LG Advanced Data Scientists Program
Deep Learning

[4: Attention and Transformer]

Prof. Sungroh Yoon

Electrical & Computer Engineering | Seoul National University

(last compiled at 21:53:00 on 2020/02/09)

Outline

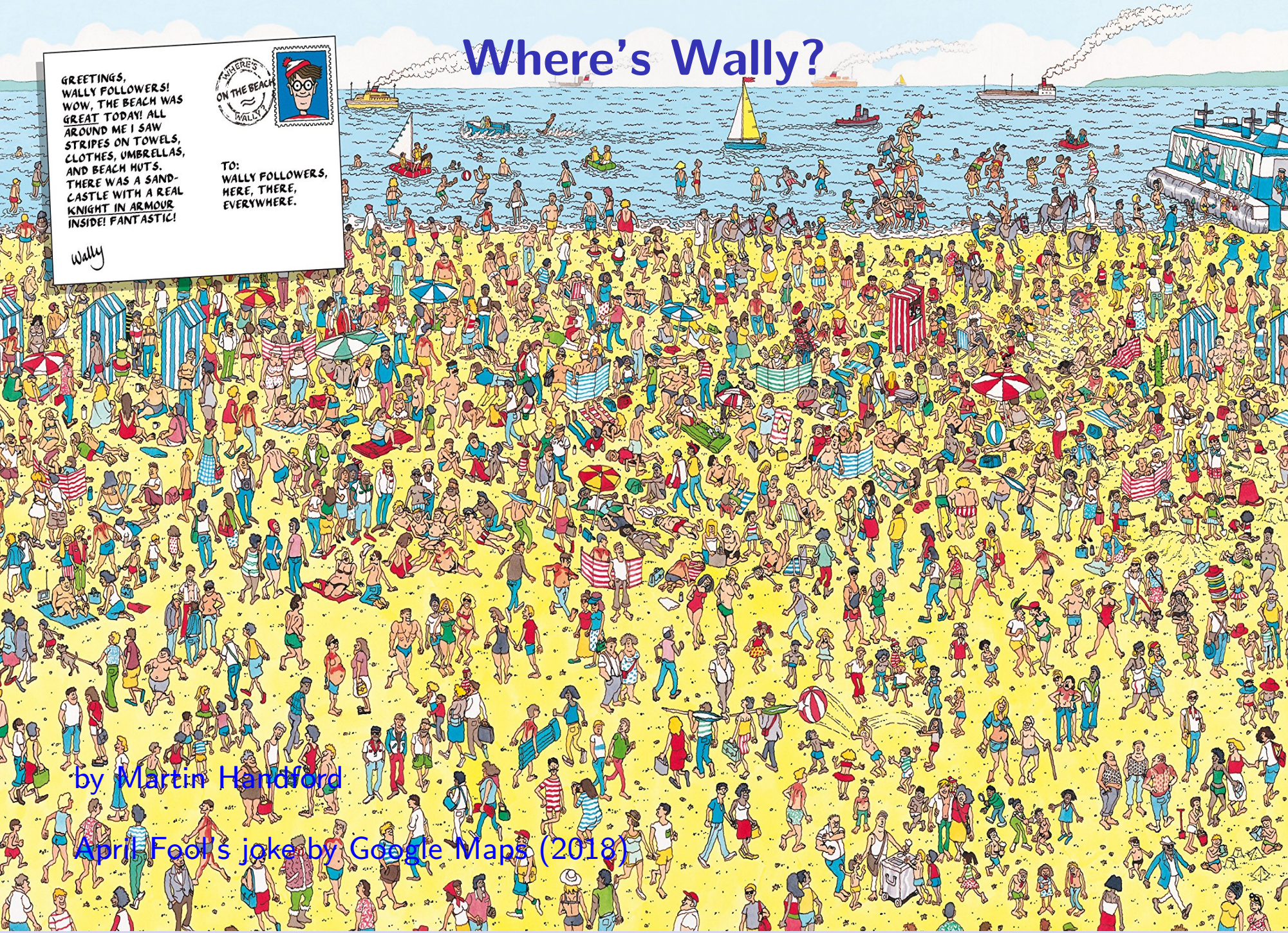
References

- online resources:
  - *A Tutorial on Attention in DL (ICML 2019)* ▸ Link
  - *Stanford CS224n: NLP with Deep Learning* ▸ Link
  - *Attn: Illustrated Attention* ▸ Link
  - *Attention? Attention!* ▸ Link
  - *The Annotated Transformer* ▸ Link
  - *Attention and Augmented RNNs* ▸ Link
  - *The Illustrated Transformer* ▸ Link

- note:
  - you should open this file in Adobe Acrobat to see animated images (other types of pdf readers will not work)

Outline

Where's Wally?



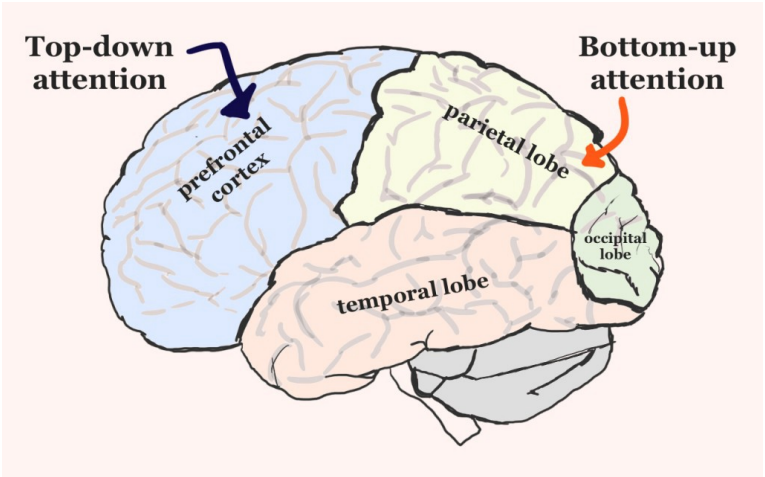
by Martin Handford

April Fool's joke by Google Maps (2018)

Attention

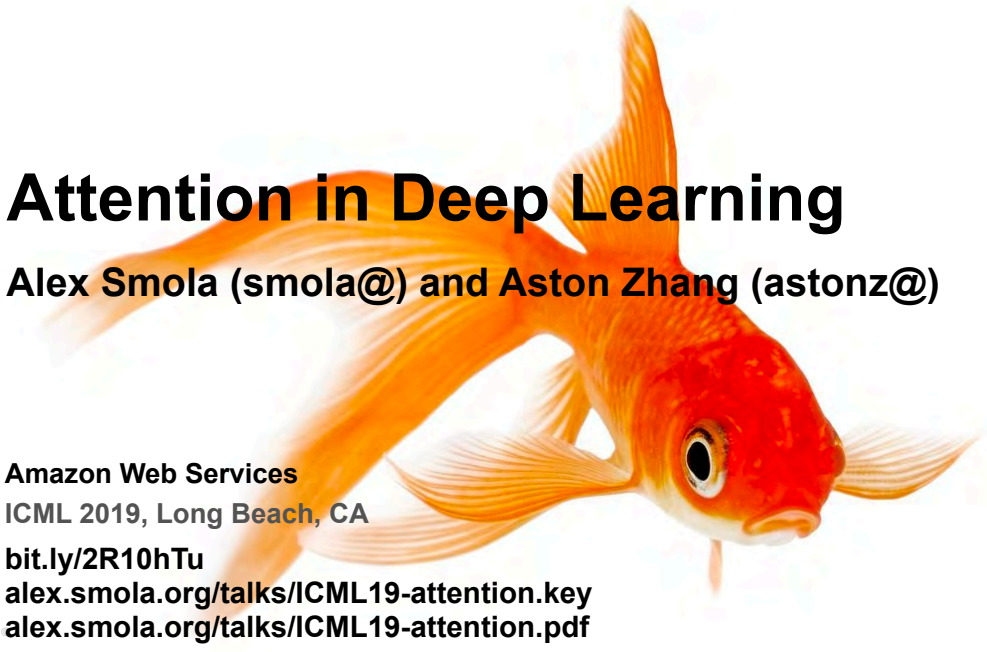
- literally (Google dictionary)
  - `n. notice taken of someone or something; the regarding of someone or something as interesting or important.`

- biologically (Wikipedia)
  - behavioral and cognitive process of *selectively concentrating* on a discrete aspect of information, while *ignoring other* perceivable information
  - for _____ saving



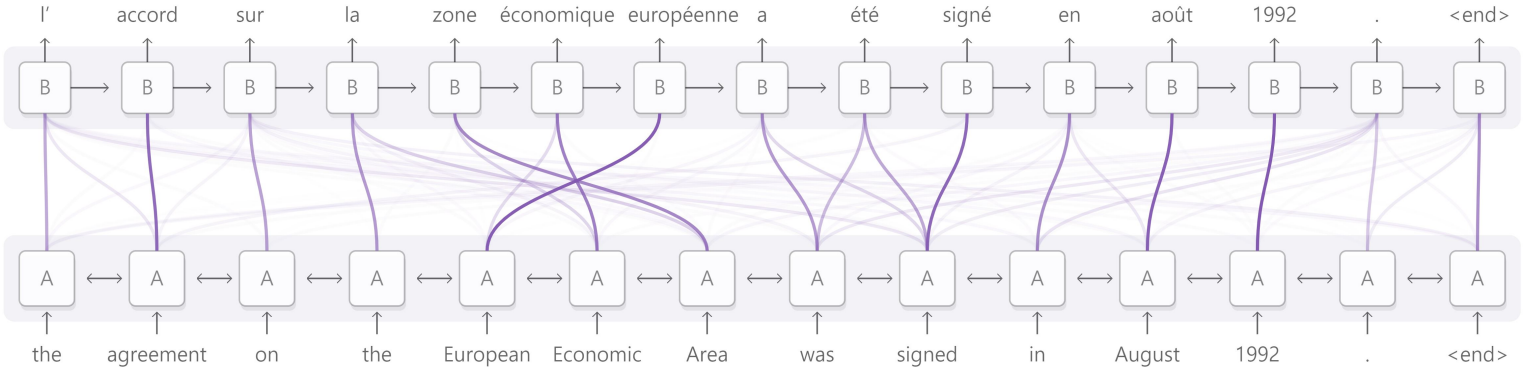
(source: MacKay, The science of staying focused, 2017)

- in machine learning
  - an iterative process
  - applicable to input, memory, and representation
  - _____ trainable (soft attention)
  - ICML 2019 tutorial: ▸ Link

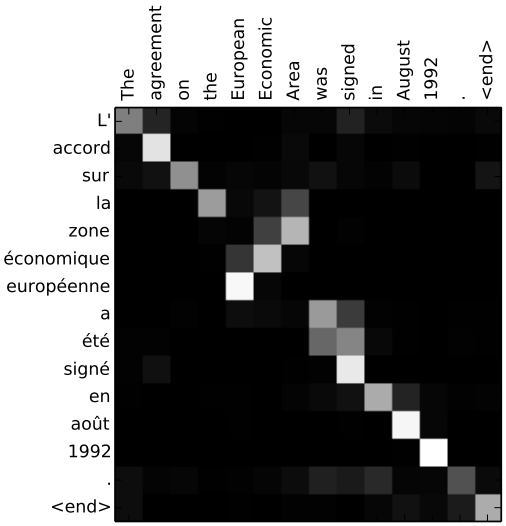


Attention in deep learning

- NLP (neural machine translation)



- attention $\Rightarrow$ learned _____ (between source and target sentences)



source: Attention and Augmented RNNs; Bahdanau et al. (2014).

Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*

- computer vision

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

(source: Xu et al.[1])

- trend + key papers

  2014 recurrent models of visual attention[2]

  2014–15 attention in neural machine translation[3]

  2015–16 attention-based RNN/CNN in NLP

  2017 self-attention[4]

---

[1] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057

[2] Mnih, V., Heess, N., Graves, A., et al. (2014). Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212

[3] Bahdanau et al. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*

[4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017a). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008

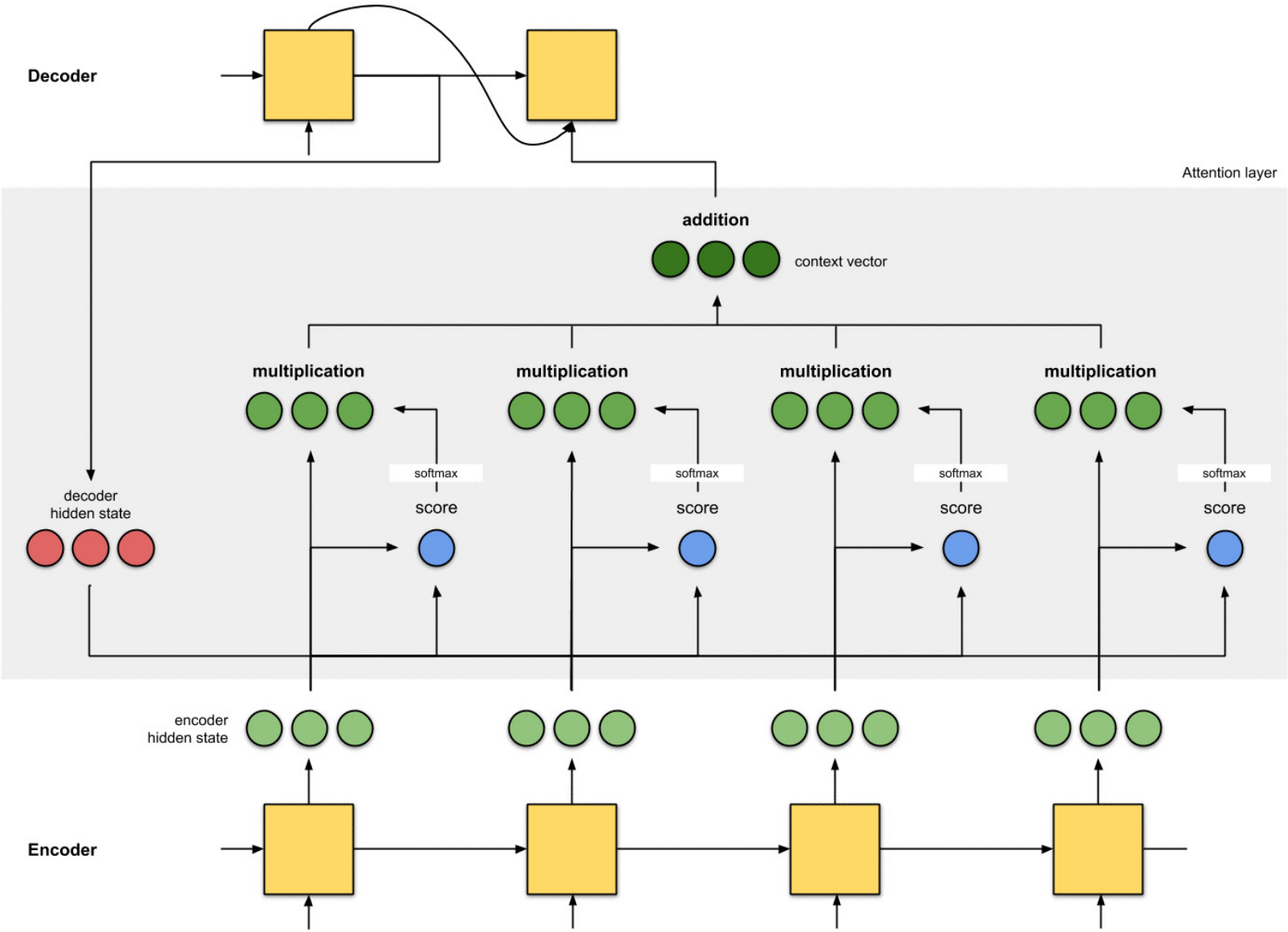
Recall: sequence-to-sequence model

- basic components
  - encoder RNN + decoder RNN
- seq2seq (w/o attention)
  - encoder RNN: input encoding + context vector computing

  $\Rightarrow$ may be overloaded $\Rightarrow$ suboptimal performance
- seq2seq with attention
  - encoder RNN: input encoding
  - attention: context vector computing

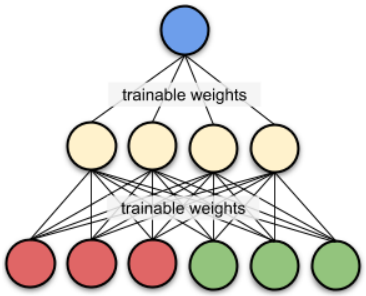
  $\Rightarrow$ offloads encoder RNN $\Rightarrow$ better performance

- seq2seq with attention ▸ source



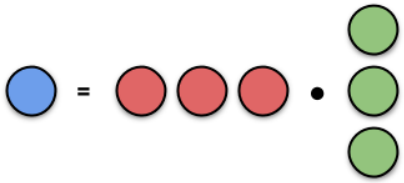
Attention scoring

Additive / Concat



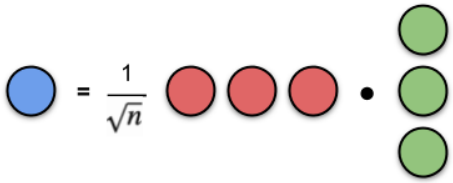
$$a = \mathbf{v}^\top \tanh(\mathbf{W}[\mathbf{s}; \mathbf{h}])$$

Dot product



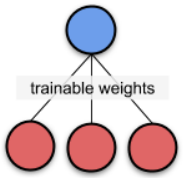
$$a = \mathbf{s}^\top \mathbf{h}$$

Scaled dot product



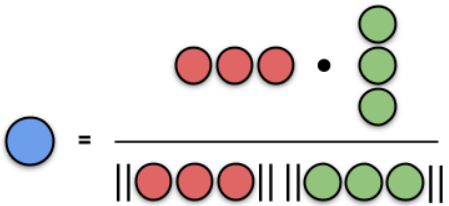
$$a = \frac{1}{\sqrt{n}} \mathbf{s}^\top \mathbf{h}, \text{ where } n = ||\mathbf{h}||$$

Location-based



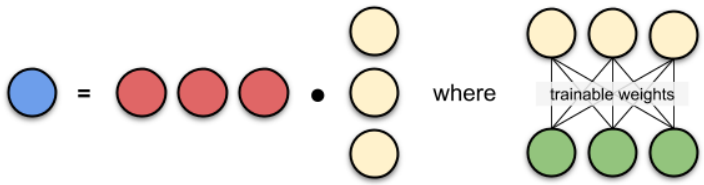
$$a = \text{softmax}(\mathbf{W}\mathbf{s})_i$$

Cosine similarity



$$a = \frac{\mathbf{s}^\top \mathbf{h}}{||\mathbf{s}|| \cdot ||\mathbf{h}||}$$

General



$$a = \mathbf{s}^\top \mathbf{W} \mathbf{h}$$

(source: Attention? Attention!, Attn: Illustrated Attention)

Key-value description of attention

- three types of vectors
  - query = (projected) decoder hidden state
  - key = (projected) encoder hidden state (for attn weight computation)
  - value = (projected) encoder hidden state (for context vector buildup)

- outcome
  - context vector = weighted sum of ____s

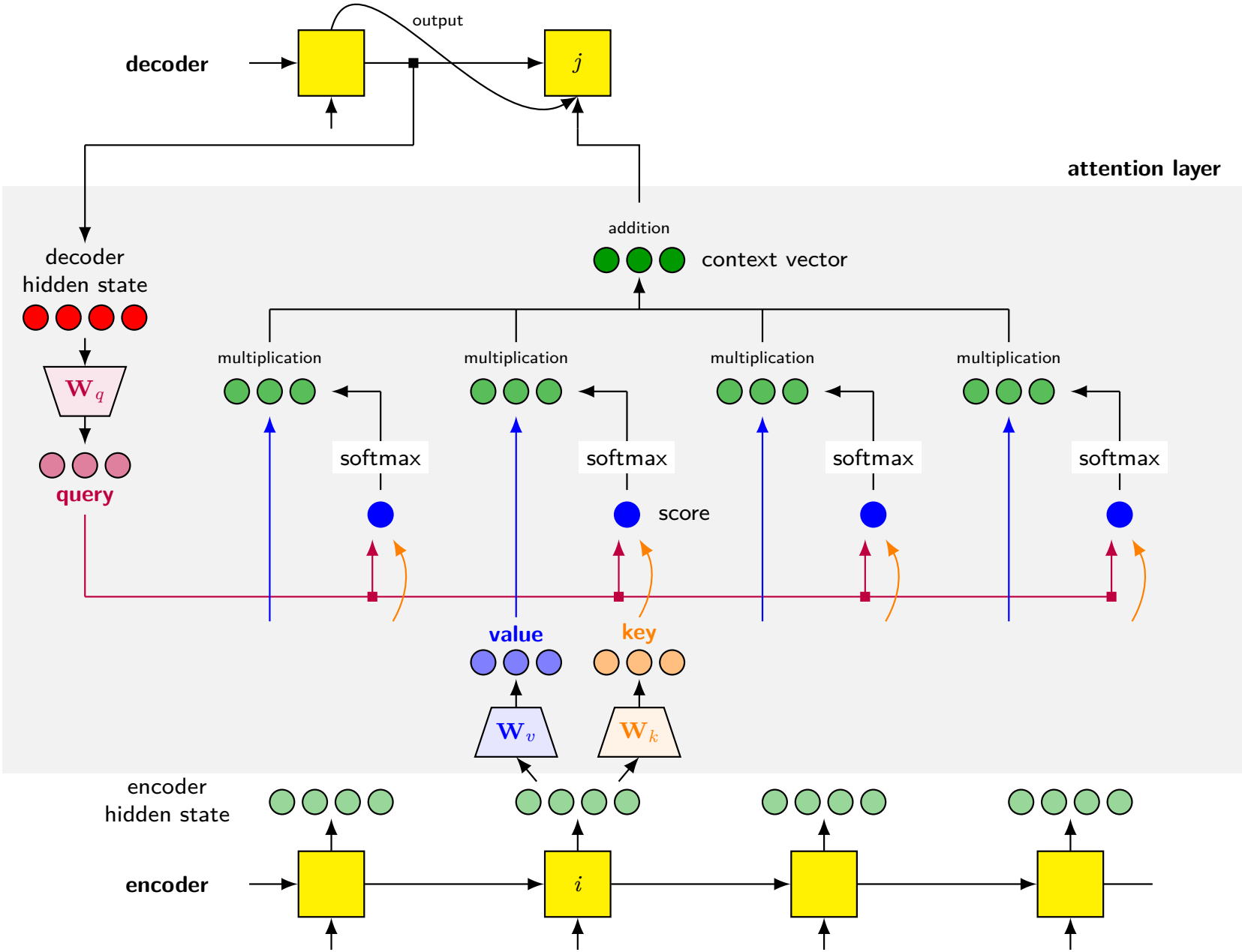
    ↑

    each weight = function of (query, key)

- idea: separation of encoder hidden state into
  - attention information and content information for context vector

    ↑ key

    ↑ value


output
decoder
j
attention layer
addition
context vector
decoder
hidden state
multiplication
multiplication
multiplication
multiplication
$\mathbf{W}_q$
softmax
softmax
softmax
softmax
query
score
value
key
$\mathbf{W}_v$
$\mathbf{W}_k$
encoder
hidden state
encoder
i

Outline

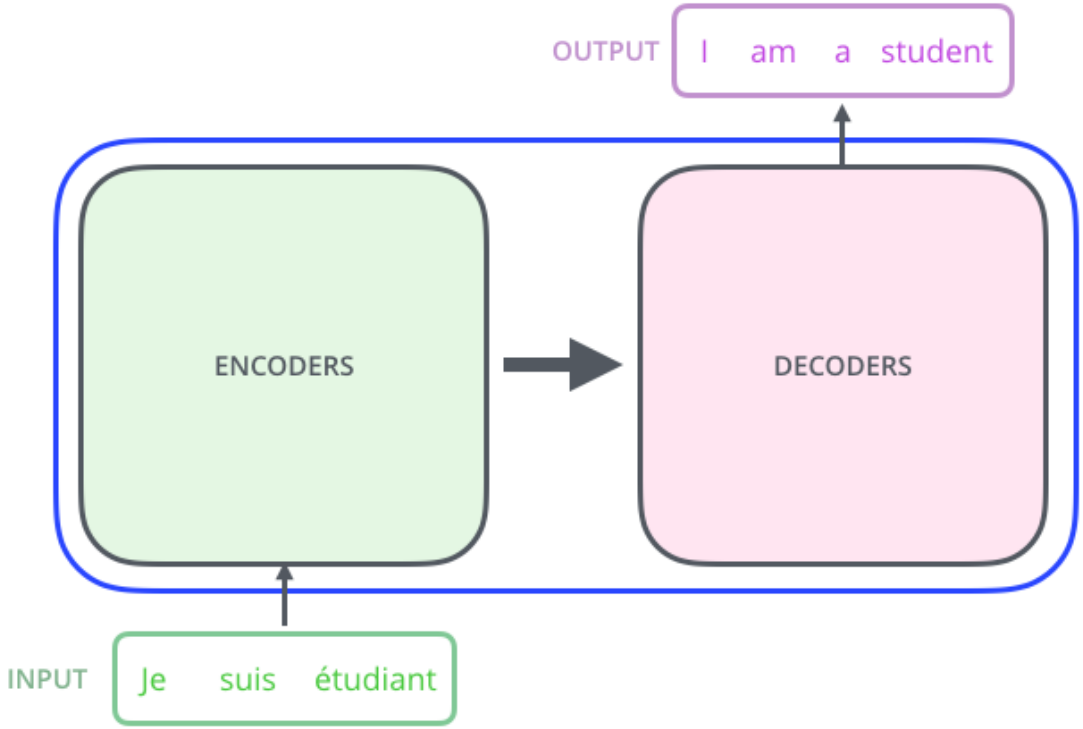
Transformer

A high-level look

- a sequence-to-sequence model
  - input sequence (arbitrary length) $\longrightarrow$ output sequence (arbitrary length)

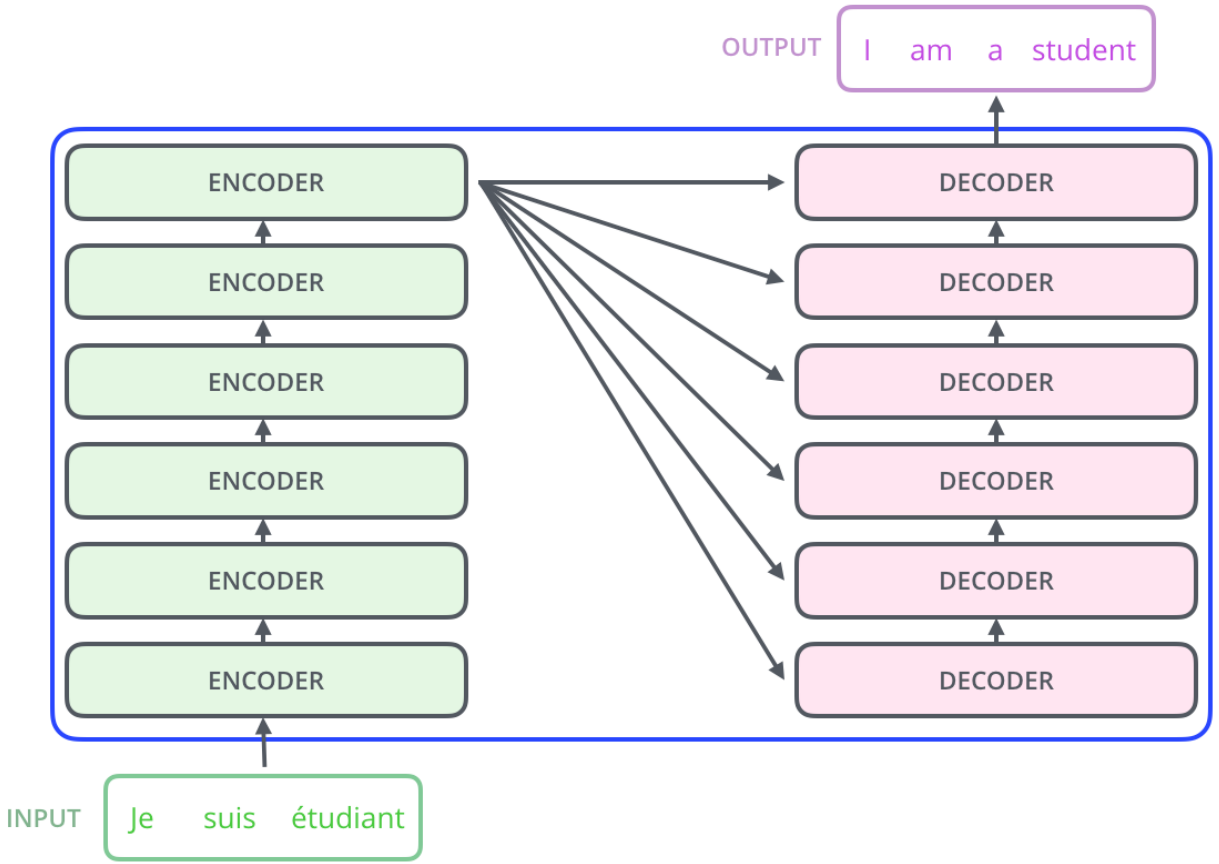
(source: The Illustrated Transformer)

- encoder-decoder architecture
  - an encoding component + a decoding component
  - and connections between them



(source: The Illustrated Transformer)

- encoding component: a stack of encoders[5]
- decoding component: a stack of decoders of the same number

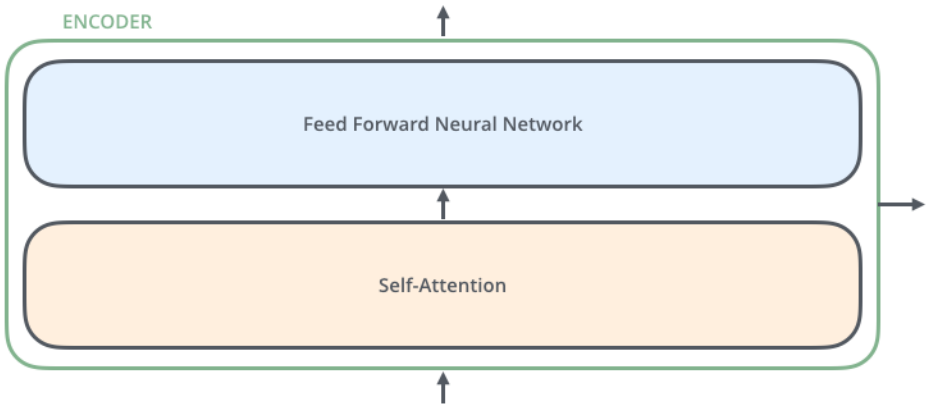


(source: The Illustrated Transformer)

[5] the original transformer paper stacks six of them on top of each other; there is nothing magical about the number six, and one can definitely experiment with other arrangements.

Encoder

- encoders: all identical in structure (yet they do not share weights)
  - each one is broken down into two sub-layers:

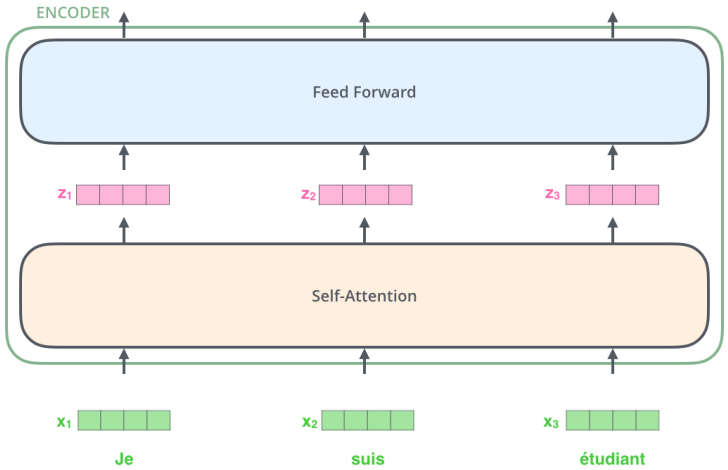


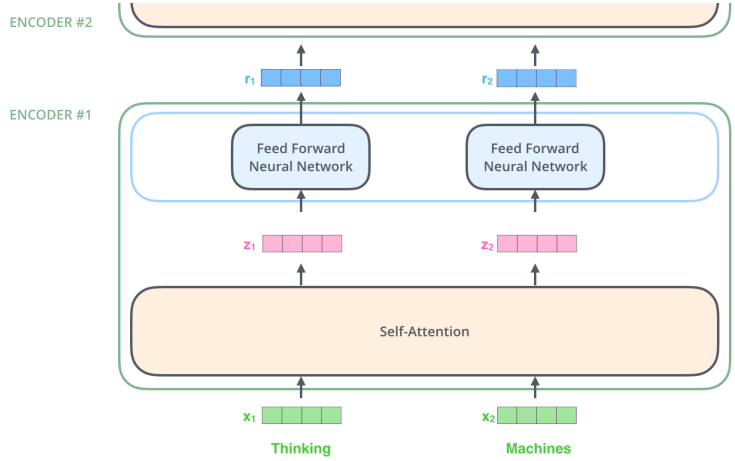
(source: The Illustrated Transformer)

- self-attention layer
  - helps the encoder look at other words in the input sentence as it _____ a specific word

- feed-forward neural net layer
  - the exact same feed-forward net is independently applied to each position

- word embedding
  - we begin by turning each input word into a vector
  - only happens in the _____-most encoder
  - embedding vector size in the transformer paper: 512

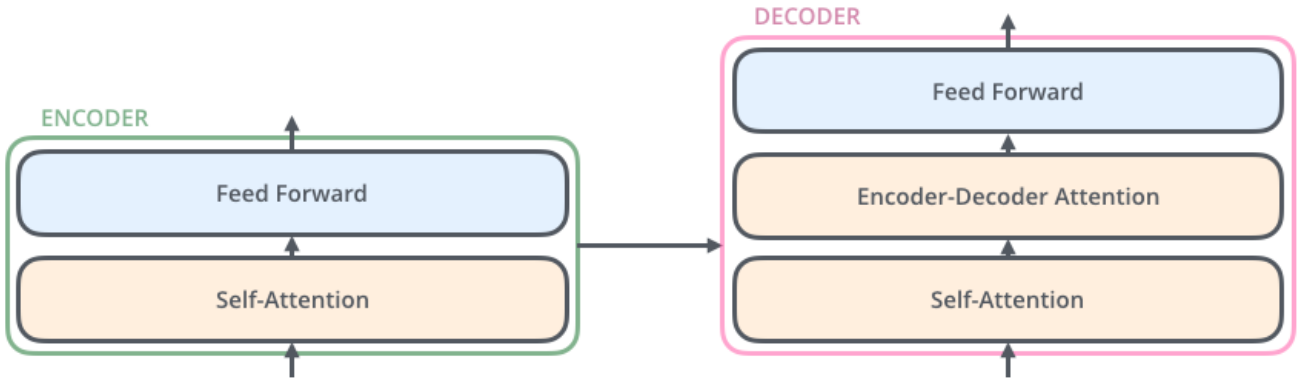




(source: The Illustrated Transformer)

Decoder

- decoder: has three sub-layers
  - feed-forward layers
  - encoder-decoder attention layer
  - self-attention layer



(source: The Illustrated Transformer)

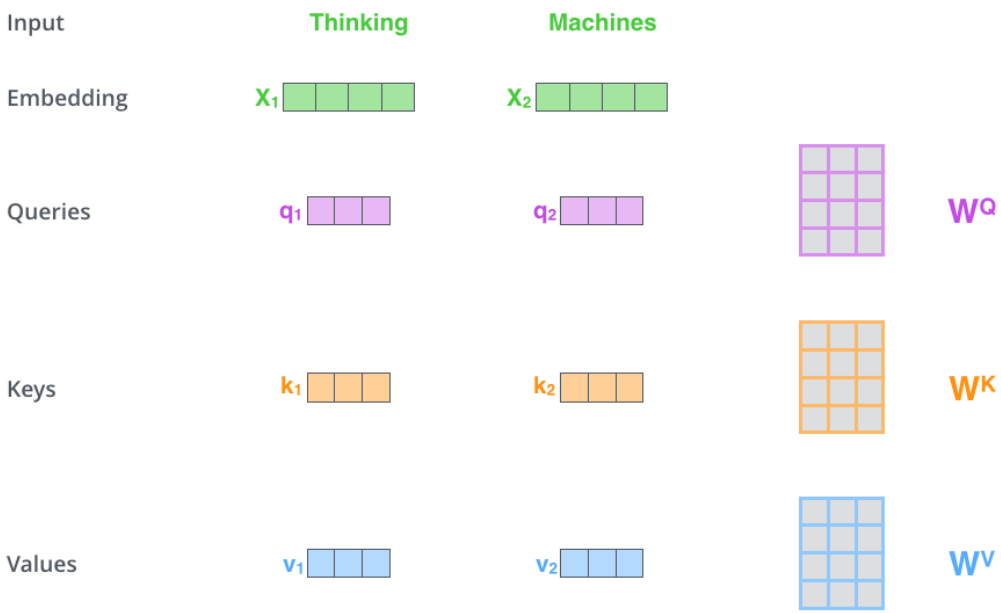
- encoder-decoder attention layer
  - helps the decoder focus on relevant parts of the _____ sentence

    (similar to what attention does in seq2seq models)

Outline

Step 1

- create three vectors from each of the encoder's input (embedding) vectors
  - query, key, value
  - how? multiply embedding vectors by three trainable matrices



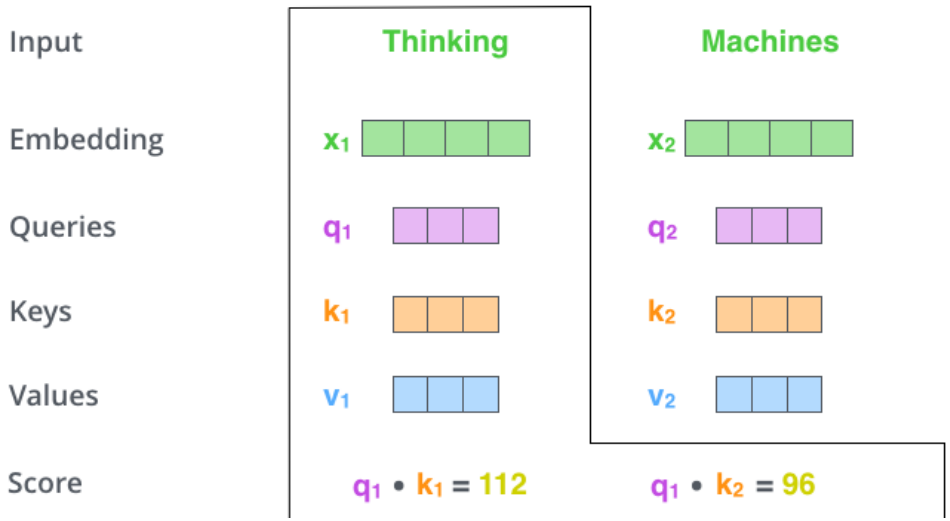
(source: The Illustrated Transformer)

- size of query/key/value vectors: _____ than embedding
  - in the original paper: 64 vs 512

Step 2

- calculate a self-attention score for each word
  - the score determines how much focus to place on other parts of the input as we encode a word at a certain position

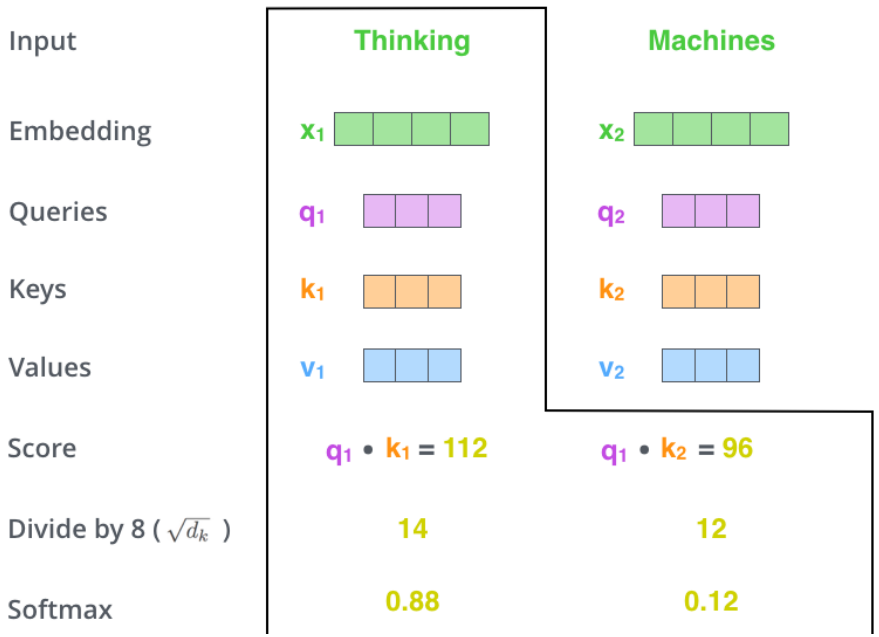
- how? take a dot product:
  - score = (query vector) · (___ vector of the word we're scoring)



(source: The Illustrated Transformer)

Step 3

- normalize attention scores
  - (3a) divide each score by $\sqrt{\text{key vector size}} = \sqrt{d_k}$ (original paper: $\sqrt{64} = 8$)
    - leads to having more stable gradients
  - (3b) softmax operation
    - gives positional _____ (which sum to 1)



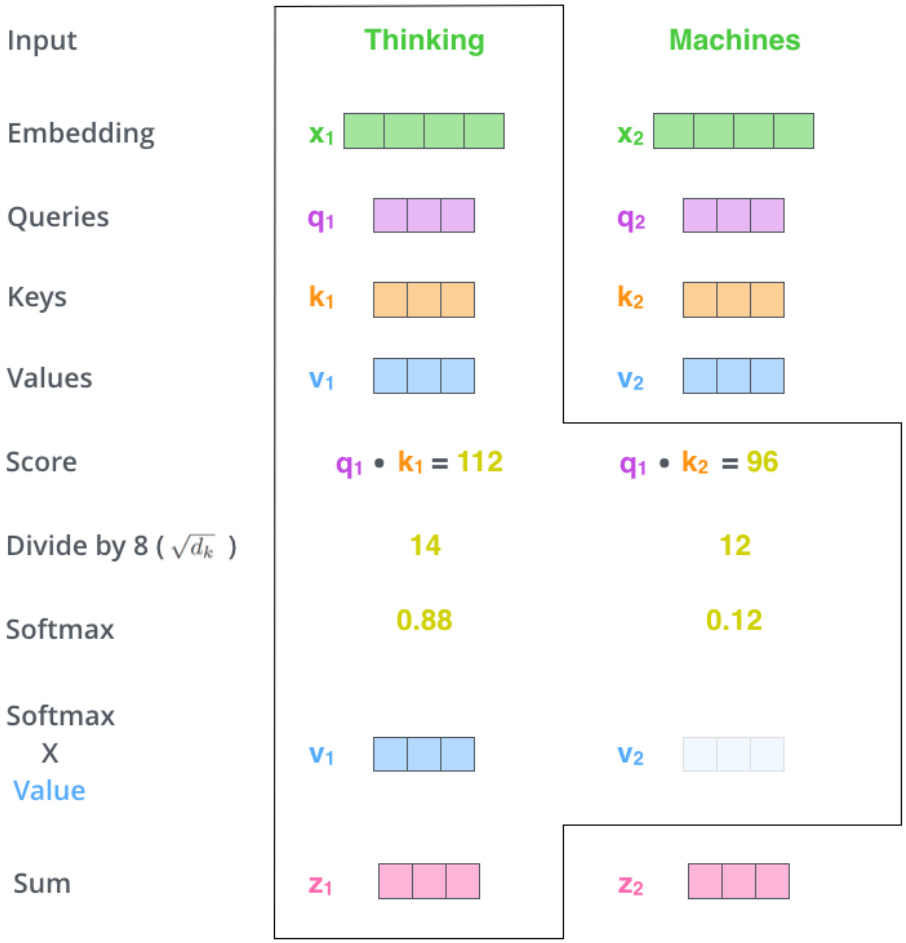
(source: The Illustrated Transformer)

Step 4

- calculate the output of self-attention layer

↑

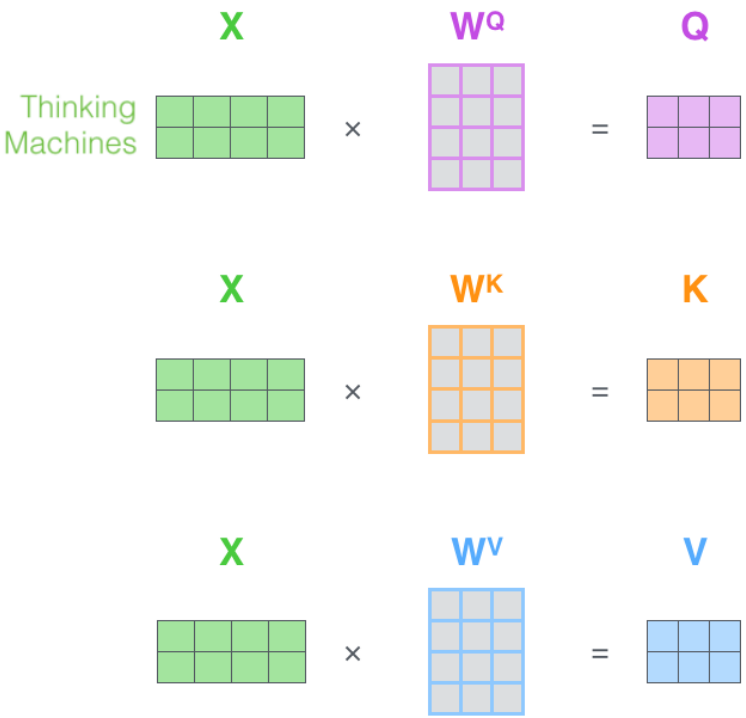
weighted sum of value vectors

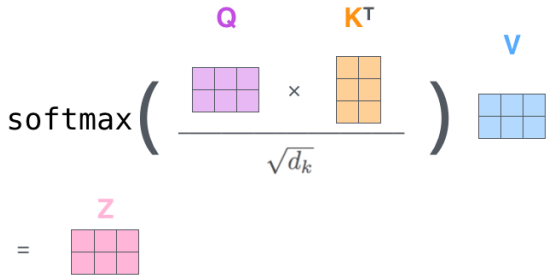


(source: The Illustrated Transformer)

Matrix calculation of self-attention

- can process all embedding vectors simultaneously
  - by stacking them as a _____



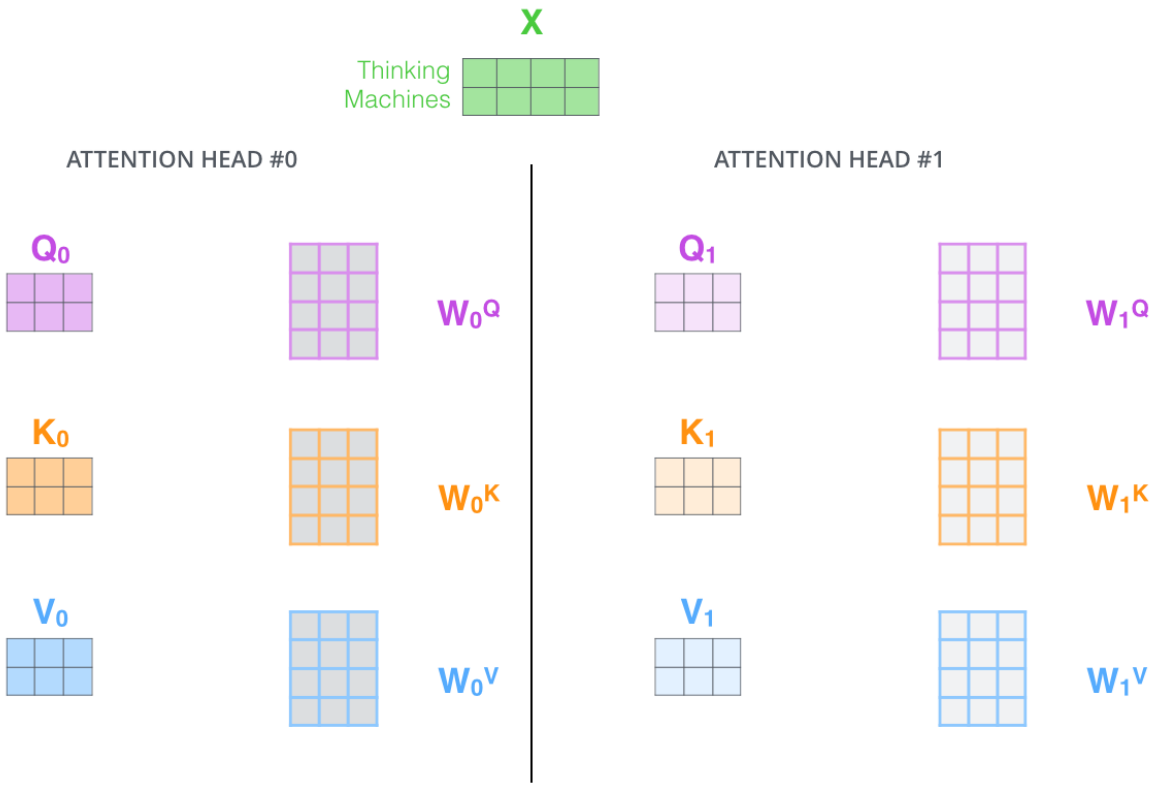


(source: The Illustrated Transformer)

Outline

Multi-head attention

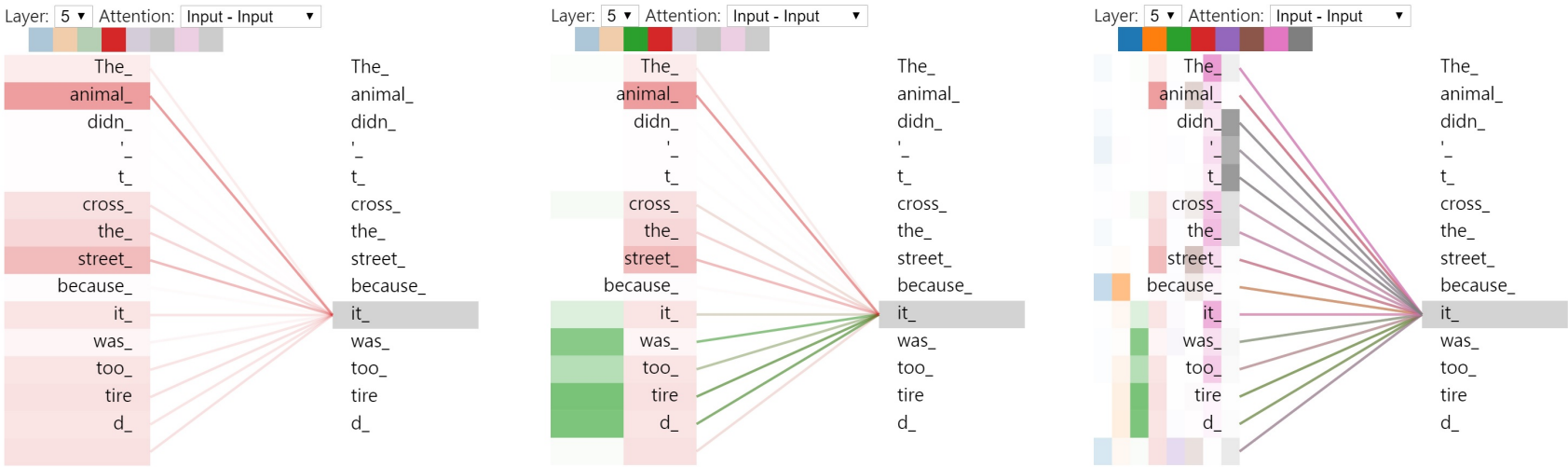
- further refinement of self-attention layer: use **multiple** attention _____
  - expands the model's ability to focus on different positions
  - gives the attention layer multiple representation subspaces



(source: The Illustrated Transformer)

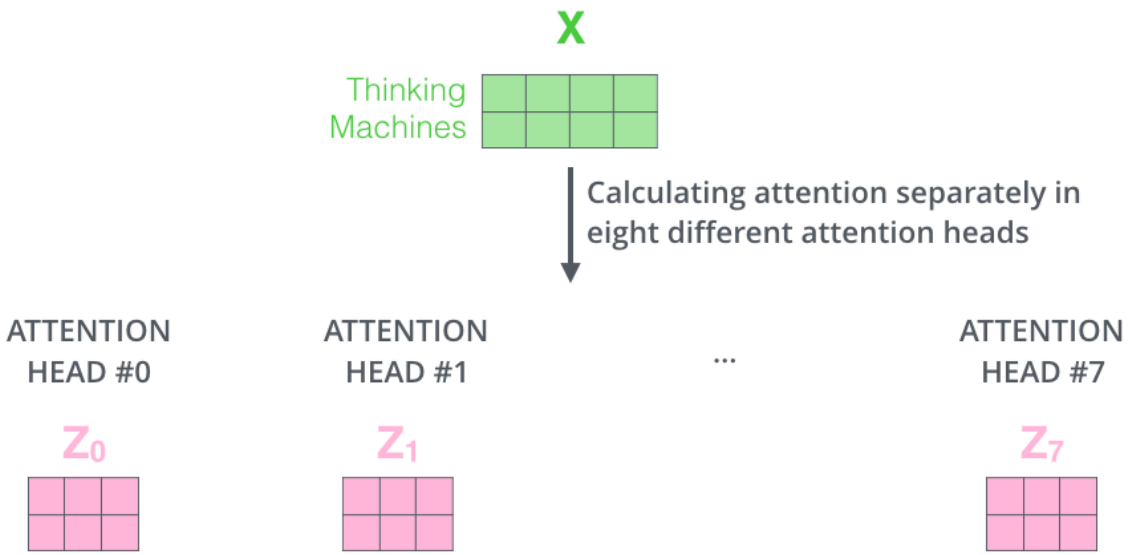
- different attention heads
  - focus differently as we encode the word "it"

e.g. 1 head, 2 heads, and 8 heads



(source: Tensor2Tensor ▸ code)

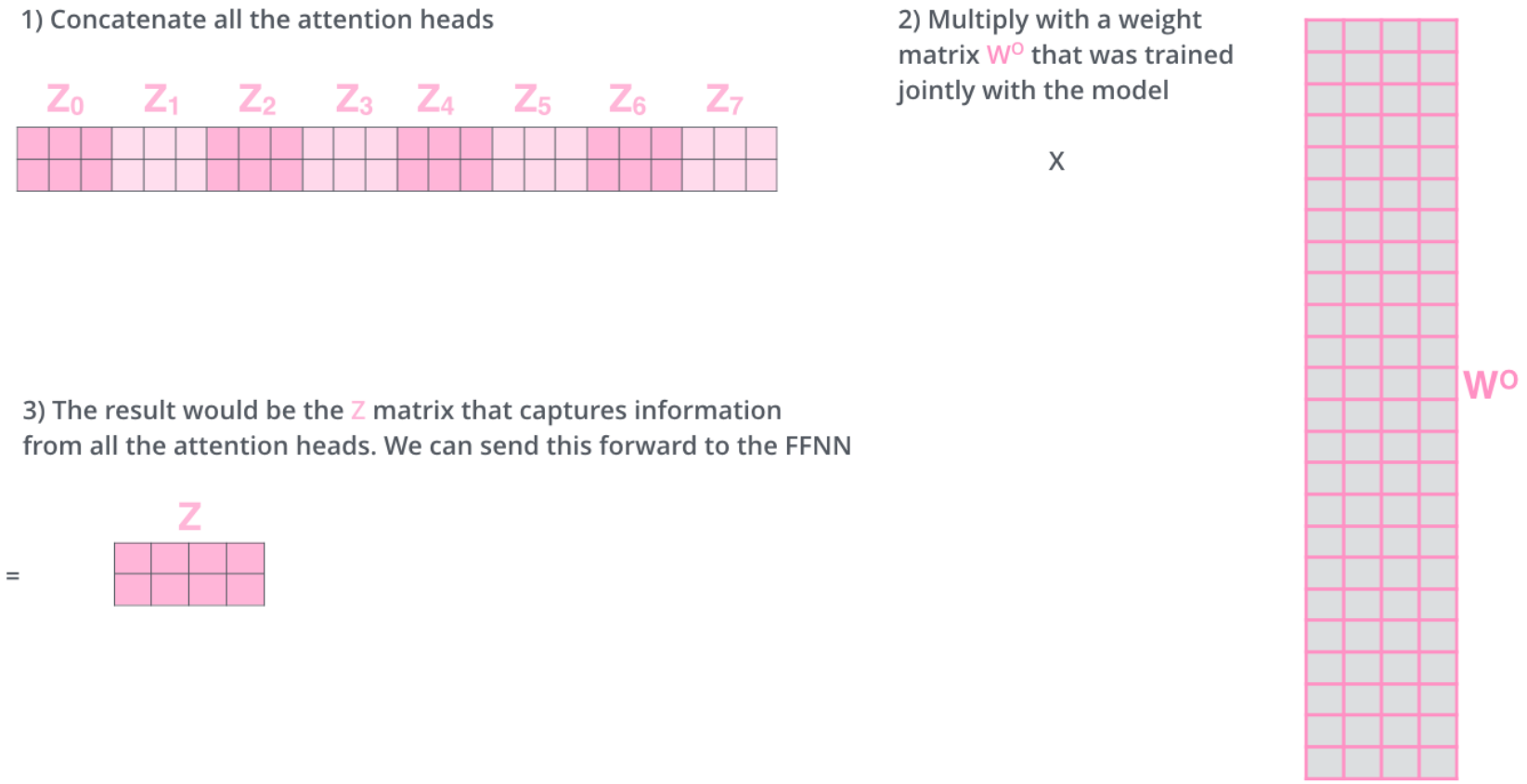
- the original transformer paper
  - 8 attention heads



(source: The Illustrated Transformer)

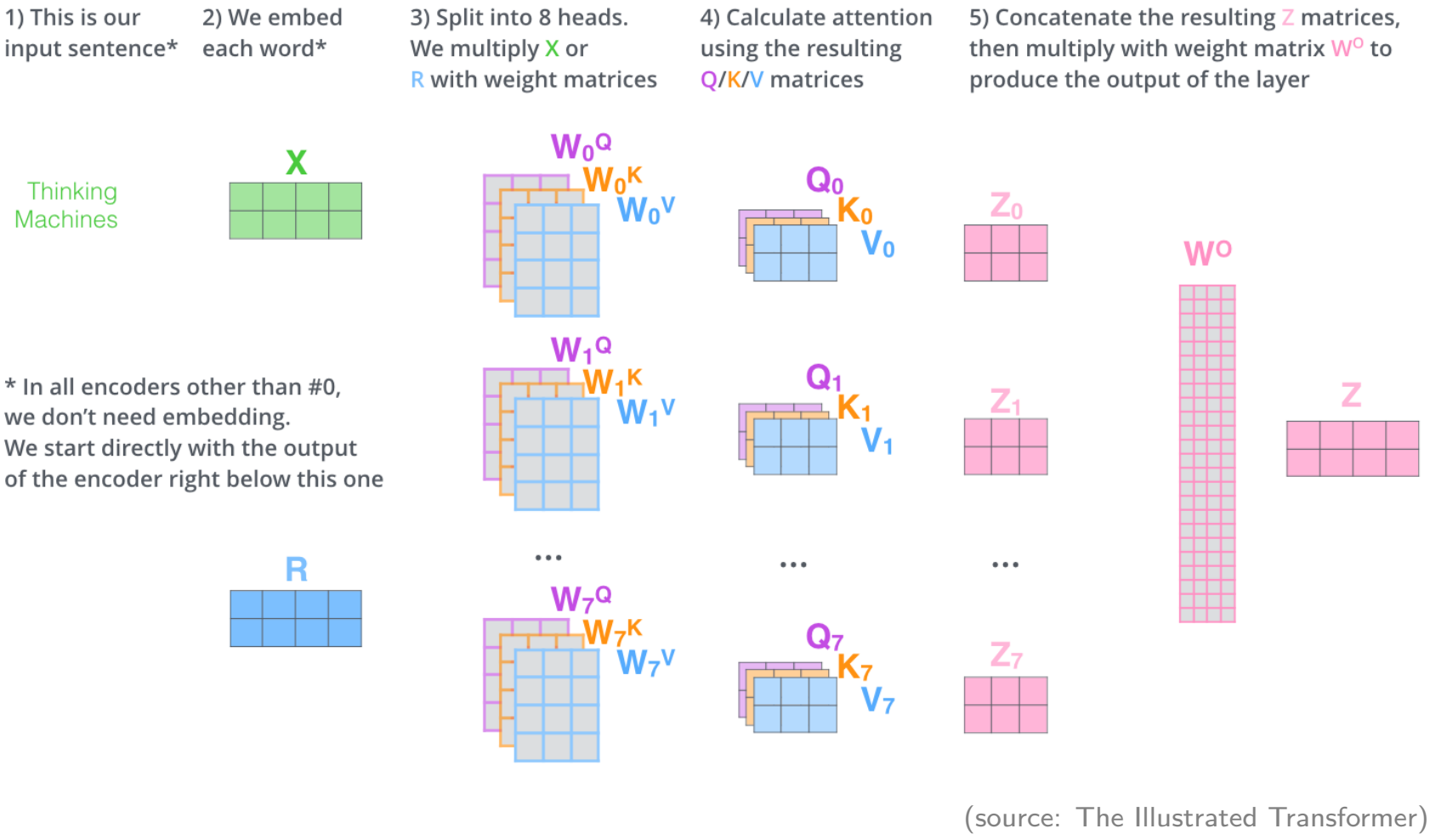
Condensing multi-head attention results

- concatenate heads and multiply with a learnable weight matrix



(source: The Illustrated Transformer)

Recap

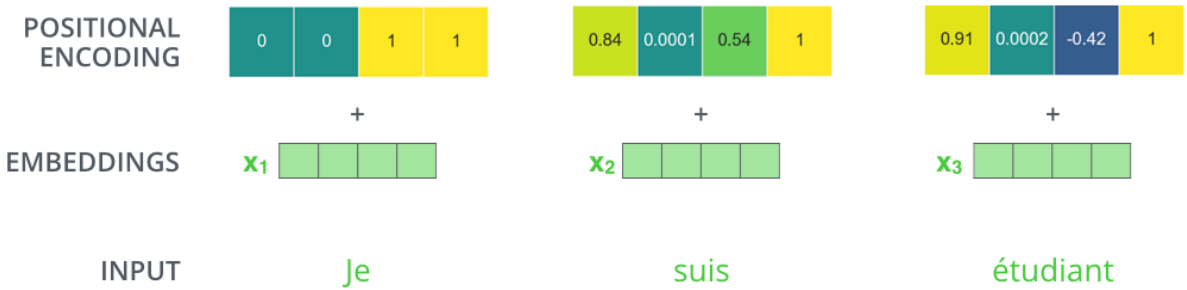


(source: The Illustrated Transformer)

Outline

Representing the order of words

- one thing missing from the model we have described so far
  - a way to represent the _____ of the words in the input sequence
- solution: add a vector ("positional encoding") to each input embedding



(source: The Illustrated Transformer)

- positional encoding vectors: follow a specific pattern the model learns

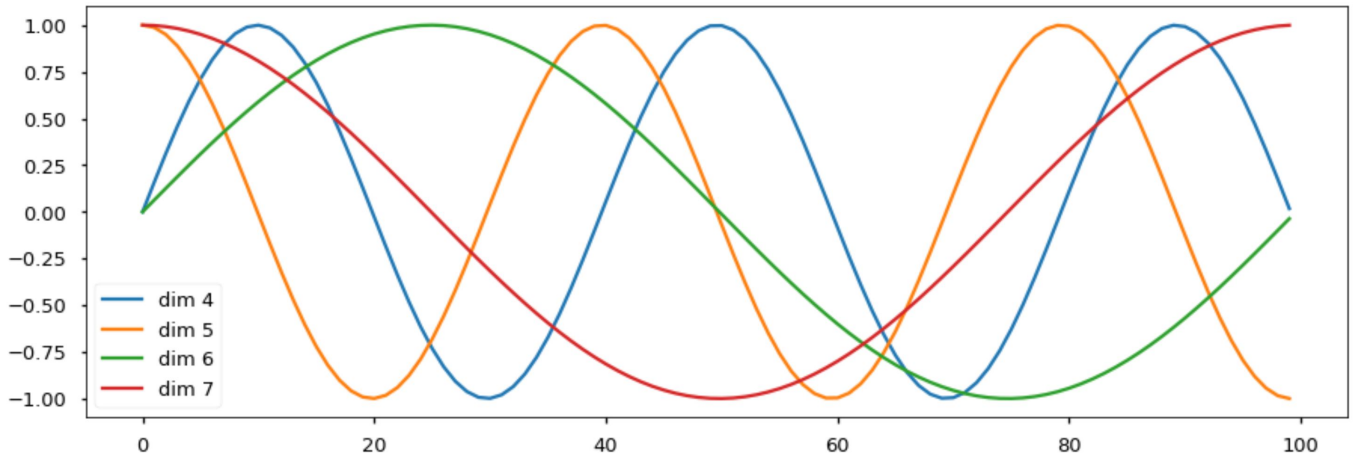
↑

helps the model to determine

  - position of each word or
  - distance between different words in input

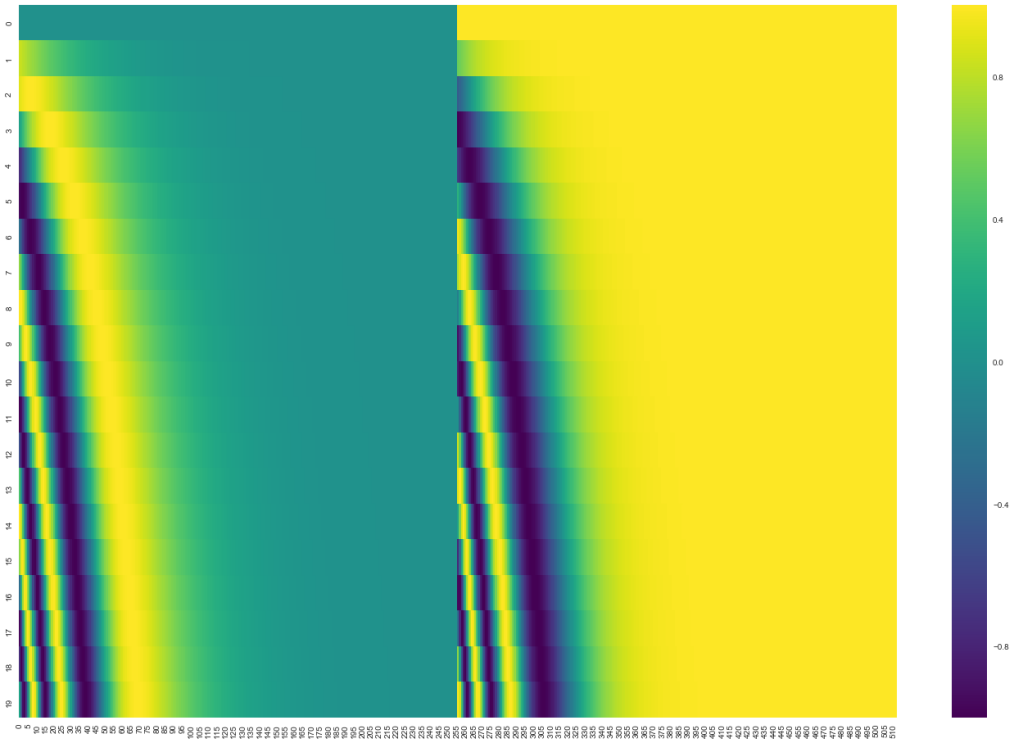
Positional patterns

- many choices for such a pattern: learned or fixed
- example: adding a *sine wave* based on position
  - frequency and offset of the wave: different for each dimension



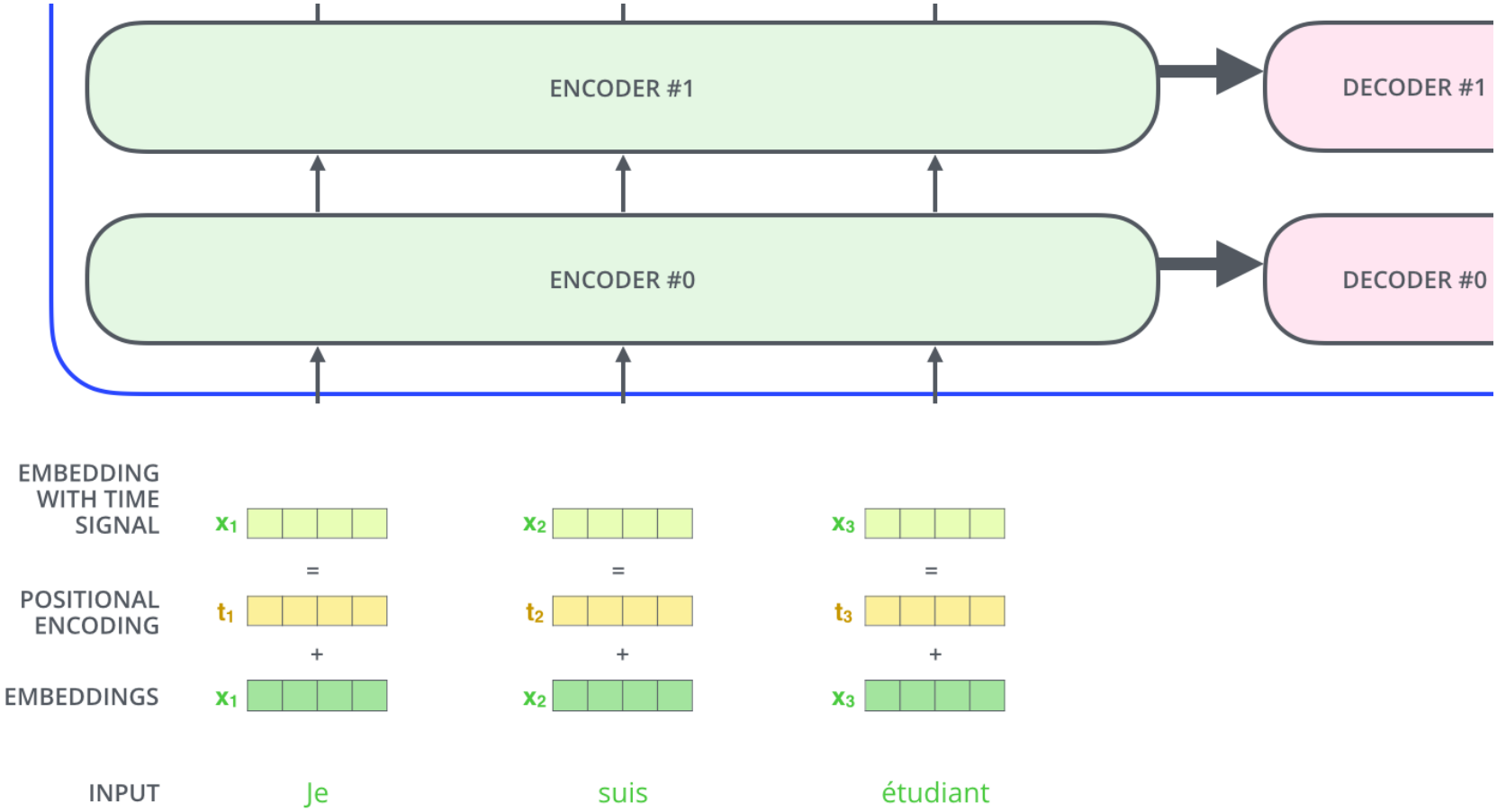
(source: The Annotated Transformer)

- positional encoding (PE) in the original transformer
  - 512 dimensions: 256 (sine) + 256 (cosine)



(source: The Illustrated Transformer)

- each row
  - a PE of a vector
- each column
  - PE dimension
- shown left
  - 20 words
  - 512 dimensions

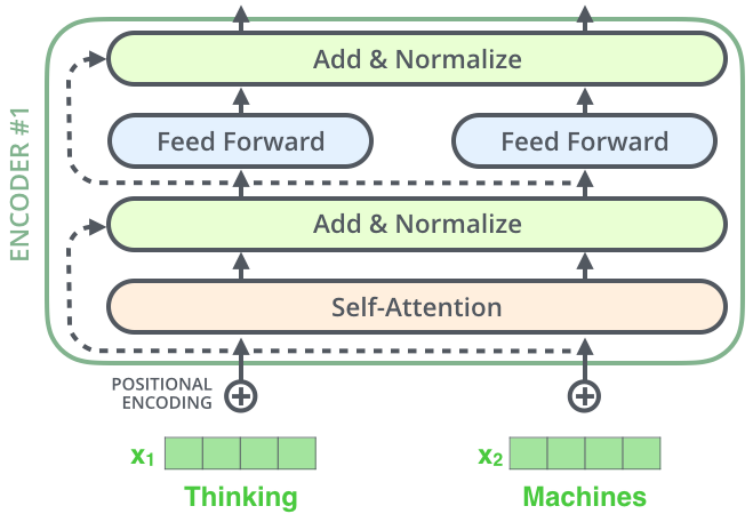


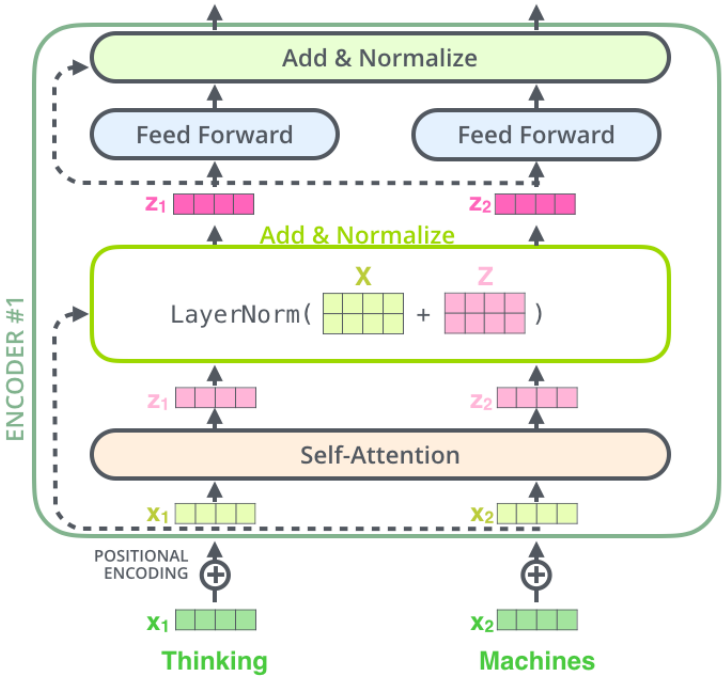
(source: The Illustrated Transformer)

Outline

Add and normalize

- each sublayer (self-attention and feed-forward) in encoder
  - (1) has a _____ connection around it and
  - (2) is followed by layer normalization[6]





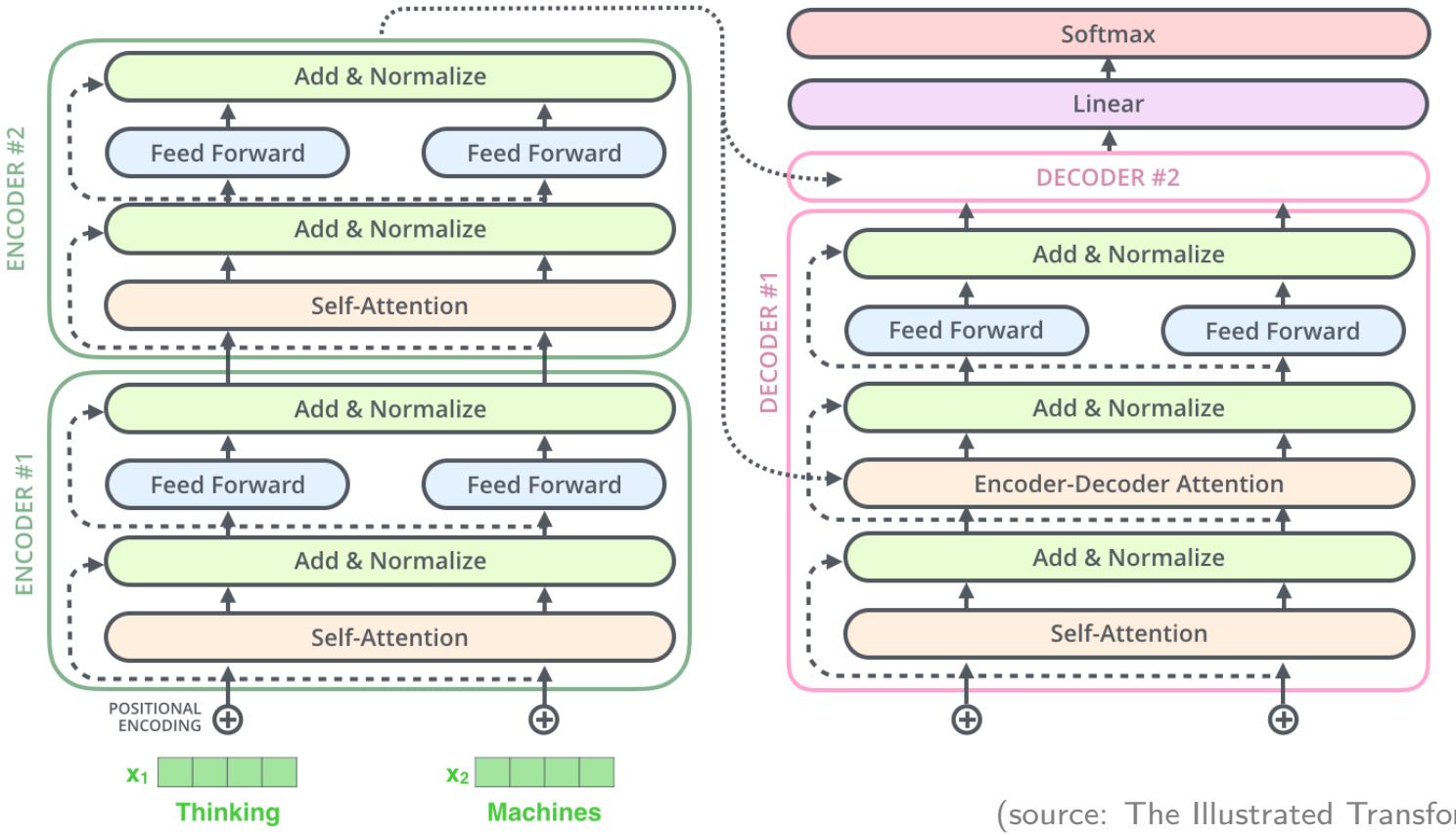
(source: The Illustrated Transformer)

[6] Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*

Outline

Decoder architecture

- overall, similar to encoder architecture
- difference #1: encoder-decoder attention (multi-head ____-attention)
  - helps the decoder focus on appropriate places in input sequence

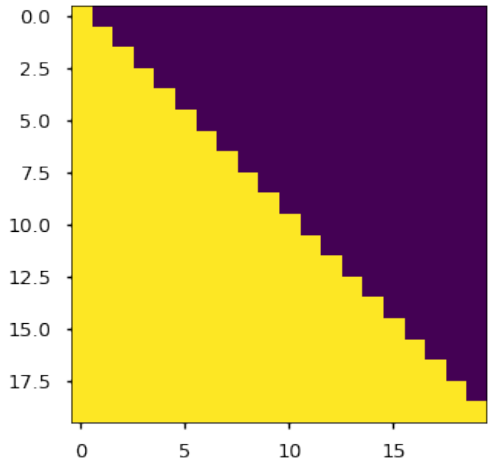


(source: The Illustrated Transformer)

- difference #2: modified self-attention (_____ multi-head self-attention)
  - for autoregressive decoding

$$P(y_1, y_2, \ldots, y_T \mid \boldsymbol{x}) = \prod_{t=1}^{T} p(y_t \mid y_1, y_2, \ldots, y_{t-1}, \boldsymbol{x})$$

  - need to prevent positions from attending to subsequent positions
  - example:



(source: The Annotated Transformer)

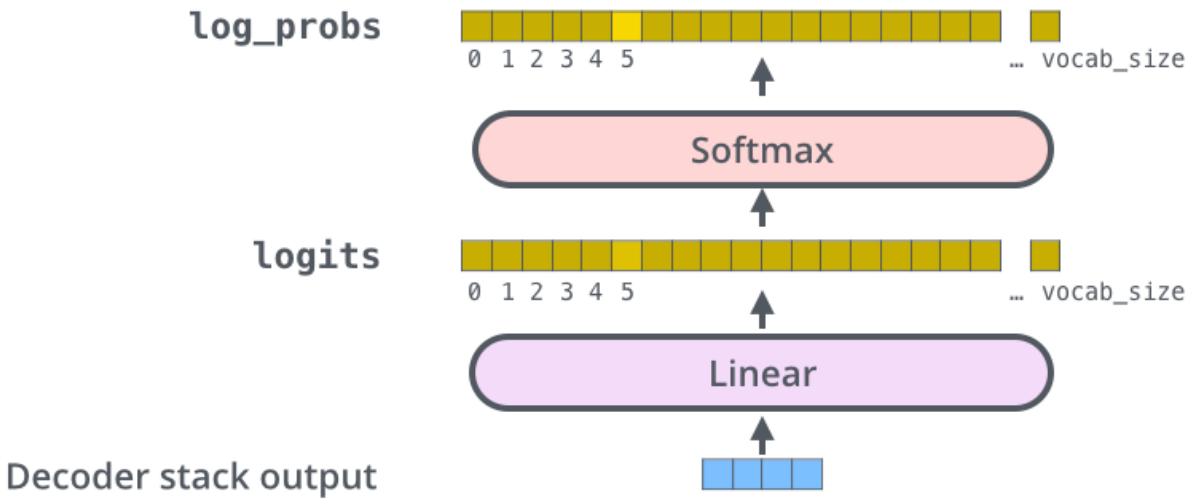
- ← attention mask shows
  - the position each target word (row) is allowed to look at (column)
- words are blocked for
  - attending to future words during training

Logits and softmax layers

- linear layer: float vector $\longrightarrow$ logits[7]

  ↑ decoder output ↑ length: _____ size (very large)

- softmax layer: logits $\longrightarrow$ probability vector



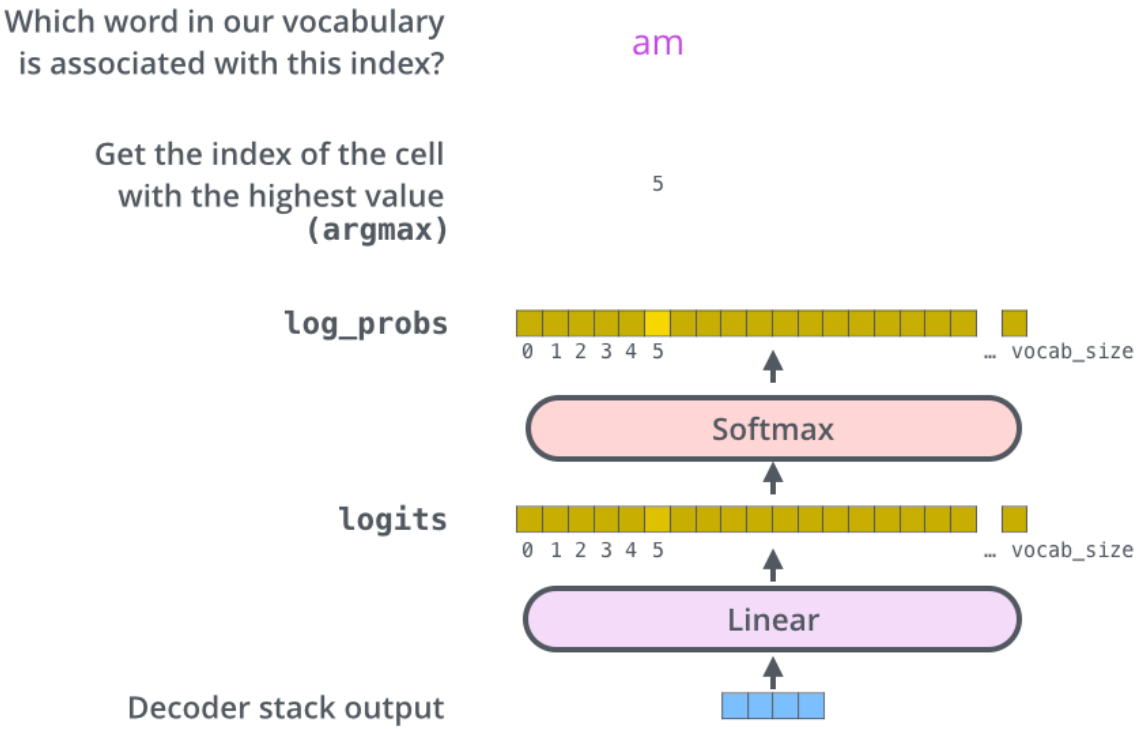
(source: The Illustrated Transformer)

✳ training loss: cross-entropy loss

[7] a vector of raw (non-normalized) predictions that a classification model generates, which is ordinarily then passed to a normalization function

Output symbol generation

- method #1: _____ decoding
  - output the word with the highest probability



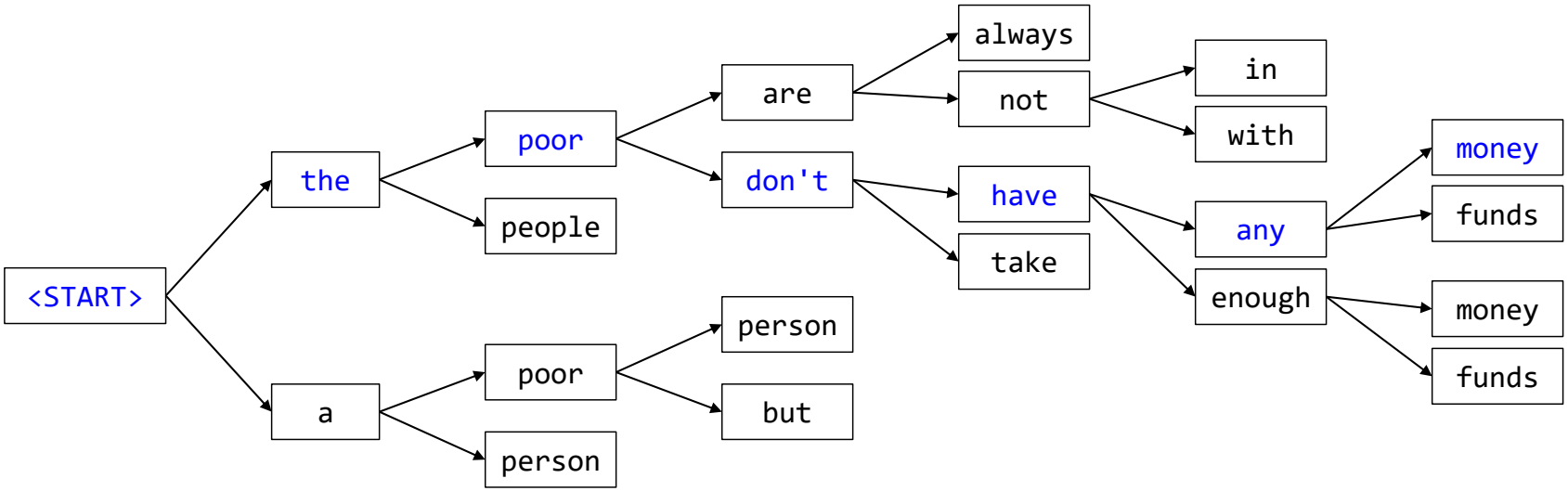
(source: The Illustrated Transformer)

- method #2: _____ search (hyperparameters: beam size k, top beams t)[8]
  - in a decoding step, consider k choices
  - run k different models simultaneously
  - in the next step, return t most probable paths; continue

e.g. $k = t = 2$

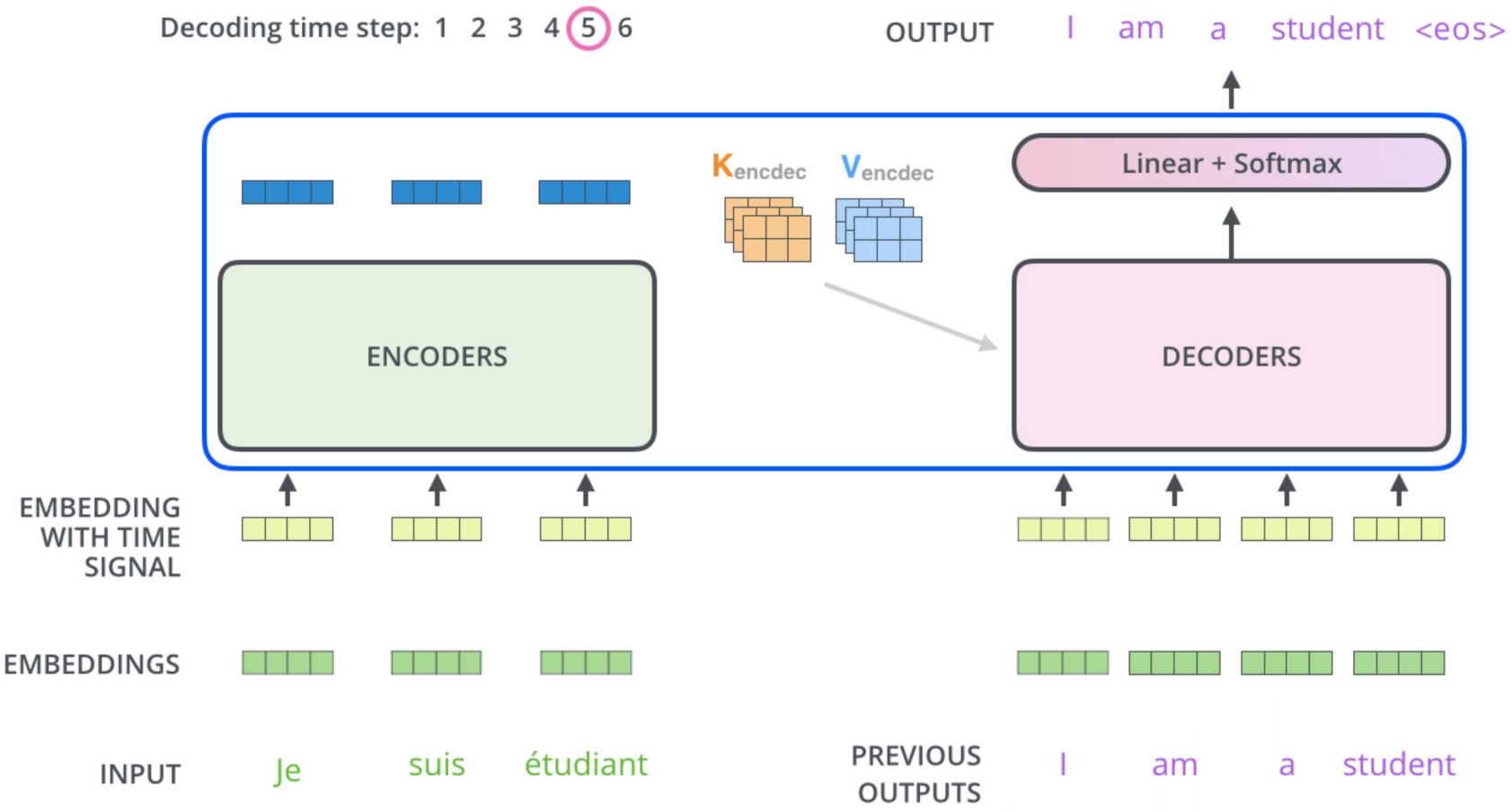
source: `les pauvres sont démunis`

target: `the poor don't have any money`



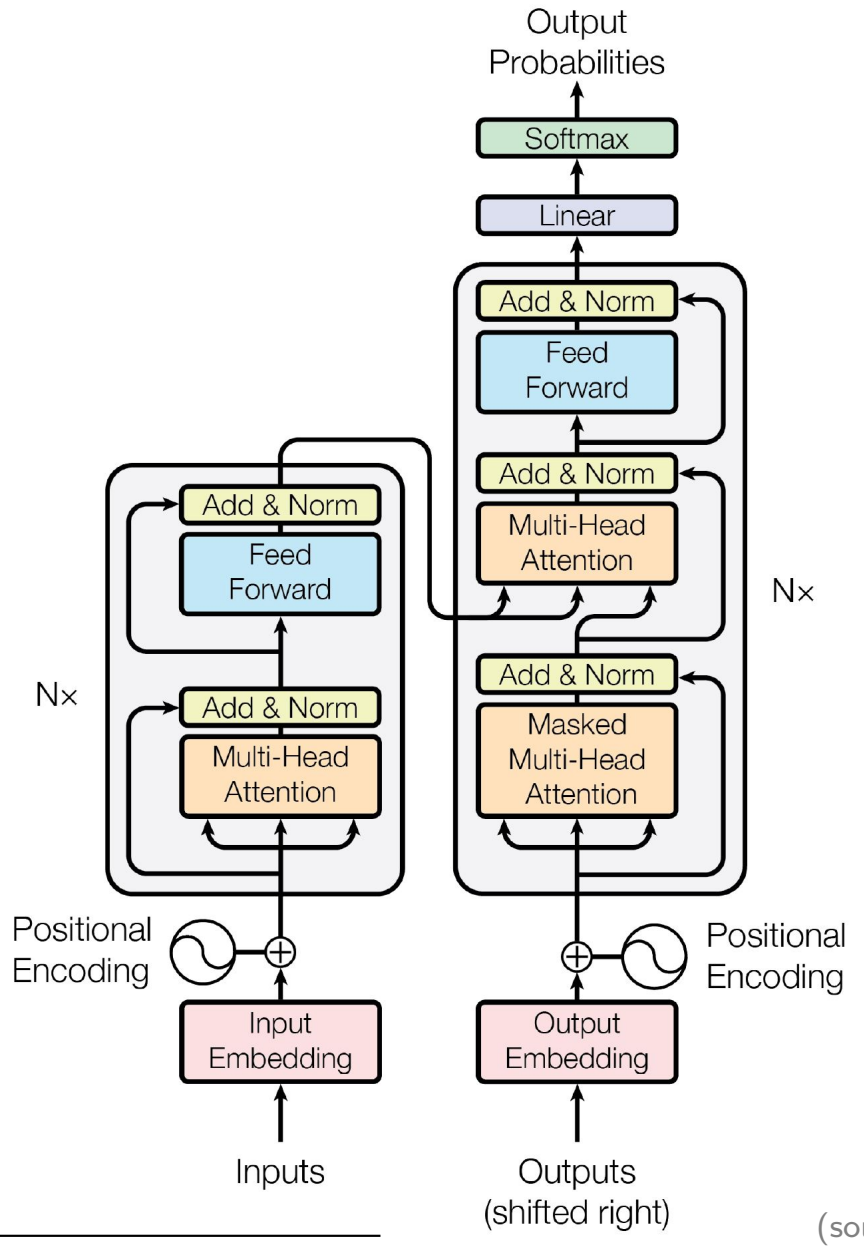
[8] often set $k = t$; $k = t = 1$ corresponds to greedy decoding

Altogether



(source: The Illustrated Transformer)

The transformer



(source: Vaswani et al., 2017[9])

[9] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017a). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008

Outline

BERT (Bidirectional Encoder Representations from Transformers[10])

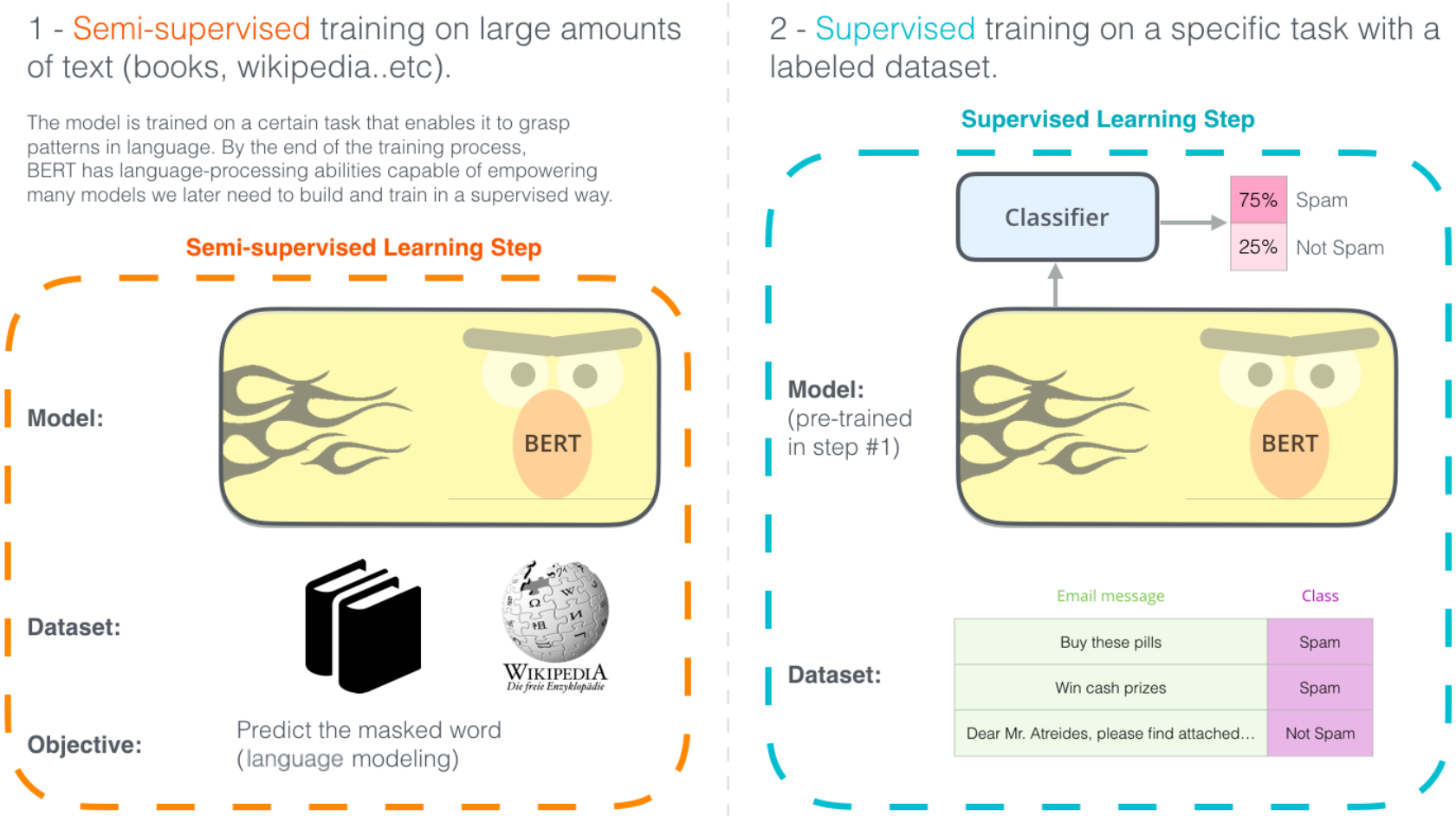
- a language representation model
  - can pre-train deep _____ representations from unlabeled text
  - by jointly conditioning on both left and right context in all layers

- pre-trained BERT can be fine-tuned with just one additional output layer
  - gives state-of-the-art models for various tasks (*e.g.* question answering)
  - without substantial task-specific architecture modifications

- in a sense, a denoising autoencoder for NLP

[10] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*

- pre-training → fine-tuning

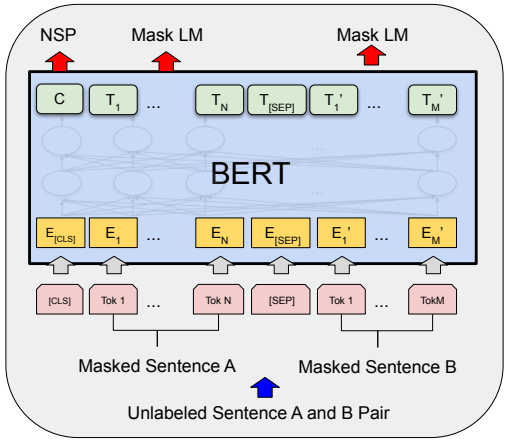


(source: Trivedi)

Defining a prediction goal

- a major challenge in training language models
- many models
  - predict the next word in a sequence

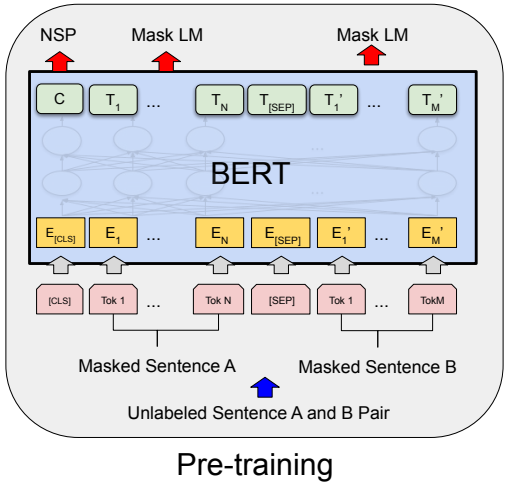
  e.g. `The child came home from ___`
  - a directional approach $\Rightarrow$ inherently limits context learning
- BERT uses two "pre-training" strategies to overcome this challenge
  1. _____ language model (MLM)
  2. next sentence prediction (NSP)
  - MLM and NSP are trained together, minimizing a combined loss function



Pre-training

(source: Devlin et al., 2018)

Masked LM (MLM)



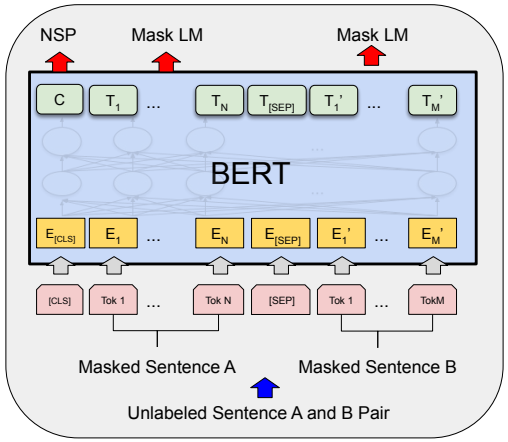
Pre-training

(source: Devlin et al., 2018)

during training:

- before feeding word sequences into BERT
  - select 15% of words in each sentence, and then
    - 80%: replace with `[MASK]` token
    - 10%: replace with random words
    - 10%: do nothing
- the model attempts to predict the original value of the masked words
  - based on the context provided by the other, _____ words

Next sentence prediction (NSP)



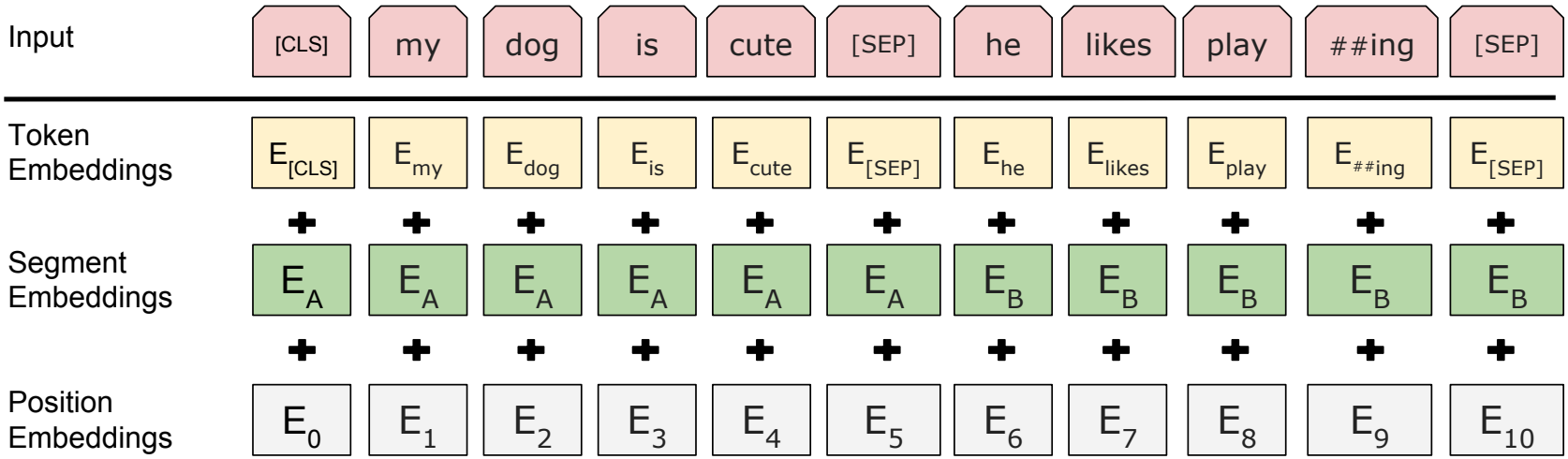
Pre-training

(source: Devlin et al., 2018)

during training:

- the model receives two sentences as input and learns to predict
  - if the second sentence is the _____ sentence in the original document
- training set composition
  - 50%: pairs of sentences actually appearing in the original document
  - 50%: random pairs of sentences chosen from the corpus

- input pre-processing[11] for NSP
  1. [CLS] token: inserted at the beginning of 1st sentence
  2. [SEP] token: inserted at the end of each sentence
  3. _____ embedding (to indicate each sentence) is added to each token
  4. positional embedding is added to each token

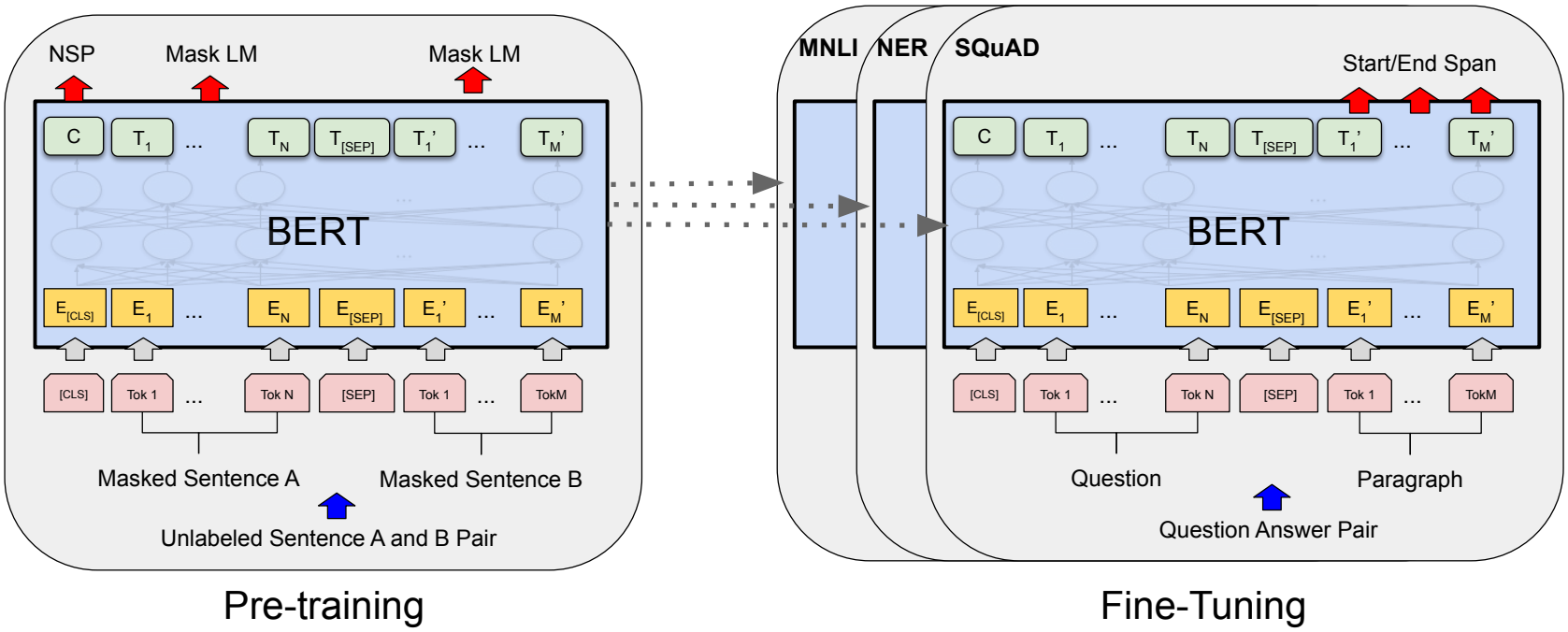


(source: Devlin et al., 2018)

[11] [CLS] and [SEP] refer to classification and separation, respectively

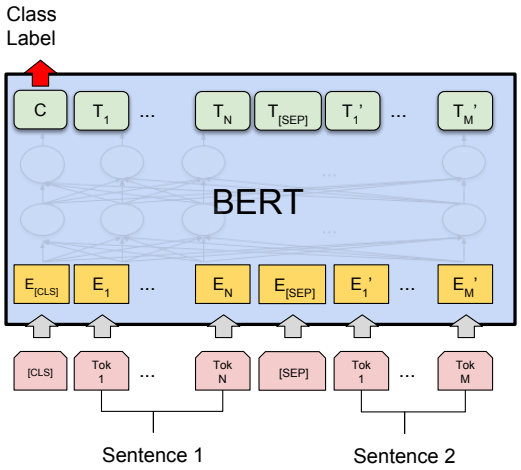
How to use BERT (fine-tuning)

- can be used for various NLP tasks (only adding a small layer to the core)
  - classification
  - question answering (QA)
  - named entity recognition (NER)

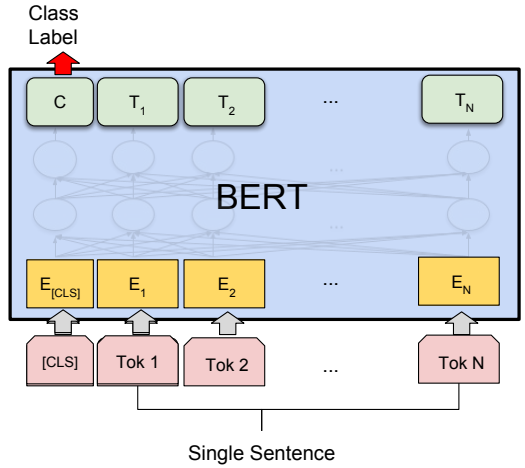


(source: Devlin et al., 2018)

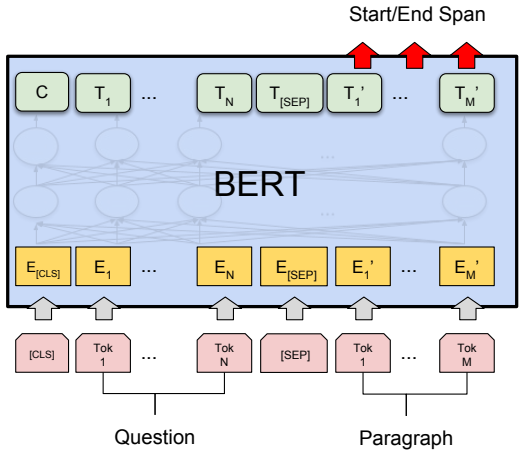
- fine-tuning BERT on different tasks



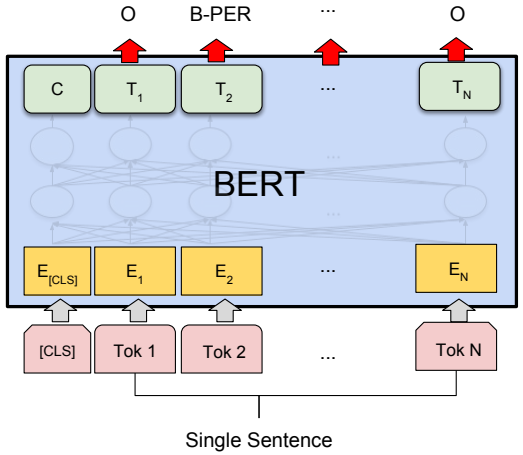
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



(b) Single Sentence Classification Tasks: SST-2, CoLA



(c) Question Answering Tasks: SQuAD v1.1

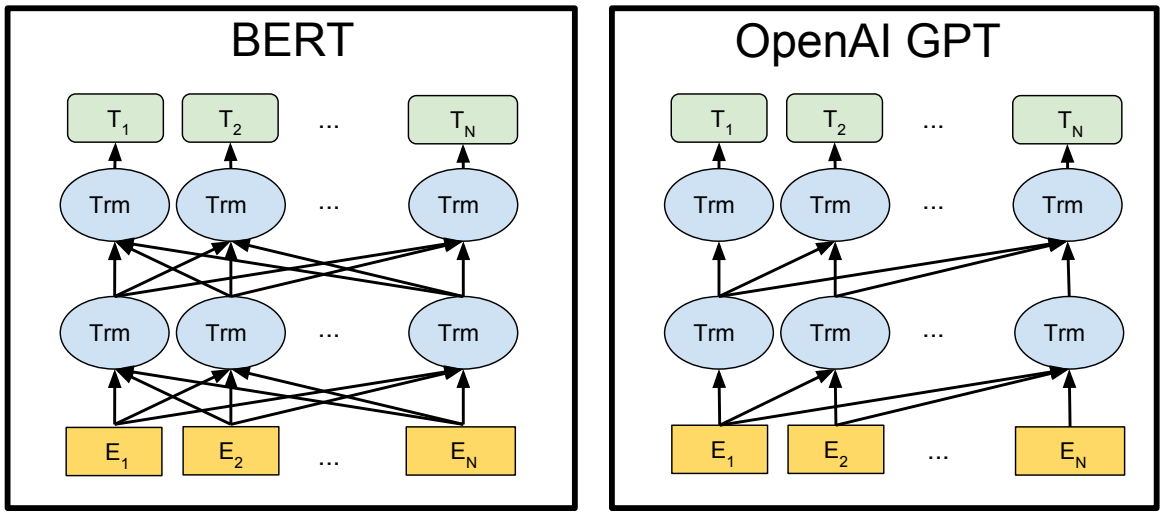


(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

(source: Devlin et al., 2018)

OpenAI GPT (Generative Pre-trained Transformer[12])

- similar idea
  - unsupervised pre-training + transformer
- key difference
  - BERT: bidirectional transformer
  - GPT: _____ transformer

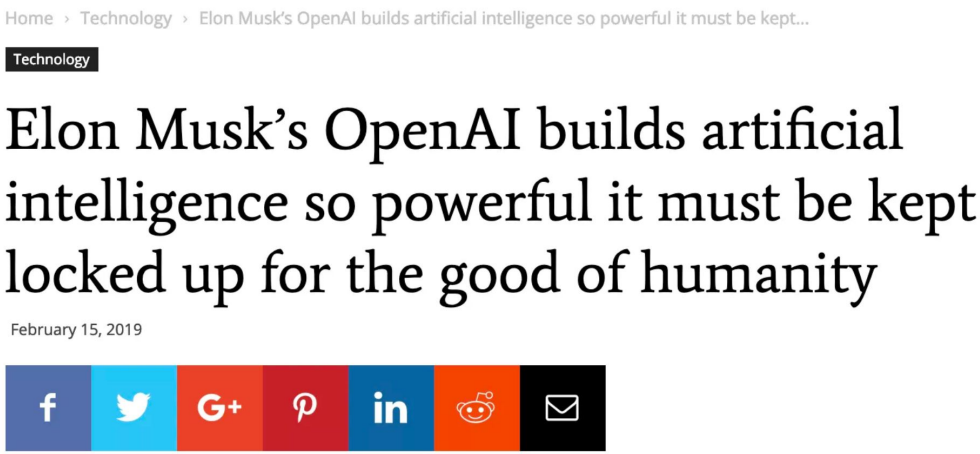


(source: Devlin et al., 2018)

[12] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. *OpenAI Blog*

OpenAI GPT-2[13]

- direct scale-up version of GPT
  - 1.5 billion parameters ($>$ 10x) trained with 8 million webpages ($>$ 10x)



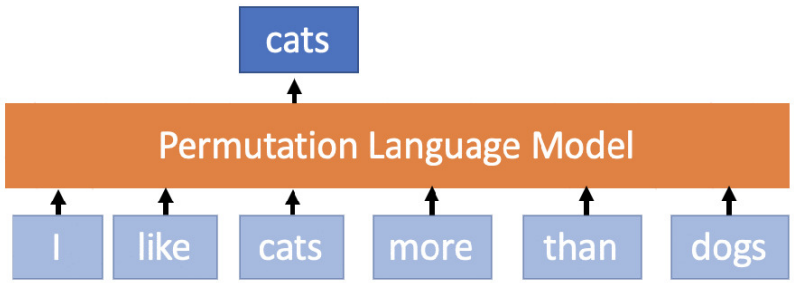
(source: The Verge, 2019)

[13] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*

XLNet[14]

- BERT limitations
  - neglects dependency between masked positions
  - suffers from a pretrain-finetune discrepancy

- XLNet: a generalized autoregressive pretraining method
  - based on transformer-XL (autoregressive model: RNN + transformer)
  - learns bidirectional contexts
    - by considering all _____ of the factorization order

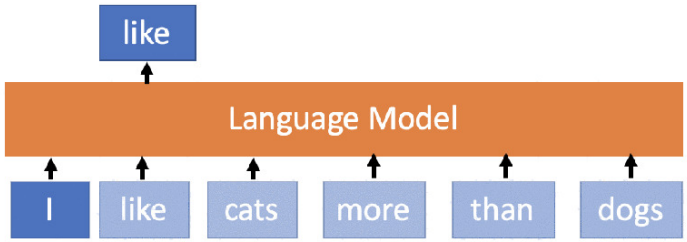


(source: Kurita)

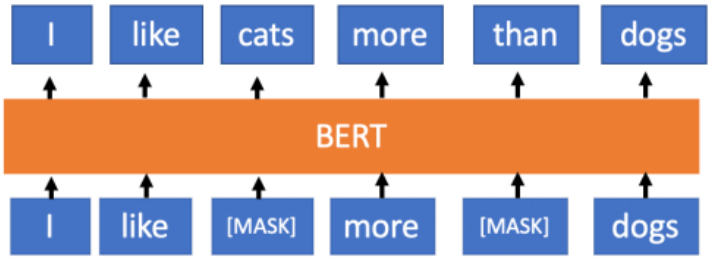
[14] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*

comparison:

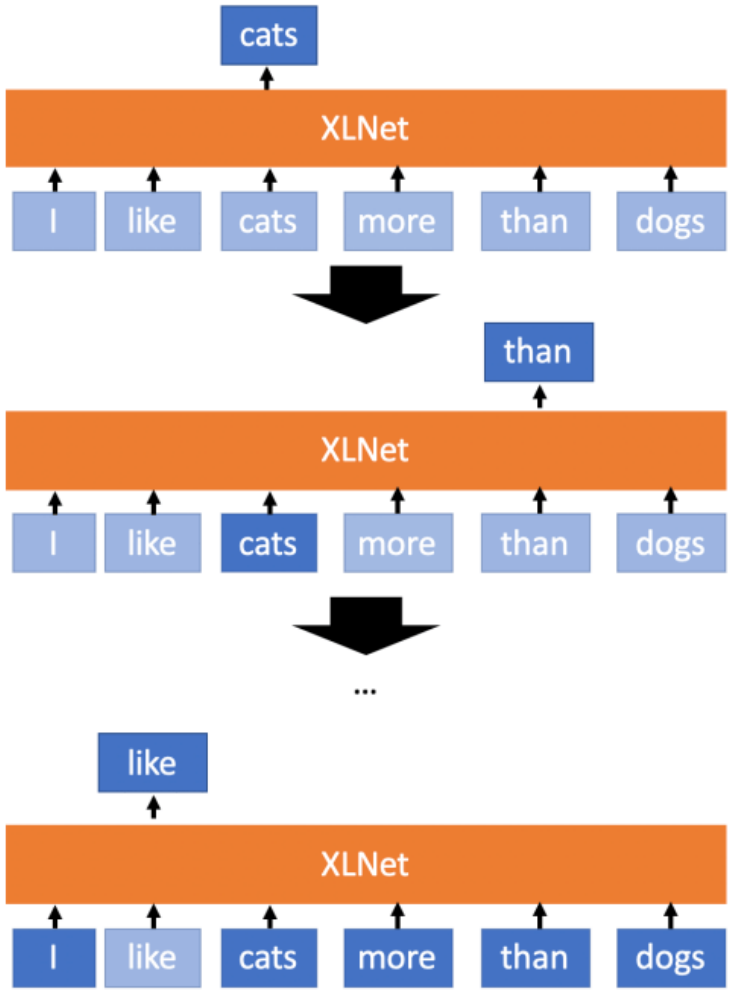
- traditional LM



- BERT[15]



- XLNet[16]



(source: Kurita)

[15] predicts all masked words simultaneously

[16] learns to predict the words in an arbitrary order but in an autoregressive, sequential manner (not necessarily left-to-right)

Outline

Summary

- "attention is all you need": attention in deep learning
  - produces state-of-the-art results in natural language processing tasks
  - self/cross-attention, soft/hard attention, global/local attention
  - potentially more efficient/effective in temporal modeling than CNN/RNN
- the transformer
  - a sequence-to-sequence model having encoder-decoder architecture
  - encoder: self-attention + feed-forward layers
  - decoder: masked self-attention + cross-attention + feed-forward layers
  - multi-head attention, residual connection, layer norm
- applications of transformer
  - powerful language representation models: BERT, OpenAI GPT/GPT-2
  - BERT: pre-training (MLM + NSP) $\rightarrow$ fine-tuning
  - XLNet: based on transformer-XL (RNN + transformer)