li, 2014)$^{\circ}$	64.2	45.1	30.4	20.3	—
	Google NIC$^{\dagger\circ\Sigma}$	66.6	46.1	32.9	24.6	—
	Log Bilinear$^{\circ}$	70.8	48.9	34.4	24.3	20.03
	Soft-Attention	70.7	49.2	34.4	24.3	**23.90**
	Hard-Attention	**71.8**	**50.4**	**35.7**	**25.0**	23.04

[Xu et al. ICML 2015]

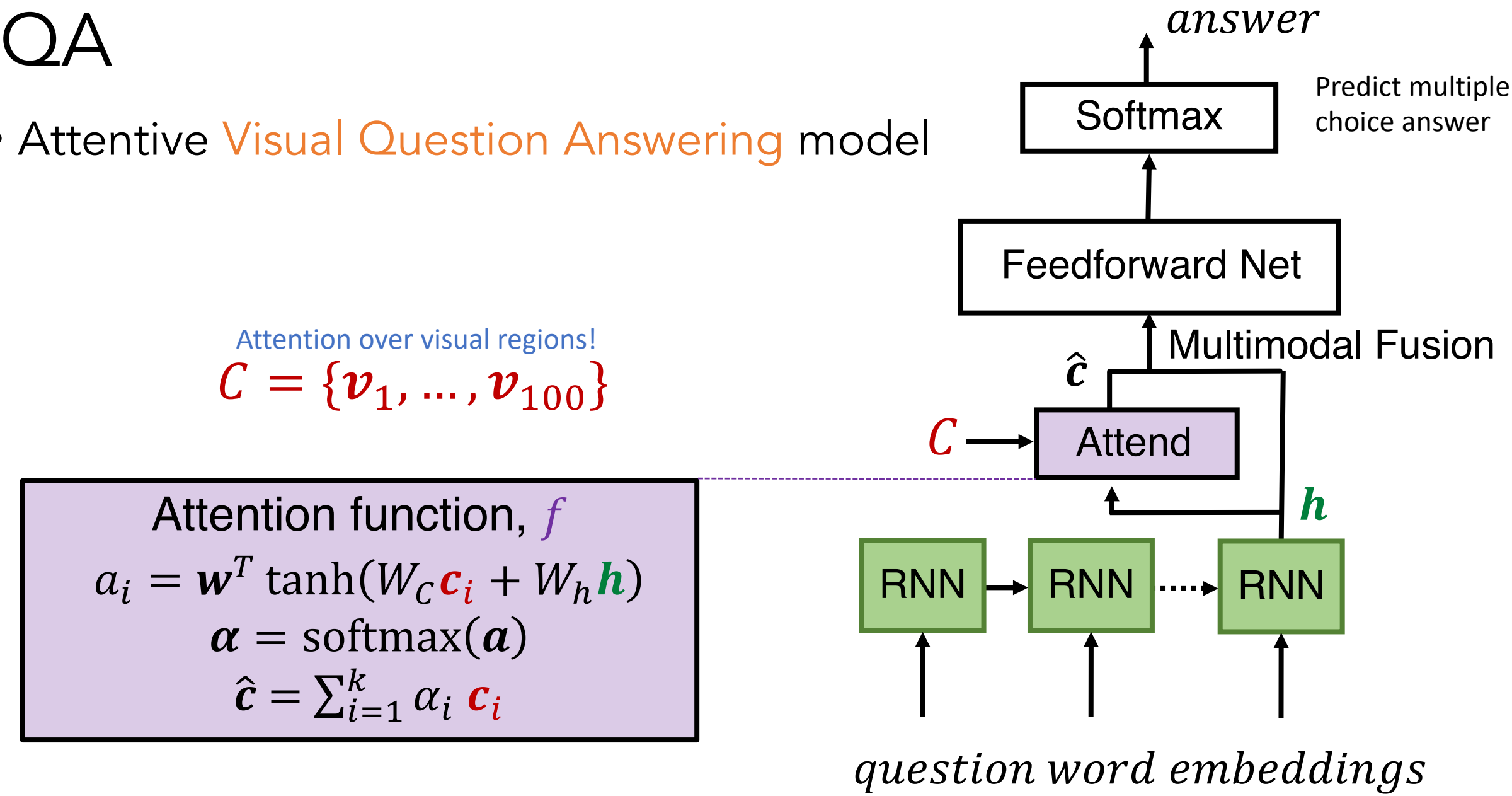
Visual Question Answering

- Given image and questions, predict answer
- Classification problem (select from provided choices or 1000 most common options)

"VQA: Visual Question Answering"
[Antol et al, ICCV 2015]

VQA

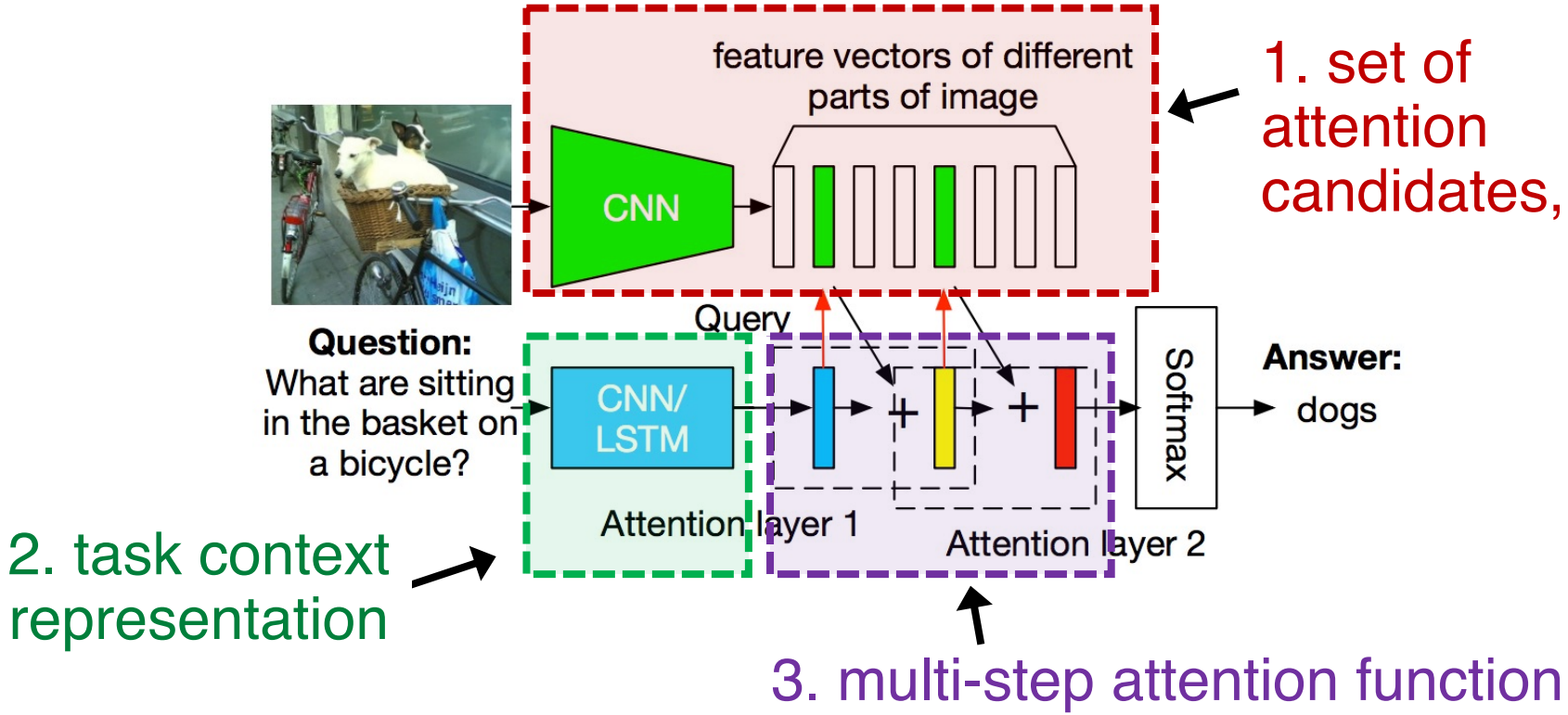
- Attentive Visual Question Answering model



Slides Credit: Peter Anderson

Stacked attention networks

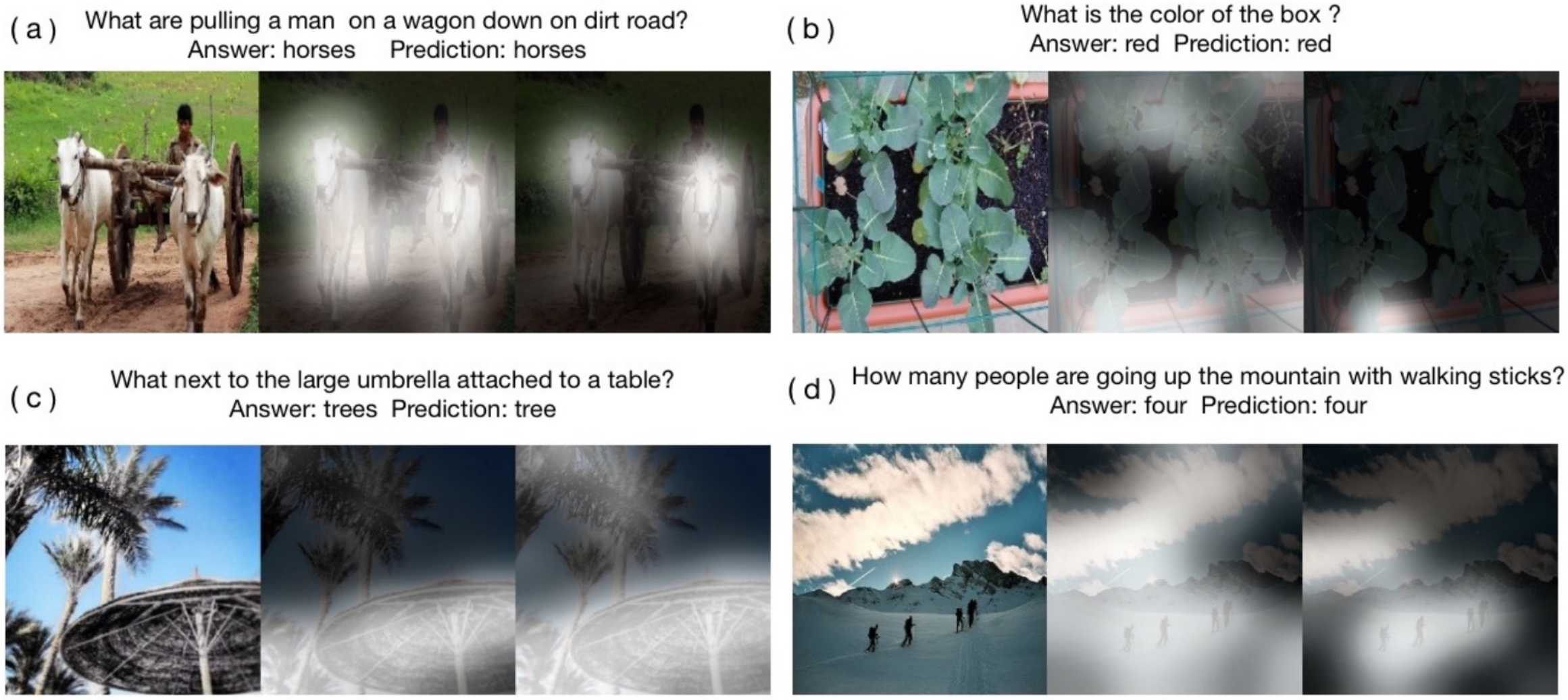
- Multiple attentions steps for VQA



Stacked Attention Networks for Image Question Answering, Yang et al. CVPR 2016

Slides Credit: Peter Anderson

Stacked attention networks

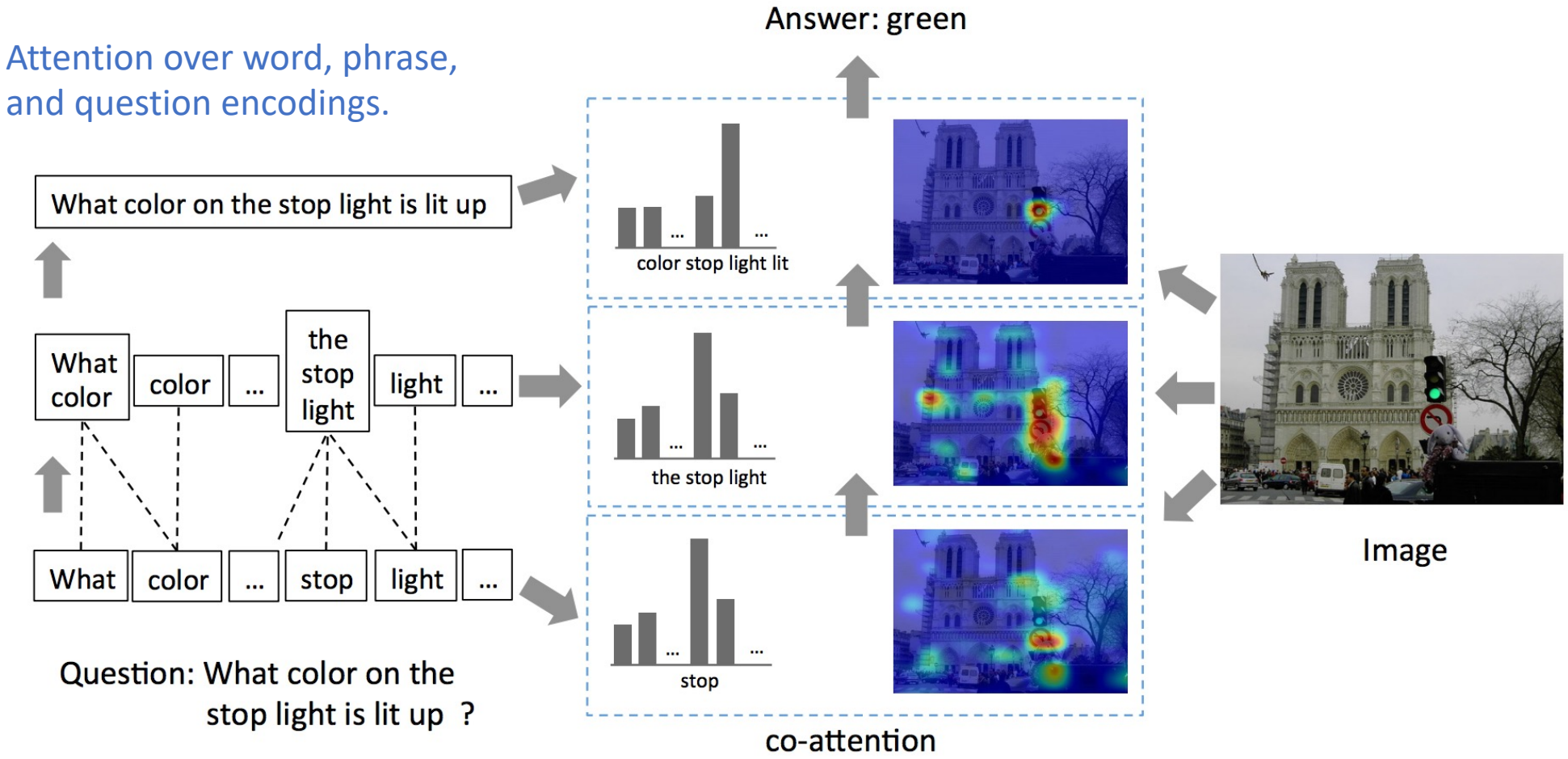


Stacked Attention Networks for Image Question Answering, Yang et al. CVPR 2016

Slides Credit: Peter Anderson

Hierarchical question-image co-attention

- Attending jointly to both question and image in VQA



Hierarchical Question-Image Co-Attention for Visual Question Answering, Lu et al. NIPS 2016

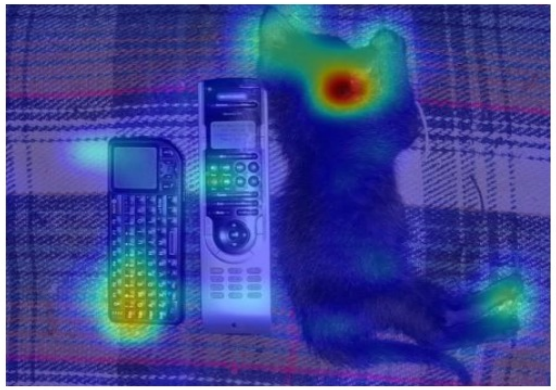
Slides Credit: Peter Anderson

Hierarchical question-image co-attention

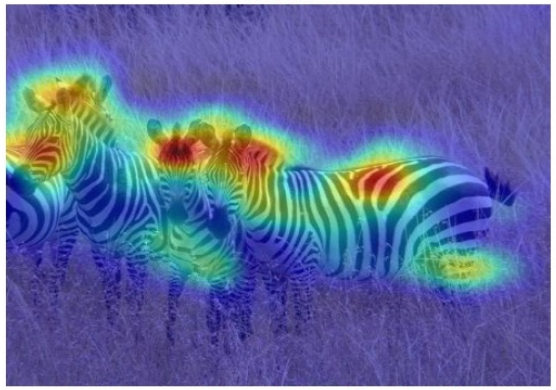
Q: what is the color of the kitten? **A**: **black**

Q: what are standing in tall dry grass look at the tourists? **A**: **zebras**

Q: where is the woman while her baby is sleeping? **A**: **kitchen**



what is the color of the kitten ?



what are standing in tall dry grass look at the tourists ?

where is the woman while her baby is sleeping ?

Hierarchical Question-Image Co-Attention for Visual Question Answering, Lu et al. NIPS 2016

Slides Credit: Peter Anderson

Attention

Attention According to Cognitive Psychology / Neuroscience

- A set of mechanisms that limit some processing to a subset of incoming stimuli (reducing computational demands)
- Can be driven 'top-down' by task demands (i.e. volitionally)
- Can be driven 'bottom-up' by salient stimuli (i.e. involuntarily)
- Visual attention can be applied to features, objects and spatial regions, as well as temporal cues (anticipating events)

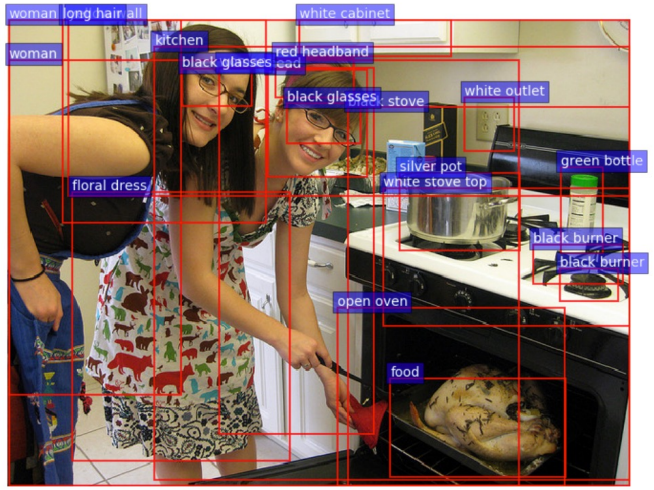
Buschman and Miller 2007, Scholl 2001

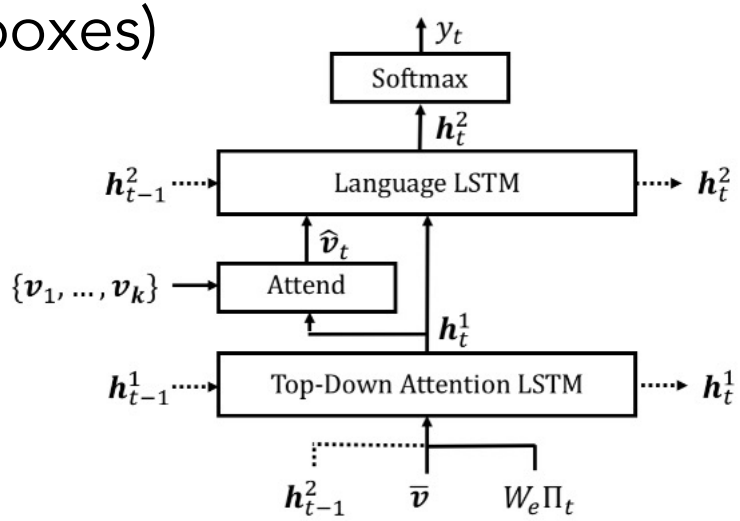
Slides Credit: Peter Anderson

Bottom-up and top-down attention

Combines 'top-down' and 'bottom-up' attention

- 'top-down' attention = *soft* attention over image conditioned on the task
  - VQA: question
  - Image captioning: what has been output before
- 'bottom-up' attention = *hard* attention using Faster-RCNN
  to identify image regions (object bounding boxes)





Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering, Anderson et al. CVPR 2018

Slides Credit: Peter Anderson

Bottom-up and top-down attention

ResNet (10×10): A man sitting on a ~~toilet~~ in a bathroom.

Up-Down: A man sitting on a couch in a bathroom.

Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering, Anderson et al. CVPR 2018

Slides Credit: Peter Anderson

FiLM: Feature-wise Linear Modulation

- Applying attention by scaling and biasing CNN layers

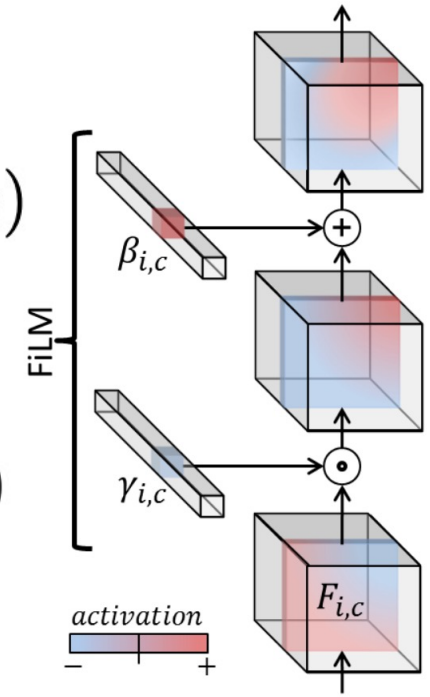
Feature-wise transformations are learned functions of input

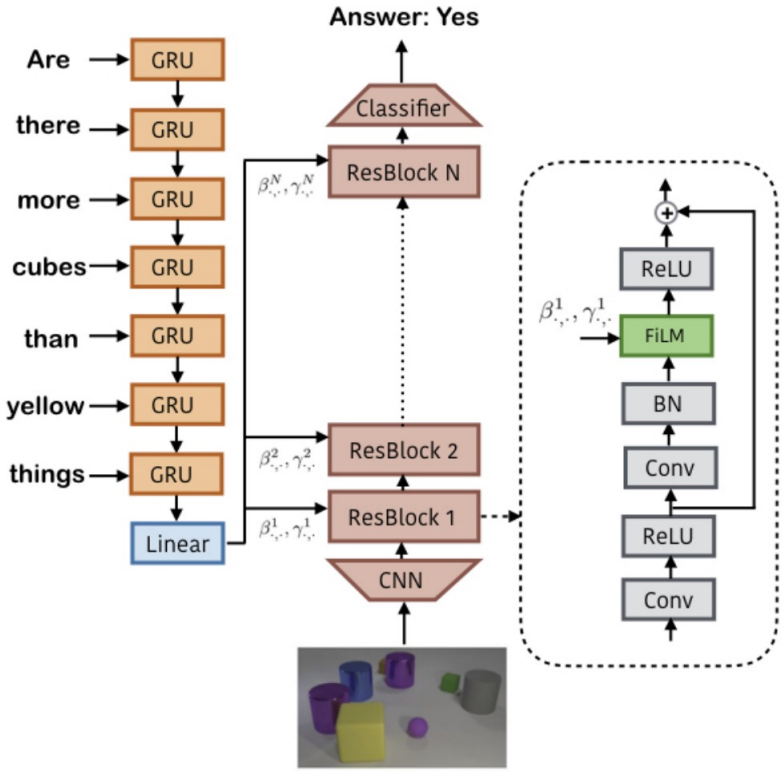
biasing

$$\beta_{i,c} = h_c(\boldsymbol{x}_i)$$

scaling

$$\gamma_{i,c} = f_c(\boldsymbol{x}_i)$$

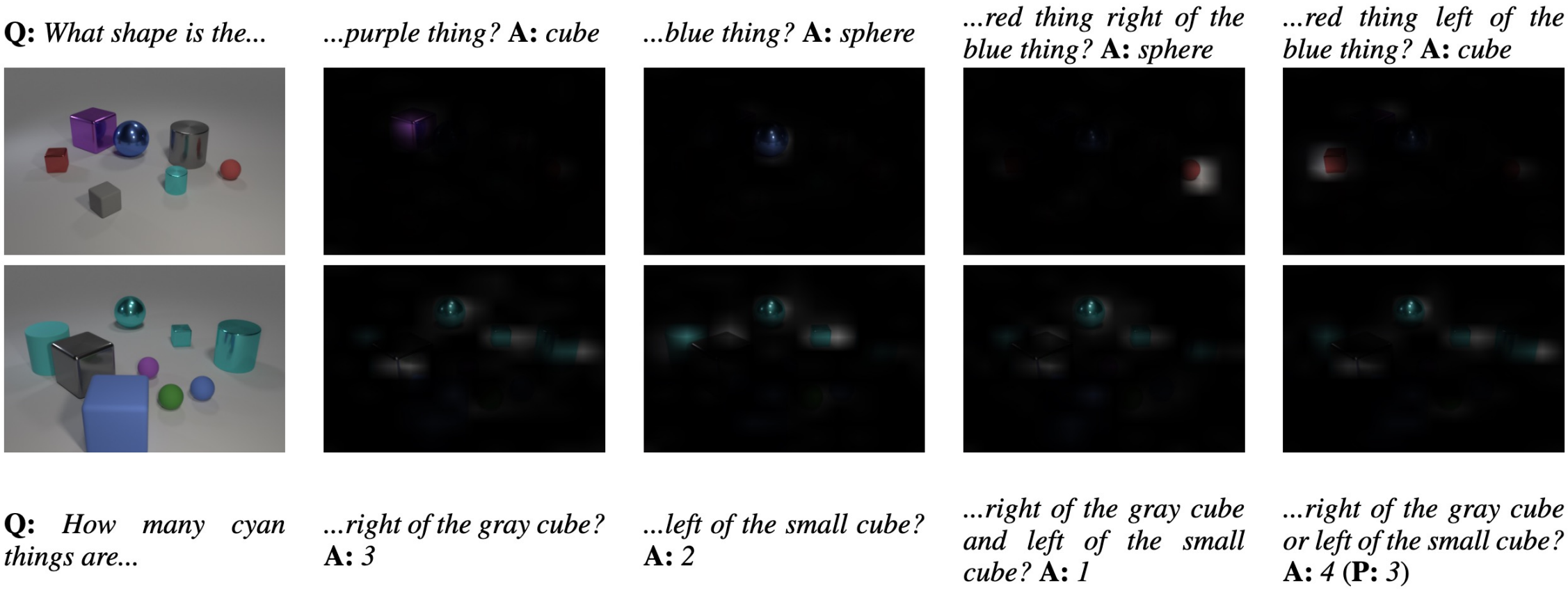




FiLM: Visual Reasoning with a General Conditioning Layer, Perez et al. AAAI 2018

Slides Credit: Peter Anderson

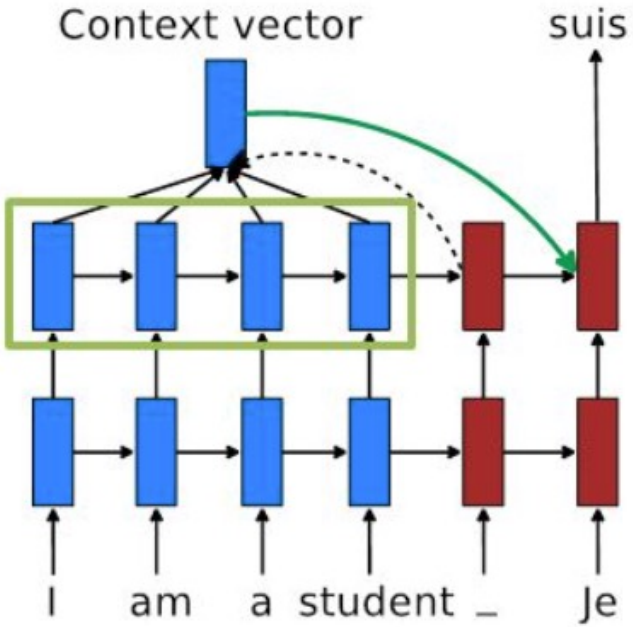
FiLM: Feature-wise Linear Modulation



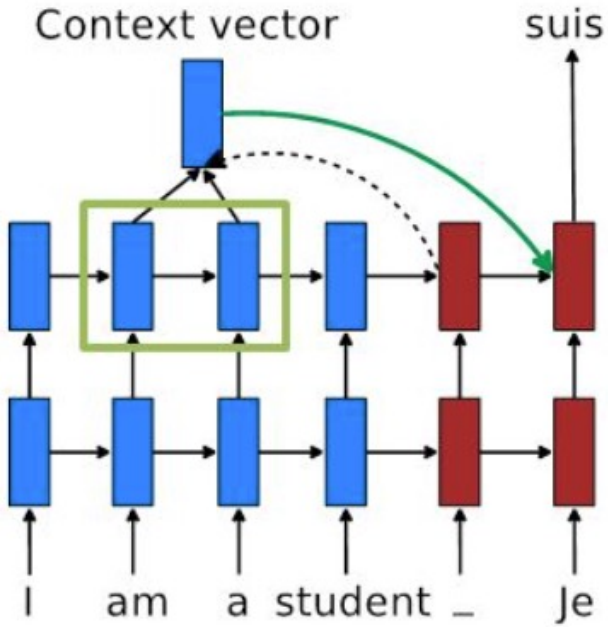
FiLM: Visual Reasoning with a General Conditioning Layer, Perez et al. AAAI 2018

‘Global’ vs ‘local’ attention

- Global: attention over the entire input
- Local: attention over a window (or subset) of the input



Global: ***all*** source states.

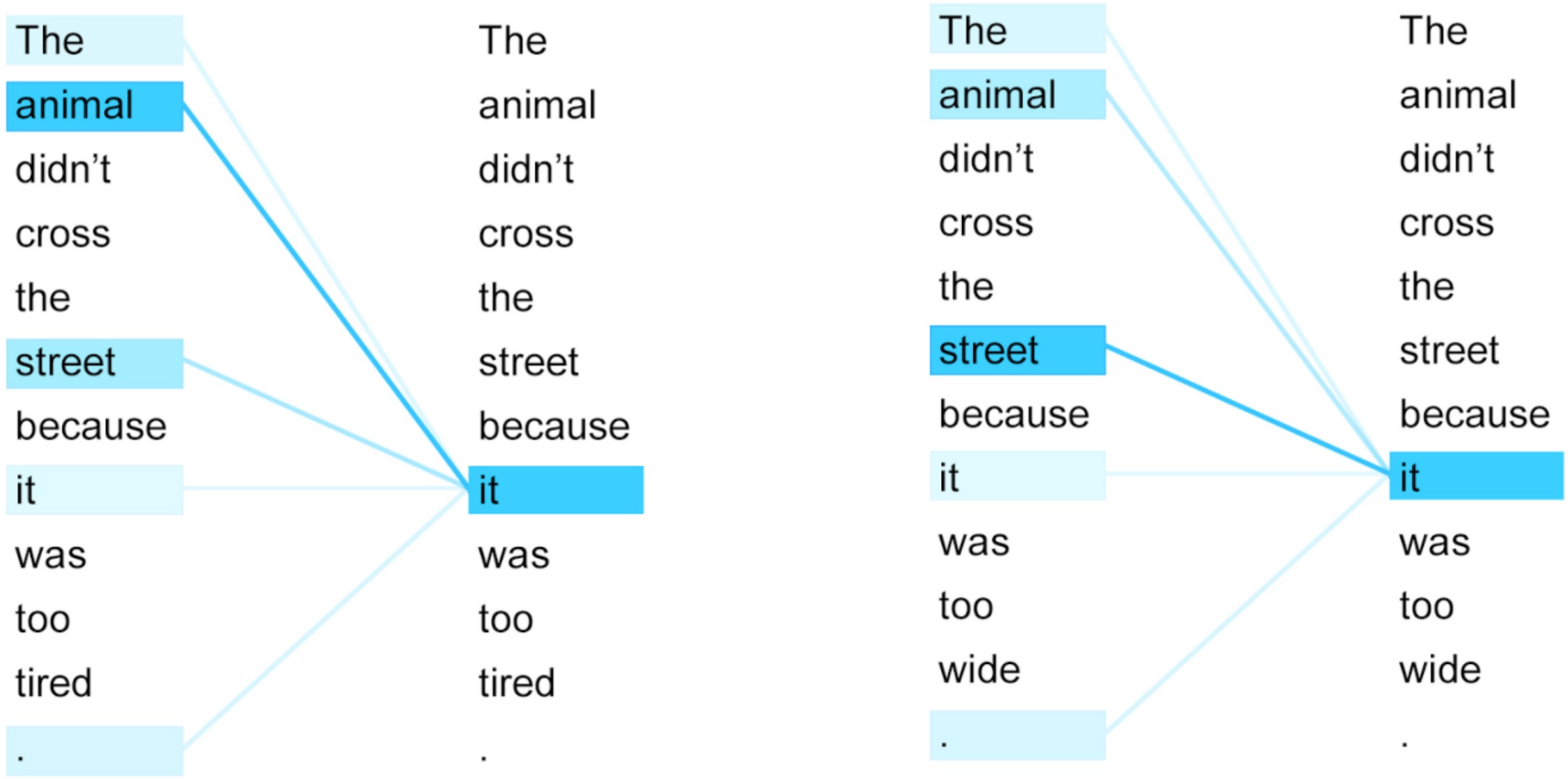


Local: ***subset*** of source states.

Luong et al, 2015

Self-attention

- Attention (correlation) with different parts of itself

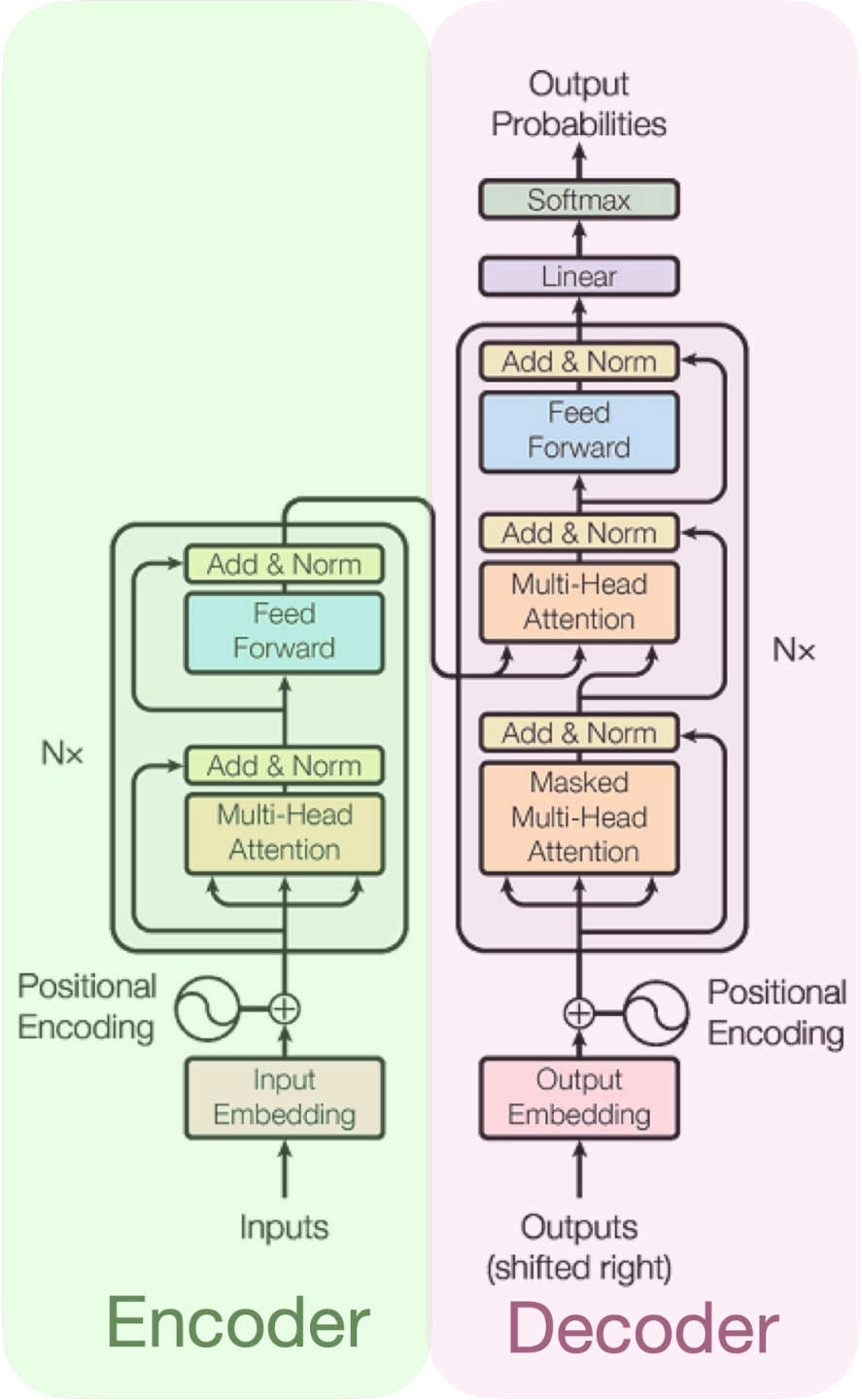


- Transformers: modules with scaled dot-product self-attention

Review of Transformers

Transformers

- NIPS'17: Attention is All You Need
- Originally proposed for NMT (encoder-decoder framework)
- Key idea: Multi-head self-attention
- No recurrence structure so training can be parallelized



Modelling Sequences -- Transformers

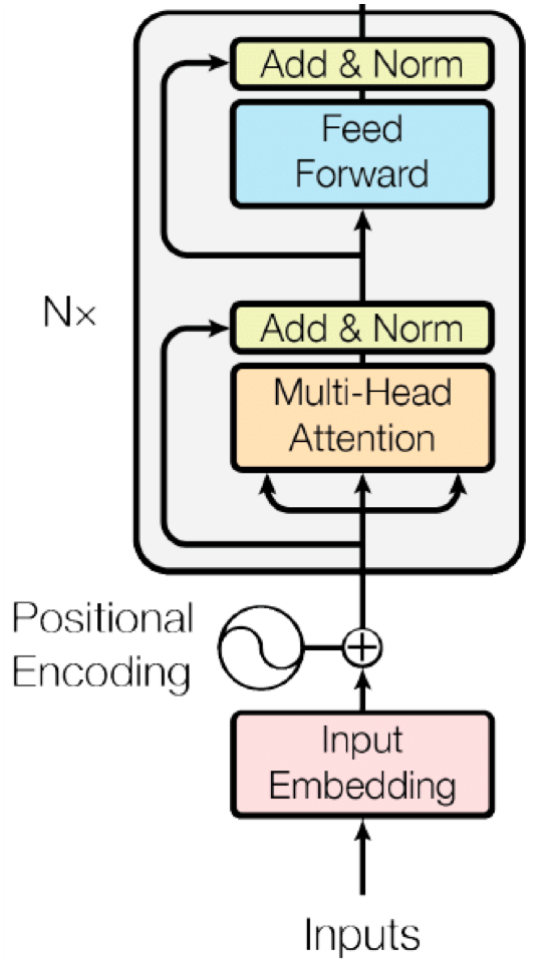
- Each Transformer block has two sub-layers
  - Multi-head attention
  - 2-layer feedforward NN (with ReLU)

- Each sublayer has a residual connection and a layer normalization

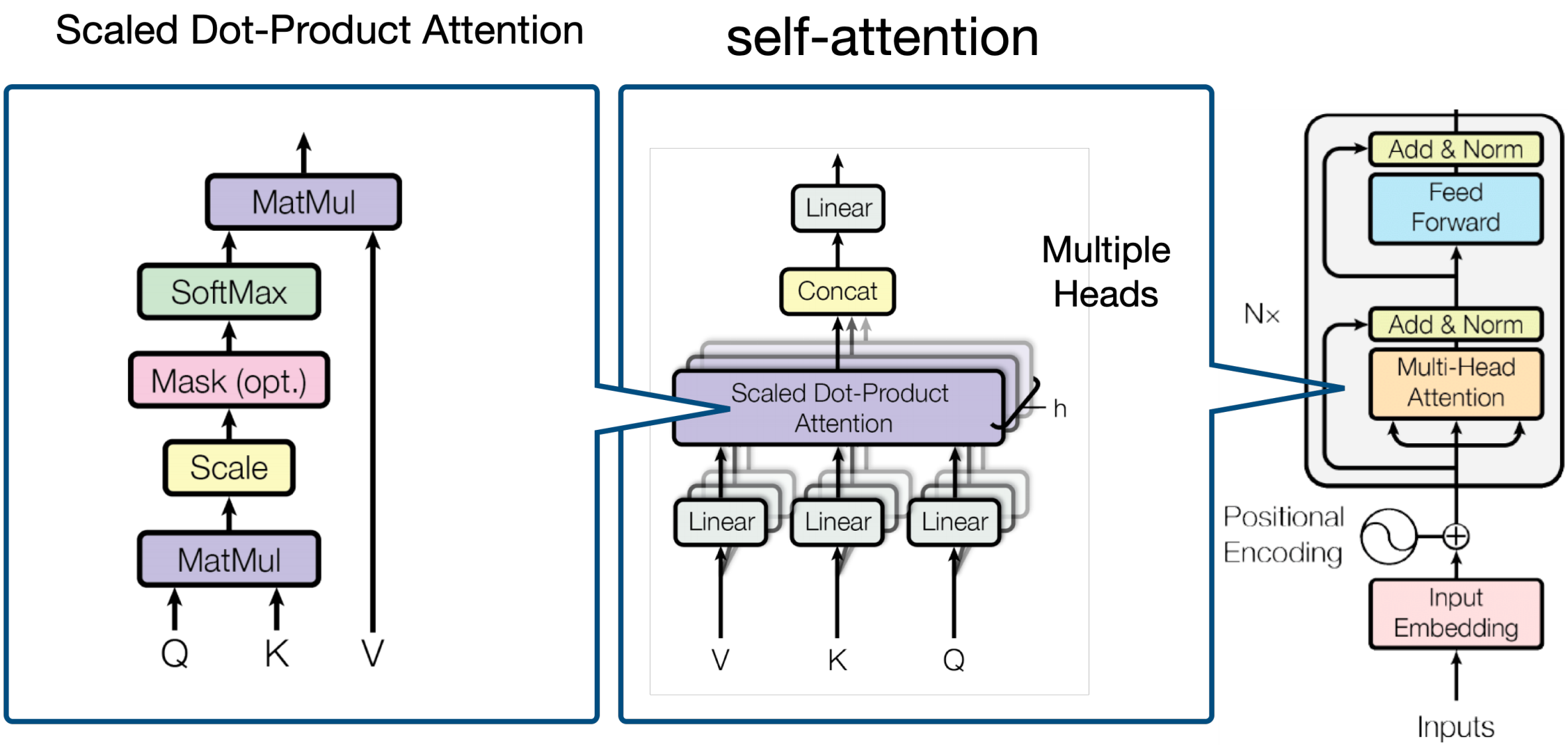
$$\text{LayerNorm}(x + \text{SubLayer}(x))$$

Helps the training process!

- Input layer has a positional encoding



Modelling Sequences -- Transformers



Types of attention scores

Attention function, f

$a_i = g(\boldsymbol{c}_i, \boldsymbol{z})$

$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{a})$

$\hat{\boldsymbol{c}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{c}_i$

- Dot-product attention:

  $g(c_i, z) = z^\top c_i$

- Scaled dot-product attention:

  $g(c_i, z) = z^\top c_i / \sqrt{d}$

- Bilinear / multiplicative attention:

  $g(c_i, z) = z^\top W c_i \in \mathbb{R}$

  where W is a weight matrix

- Additive attention (essentially MLP):

  $g(c_i, z) = v^\top \tanh\left(W_1 c_i + W_2 z\right)$

  where W_1, W_2 are weight matrices and v is a weight vector

Types of attention scores

Attention function, f

$$a_i = g(\boldsymbol{c}_i, \boldsymbol{z})$$
$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{a})$$
$$\hat{\boldsymbol{c}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{c}_i$$

- Dot-product attention:
$$g(c_i, z) = z^\top c_i$$
- Scaled dot-product attention:
$$g(c_i, z) = z^\top c_i / \sqrt{d}$$
- Bilinear / multiplicative attention:
$$g(c_i, z) = z^\top W c_i \in \mathbb{R}$$
where W is a weight matrix
- Additive attention (essentially MLP):
$$g(c_i, z) = v^\top \tanh\left(W_1 c_i + W_2 z\right)$$
where W_1, W_2 are weight matrices and v is a weight vector

Query-key-value view of attention

Attention function, f

$$a_i = g(\boldsymbol{c}_i, \boldsymbol{z})$$
$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{a})$$
$$\hat{\boldsymbol{c}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{c}_i$$

→

Attention function, f

$$a_i = g(\boldsymbol{k}_i, \boldsymbol{q})$$
$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{a})$$
$$\hat{\boldsymbol{c}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{v}_i$$

Projected query,key,value →

$$\boldsymbol{q} = W_Q \, \boldsymbol{z}$$
$$\boldsymbol{k}_i = W_K \, \boldsymbol{c}_i$$
$$\boldsymbol{v}_i = W_V \, \boldsymbol{c}_i$$

→ Matrix form

$$\boldsymbol{q} = W_Q \, \boldsymbol{z}$$
$$K = W_K \, C^T$$
$$V = W_V \, C^T$$

$$C \in \mathbb{R}^{N \times d_C}$$

Query-key-value view of attention

Attention function, f

$$a_i = g(\boldsymbol{c}_i, \boldsymbol{z})$$
$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{a})$$
$$\hat{\boldsymbol{c}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{c}_i$$

→

Attention function, f

$$a_i = g(\boldsymbol{k}_i, \boldsymbol{q})$$
$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{a})$$
$$\hat{\boldsymbol{c}} = \sum_{i=1}^{k} \alpha_i \, \boldsymbol{v}_i$$

Projected query,key,value →

$$\boldsymbol{q} = W_Q \, \boldsymbol{z}$$
$$\boldsymbol{k}_i = W_K \, \boldsymbol{c}_i$$
$$\boldsymbol{v}_i = W_V \, \boldsymbol{c}_i$$

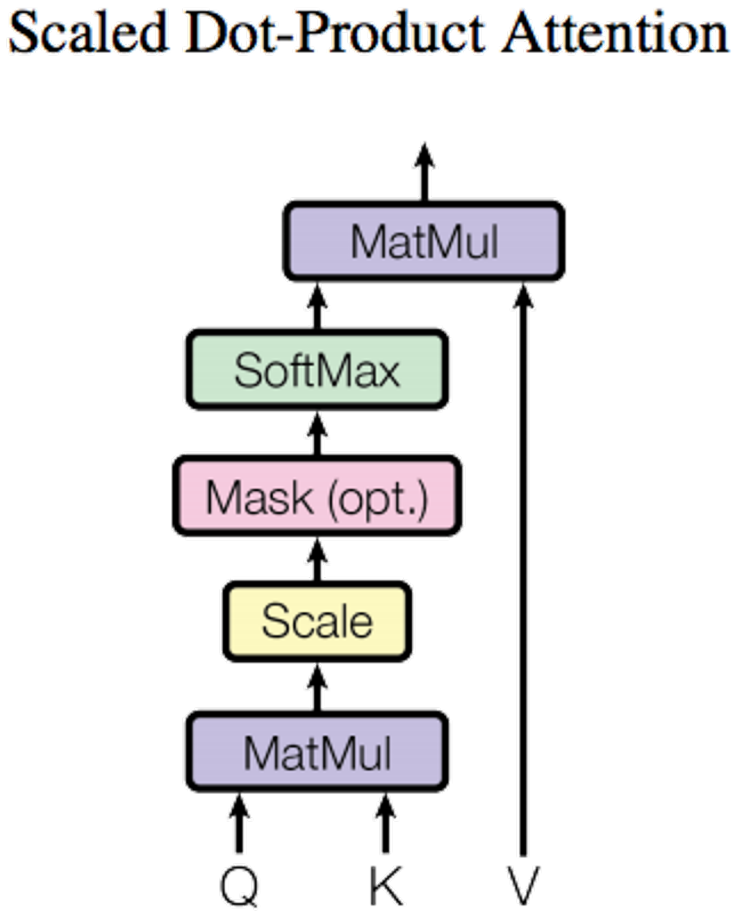
→ Matrix form

$$\boldsymbol{Q} = W_Q \, \boldsymbol{Z}^T$$
$$K = W_K \, C^T$$
$$V = W_V \, C^T$$

$$Z \in \mathbb{R}^{M \times d_Z}, C \in \mathbb{R}^{N \times d_C}$$

Scaled Dot Product Attention

Efficient, stable training



Let $Z \in \mathbb{R}^{M \times d_z}$ be a matrix of task context vectors to attend to

Let $C \in \mathbb{R}^{N \times d_c}$ be a matrix of input vectors to attend over

$\boldsymbol{SDPAttention(Z, C)}$:

$Q = W_Q Z^T$ $\quad W_Q \in \mathbb{R}^{d_q \times d_z}$ $\quad d_q = d_k$

$K = W_K C^T$ $\quad W_K \in \mathbb{R}^{d_k \times d_c}$

$V = W_V C^T$ $\quad W_V \in \mathbb{R}^{d_v \times d_c}$

Return $\hat{V} = softmax\left(\frac{Q^T K}{\sqrt{d_k}}\right) V$

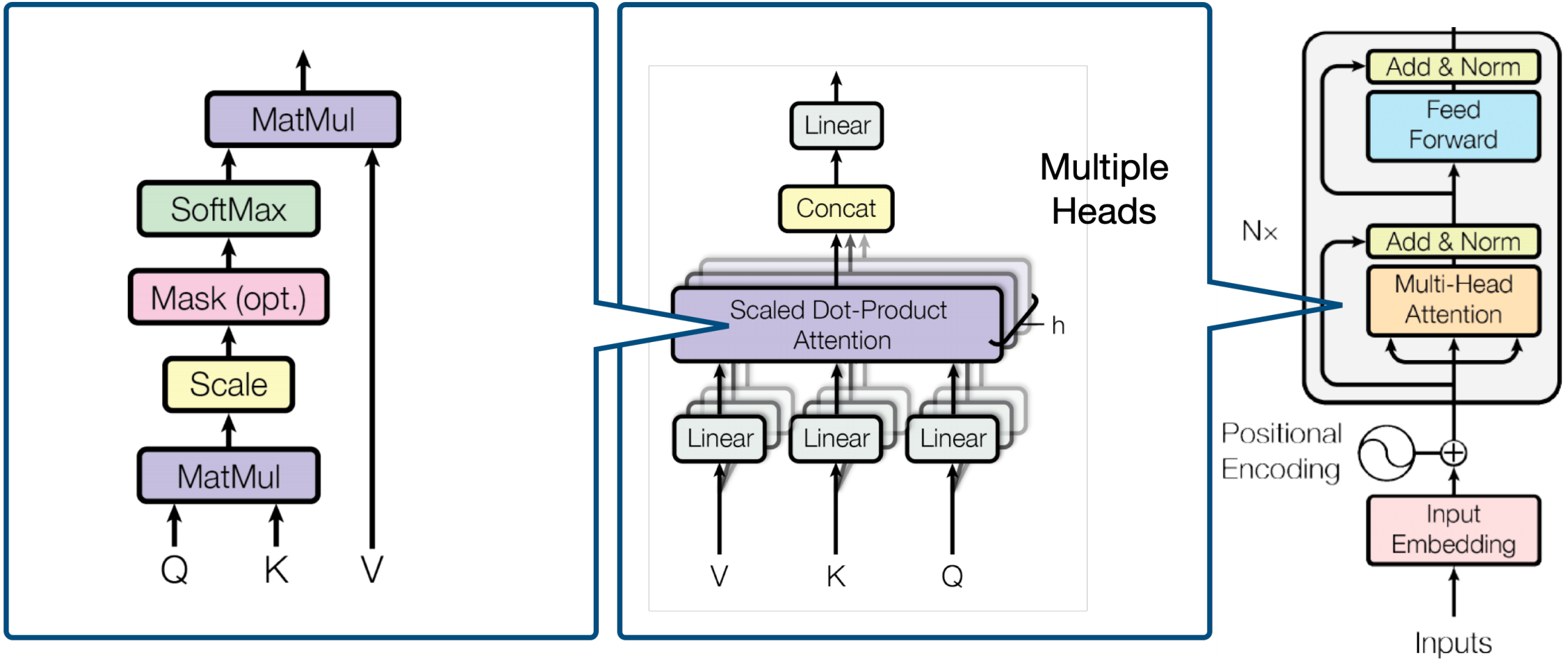
$\hat{V} \in \mathbb{R}^{M \times d_v}$ be a matrix of attended values

Attention Is All You Need https://arxiv.org/pdf/1706.03762.pdf

Modelling Sequences -- Transformers

Scaled Dot-Product Attention

self-attention $\boldsymbol{SDPAttention}(\boldsymbol{C}, \boldsymbol{C})$:



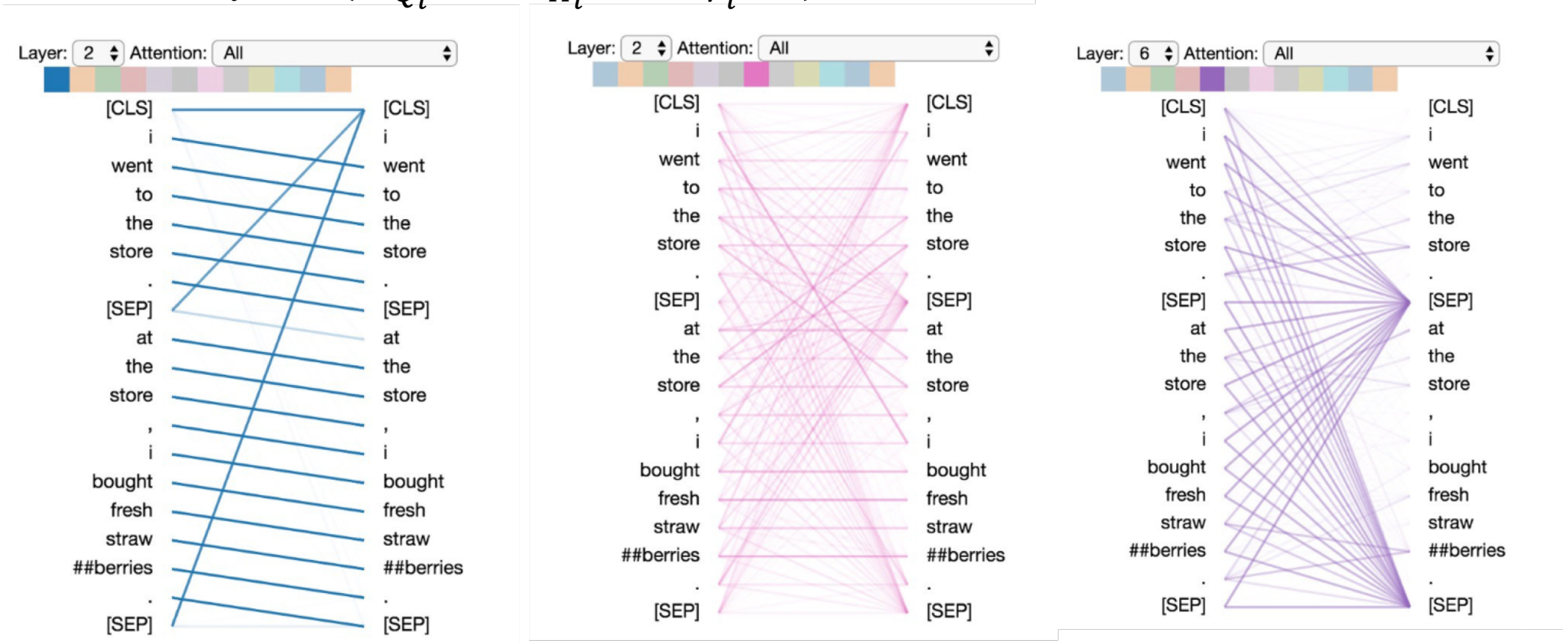
Multi-head attention

One head is not expressive enough. Let's have multiple heads!

$$A(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W_O$$

$$\text{head}_i = A(W_{Q_i}X^T, W_{K_i}X^T, W_{V_i}X^T)$$

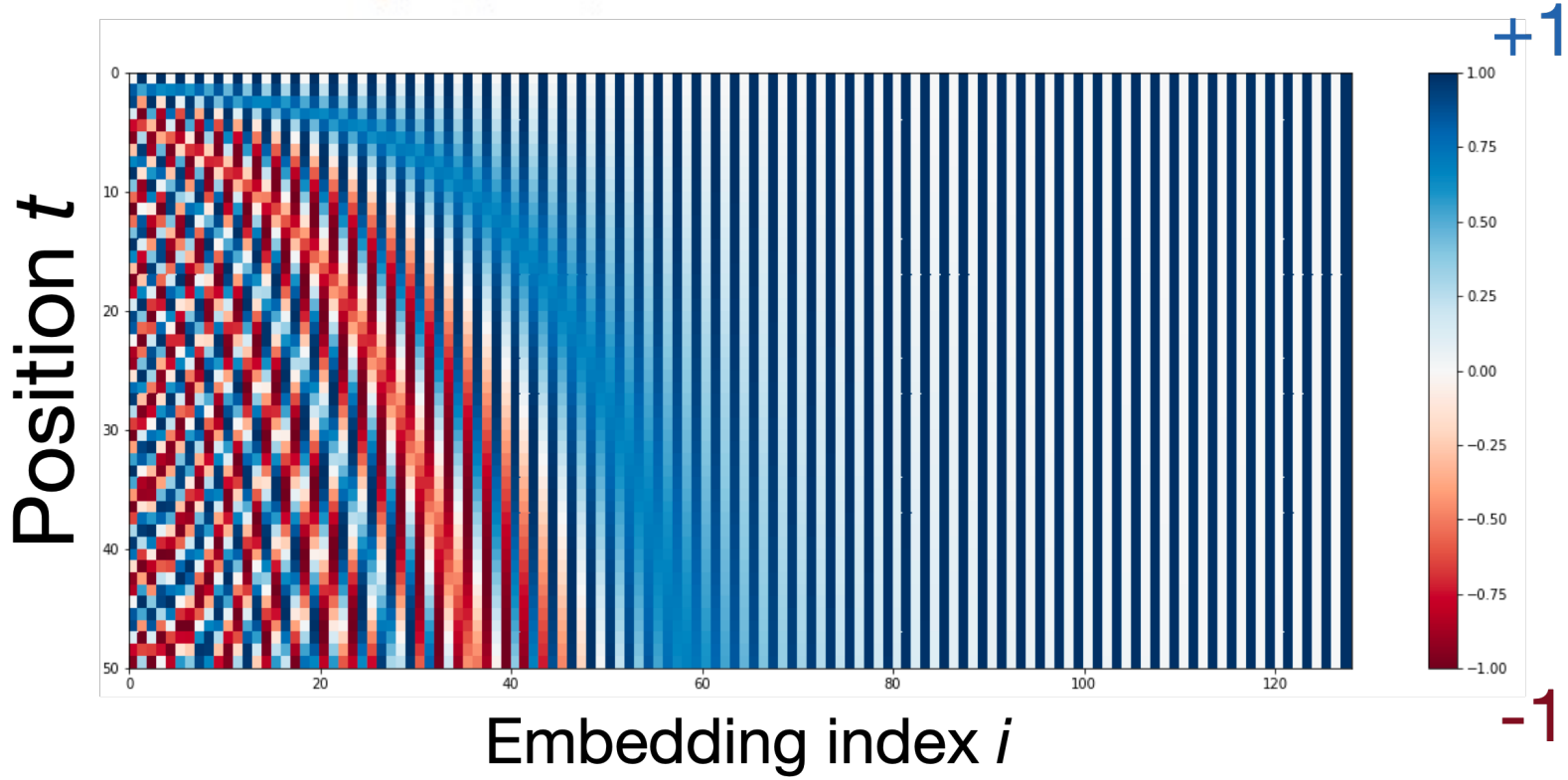
In practice, $h = 8$,
$d = d_{out}/h, W_O \in \mathbb{R}^{d_{out} \times d_{out}}$



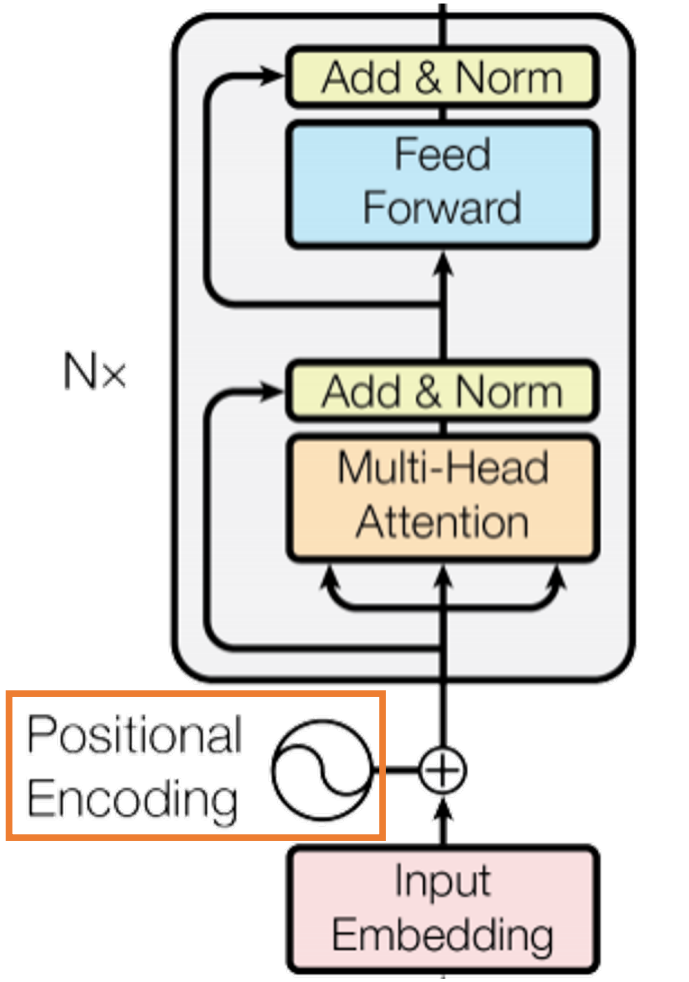
Transformers: Encoding position

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$



$\boldsymbol{SDPAttention(Y, Y)}$:



Attention Is All You Need https://arxiv.org/pdf/1706.03762.pdf

Modelling Sequences -- Transformers

- Each Transformer block has two sub-layers
  - Multi-head attention
  - 2-layer feedforward NN (with ReLU)

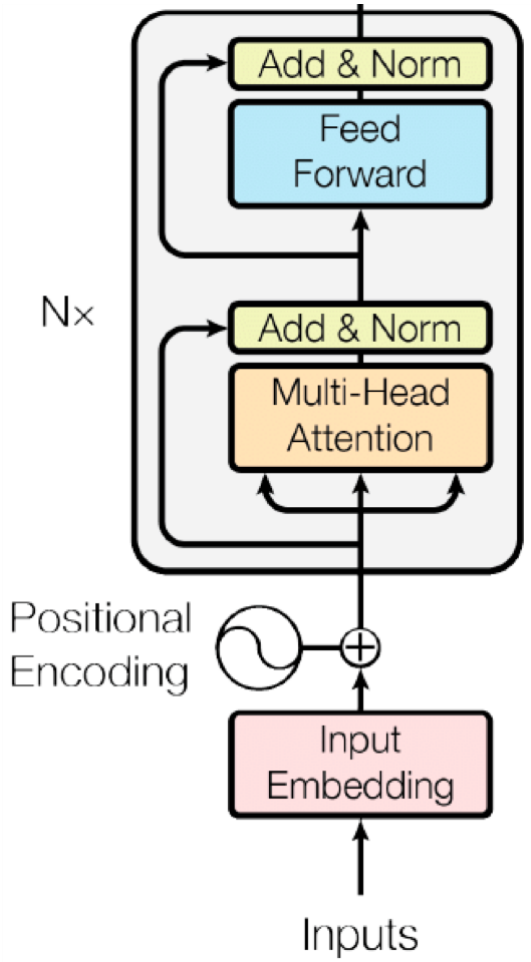
Provides non-linearity

- Each sublayer has a residual connection and a layer normalization

$$\text{LayerNorm}(x + \text{SubLayer}(x))$$

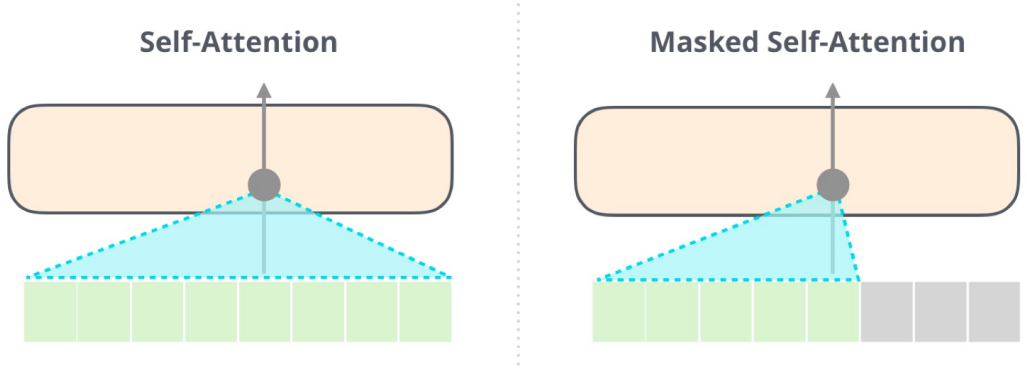
Helps the training process!

- Input layer has a positional encoding

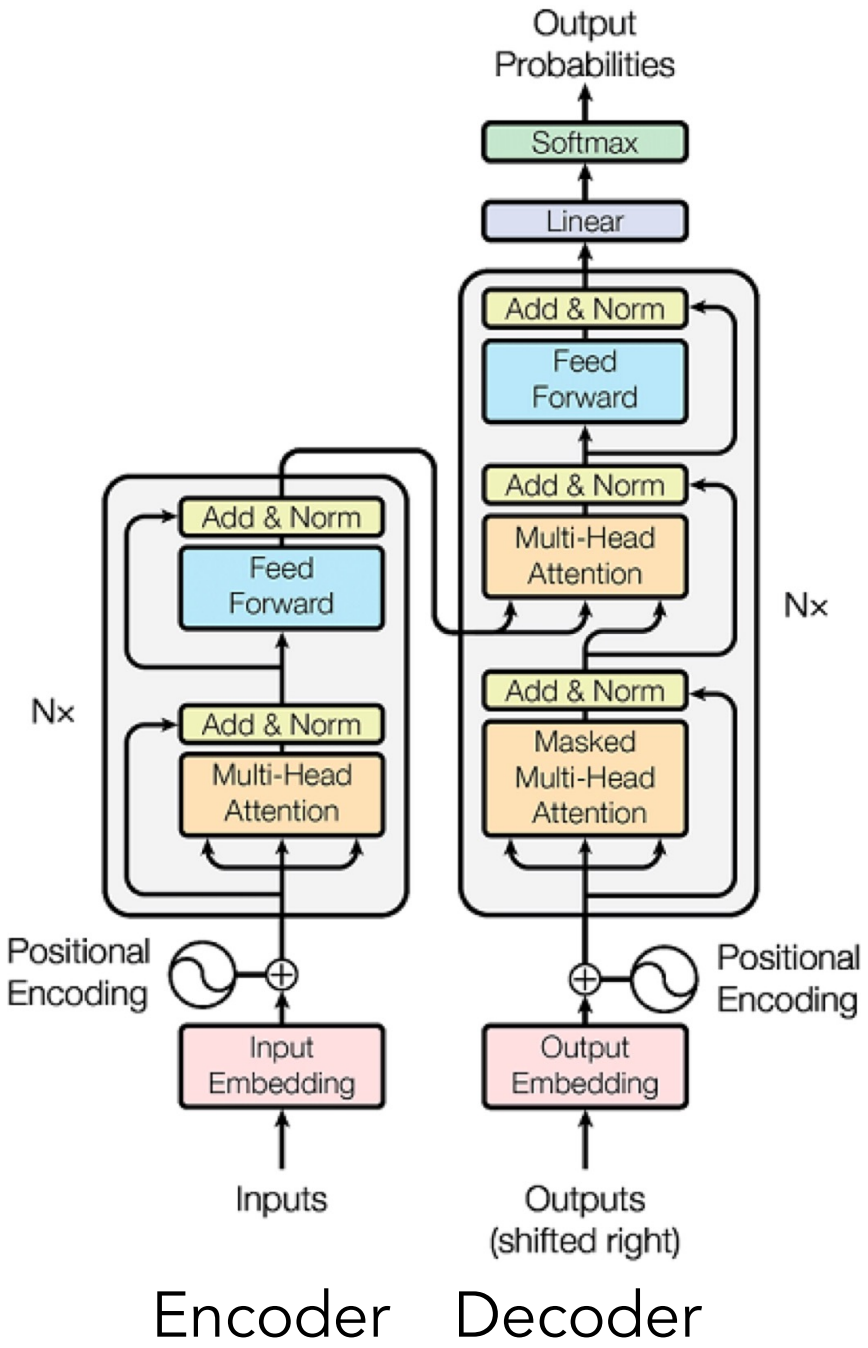


Transformers

- Encoder: Multi-headed self-attention
- Decoder
  - Masked self-attention

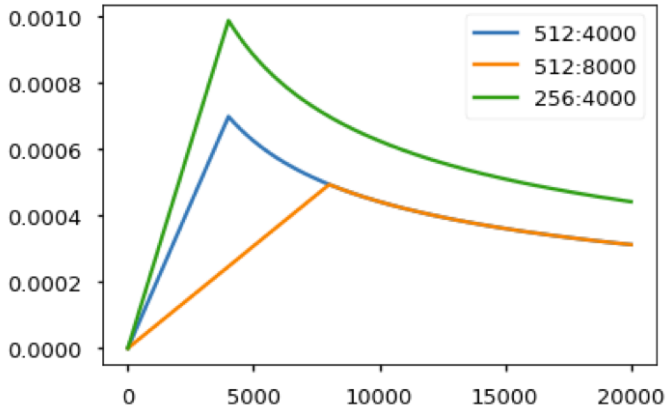


  - Cross attention
    - queries: previous decoder layer
    - keys/values: output of encoder
  - Autoregressive decoding

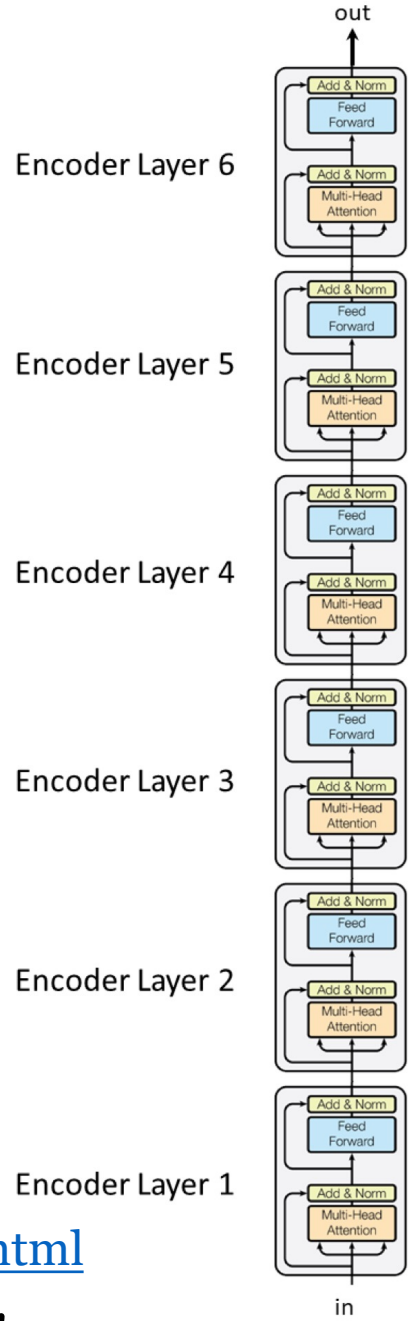


Transformers

- Stacked into multi-layers
- For language, input embedding is subwords
  - Byte-pair encoding (BPE) / Word pieces
- Other training details:
  - Learning rate with warmup and decay



  - Label smoothing: one-hot vector + noise



The Annotated Transformer http://nlp.seas.harvard.edu/2018/04/03/attention.html

A Jupyter notebook which explains how Transformer works line by line in PyTorch!

Transformers are used for everything!

10 Novel Applications using Transformers [DL]

Transformers have had a lot of success in training neural language models. In the past few weeks, we've seen several trending papers with code applying Transformers to new types of task:

🏞️ **Transformer for Image Synthesis** - 🔗 Esser et al. (2020)
🎱 **Transformer for Multi-Object Tracking** - 🔗 Sun et al. (2020)
🎶 **Transformer for Music Generation** - 🔗 Hsiao et al. (2021)
💃 **Transformer for Dance Generation with Music** - 🔗 Huang et al. (2021)
🔮 **Transformer for 3D Object Detection** - 🔗 Bhattacharyya et al. (2021)
🗿 **Transformer for Point-Cloud Processing** - 🔗 Guo et al. (2020)
⏰ **Transformer for Time-Series Forecasting** - 🔗 Lim et al. (2020)
👁️ **Transformer for Vision-Language Modeling** - 🔗 Zhang et al. (2021)
🛣️ **Transformer for Lane Shape Prediction** - 🔗 Liu et al. (2020)
🌠 **Transformer for End-to-End Object Detection** - 🔗 Zhu et al. (2021)

PapersWithCode newsletter (1/20/2021)
https://paperswithcode.com/newsletter/3

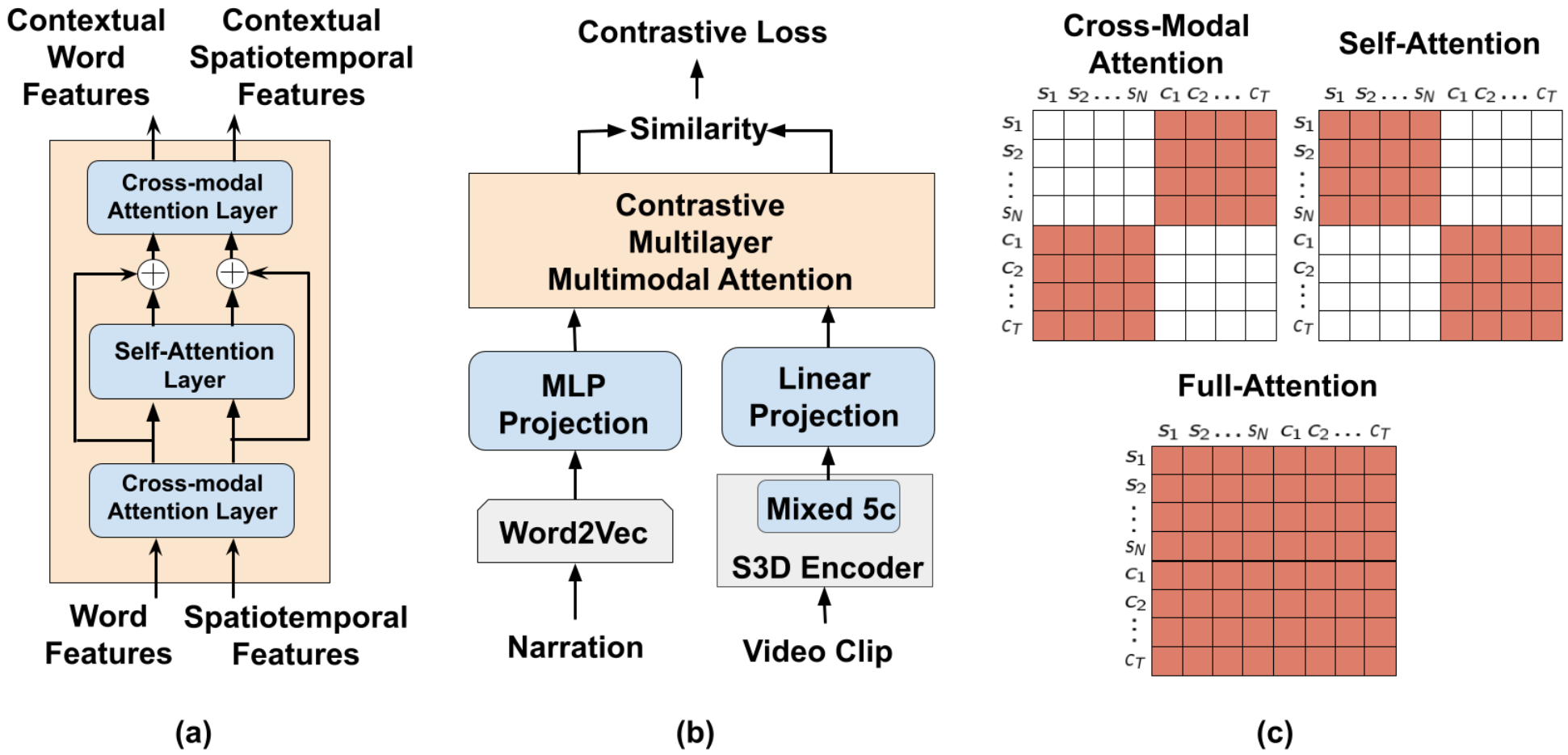
Grounding of interactions in videos

Phrase Grounding (prior work)

Narrated Interaction Localization (this paper)

Look at What I am Doing: Self-Supervised Spatial Grounding of Narrations in Instructional Videos
[Tan et al., NeurIPS 2021]

Attention for grounding of interactions



Look at What I am Doing: Self-Supervised Spatial Grounding of Narrations in Instructional Videos [Tan et al., NeurIPS 2021]

Next week

- Monday: Paper presentations and discussions
  - (Sihui) Bottom-up and top-down attention
  - (Xiaohao) FiLM: Feature-wise Linear Modulation
- Wednesday: Pretraining with Transformers