Contextual Representation

COMP90042

Natural Language Processing

Lecture 11

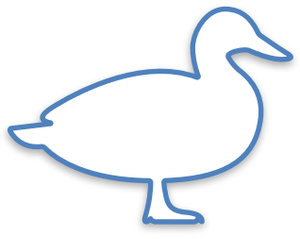
Semester 1 2022 Week 6

Jey Han Lau

THE UNIVERSITY OF MELBOURNE

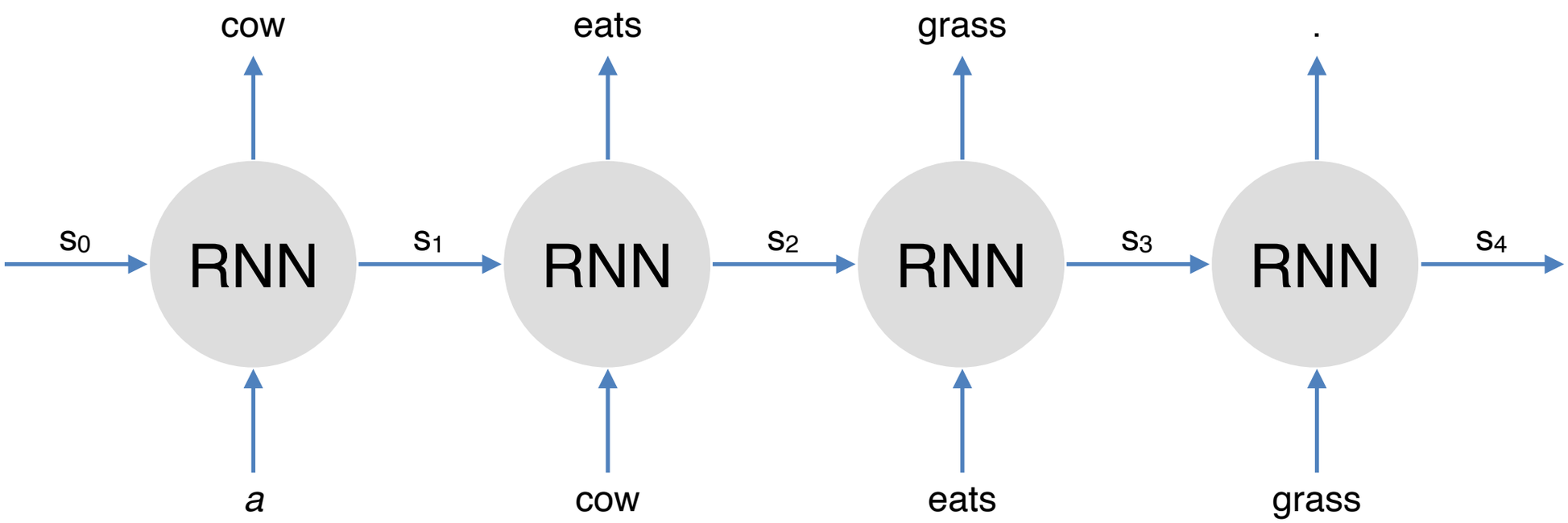
Word Vectors/Embeddings

- Each word type has one representation
  - Word2Vec
- Always the same representation regardless of the context of the word
- Does not capture multiple senses of words

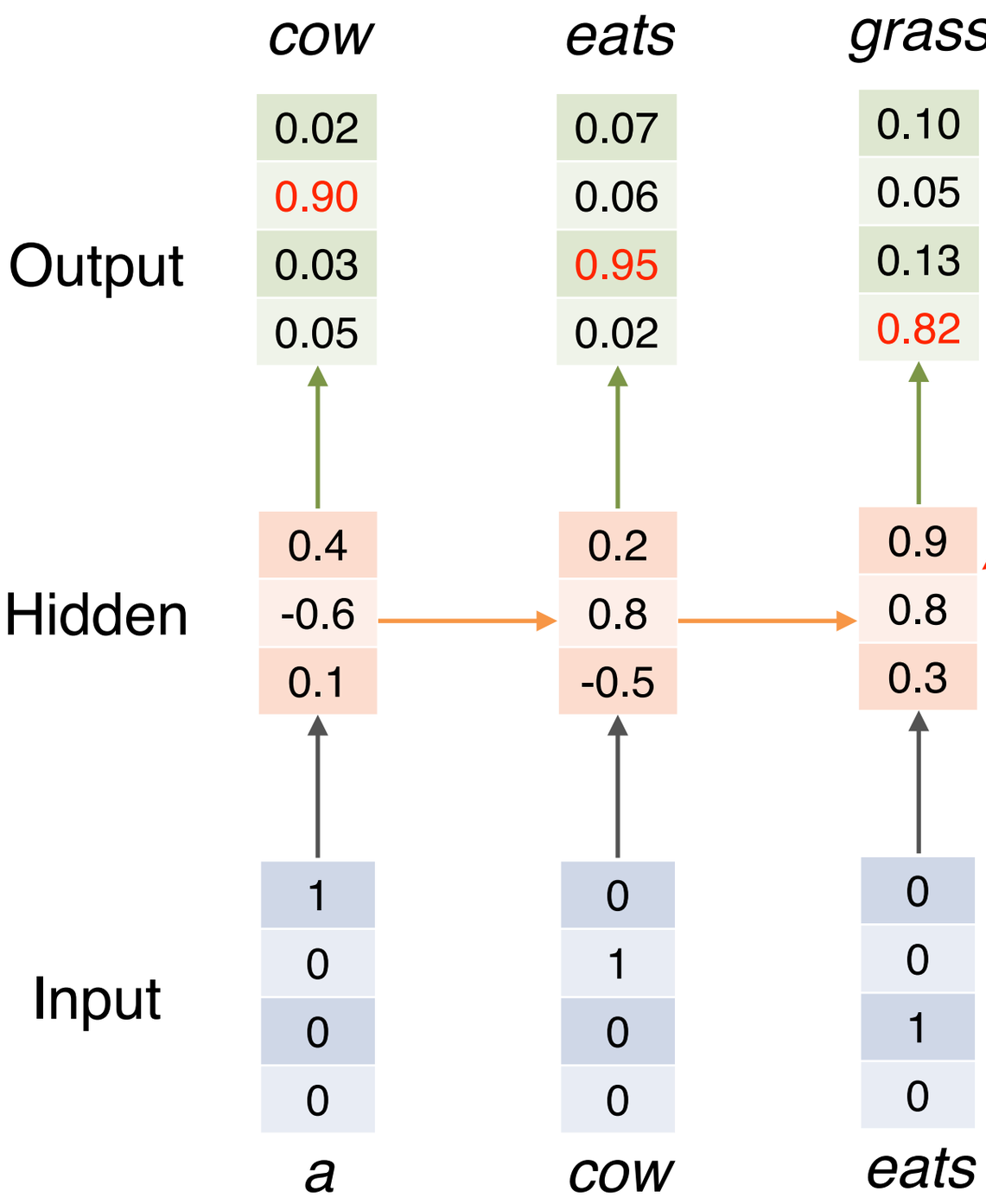


- Contextual representation = representation of words based on context
- Pretrained contextual representations work *really well* for downstream applications!

RNN Language Model



RNN Language Model



Contextual representation!

Word embedding matrix

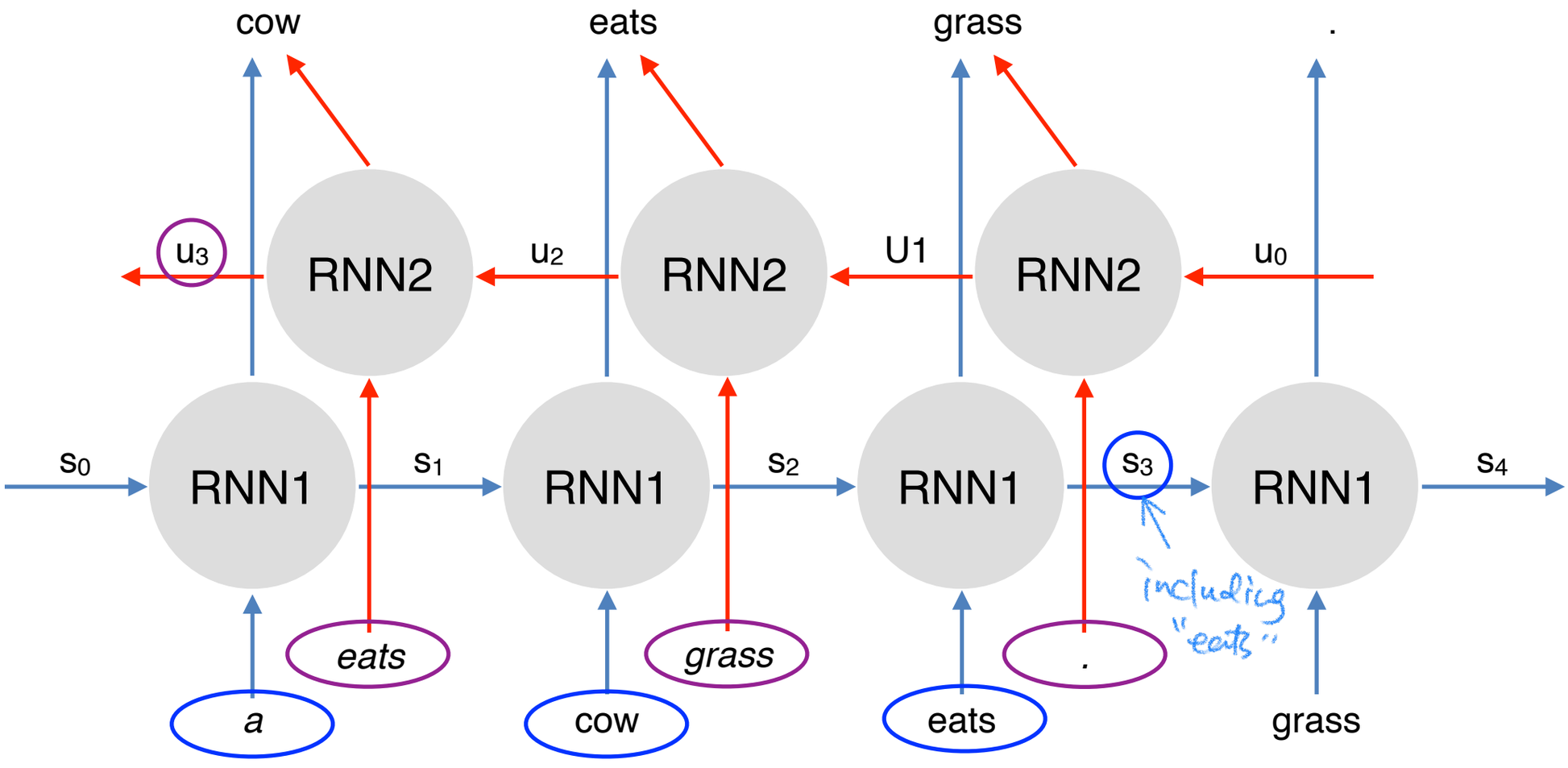
$$s_i = \tanh(W_s s_{i-1} + W_x x_i + b)$$

$$y_i = \text{softmax}(s_i)$$

Solved?

- Almost, but the contextual representation only captures context to the left
- Solution: use a bidirectional RNN instead!

Bidirectional RNN



a cow eats grass .

$$[s_3, u_3]$$

Bidirectional Contextual Representation!

Outline

- ELMo
- BERT
- Transformers

ELMo

ELMo: **E**mbeddings from **L**anguage **Mo**dels

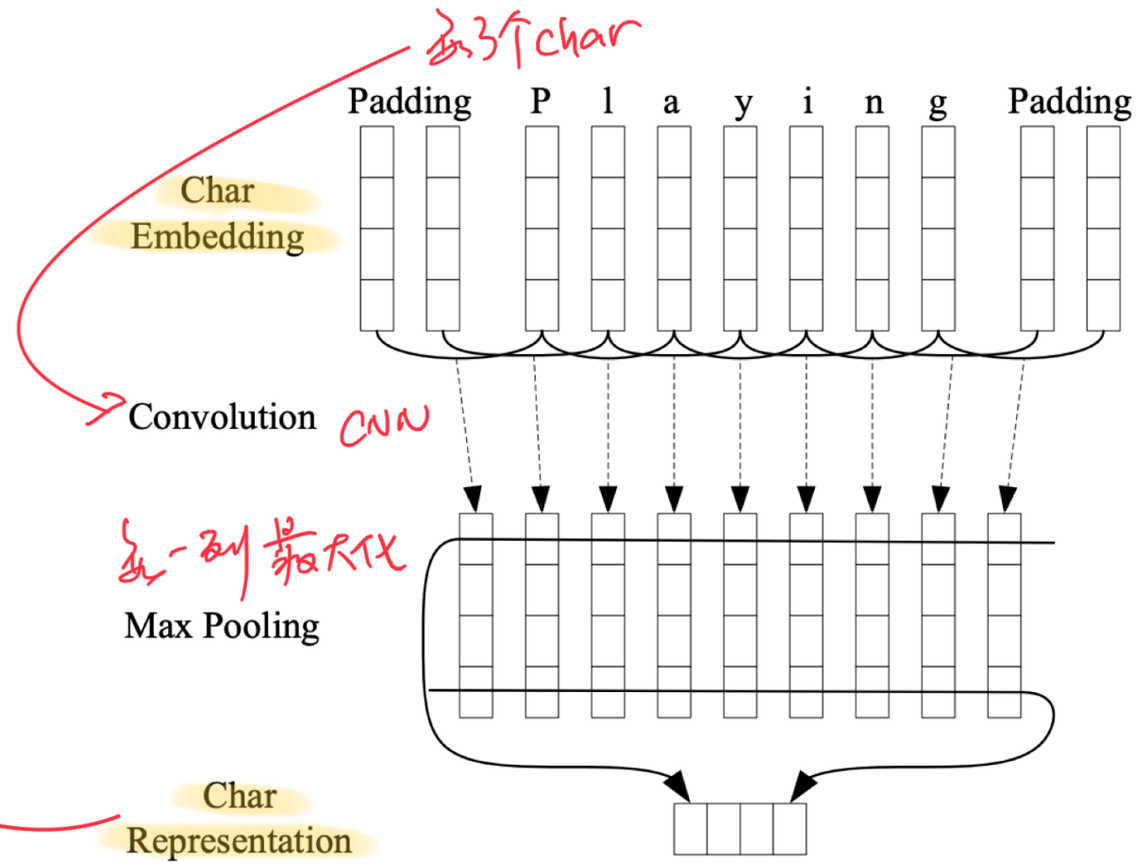
- Peters et al. (2018): https://arxiv.org/abs/1802.05365v2
- Trains a bidirectional, multi-layer LSTM language model over 1B word corpus
- Combine hidden states from **multiple layers** of LSTM for downstream tasks
  - Prior studies use only top layer information
- Improves task performance significantly!

ELMo

- Number of LSTM layers = 2
- LSTM hidden dimension = 4096
- Character convolutional networks (CNN) to create word embeddings
  - No unknown words

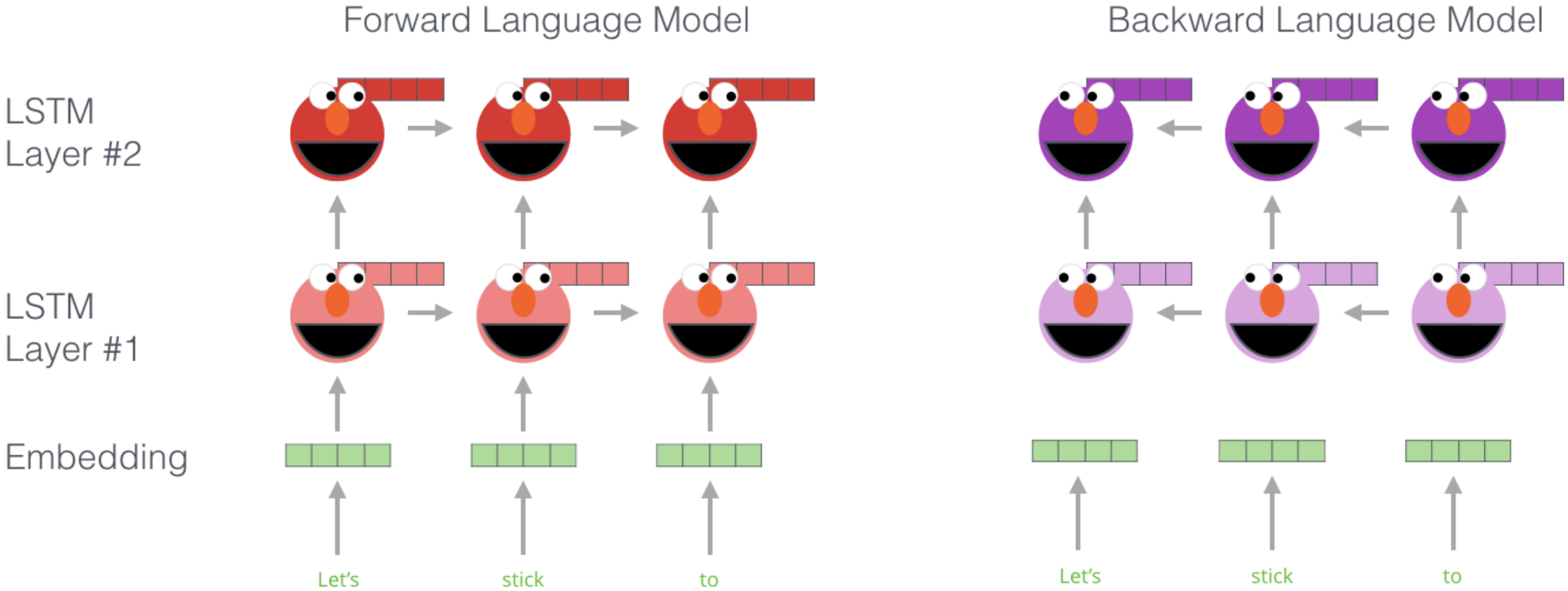
所有单词由26个字母组成

初始化 ✓



Extracting Contextual Representation

Embedding of “stick” in “Let’s stick to” - Step #1

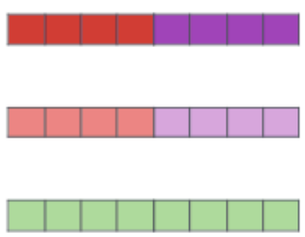


http://jalammar.github.io/illustrated-bert/

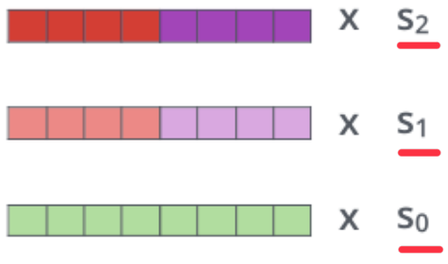
Extracting Contextual Representation

Embedding of “stick” in “Let’s stick to” - Step #2

1- Concatenate hidden layers

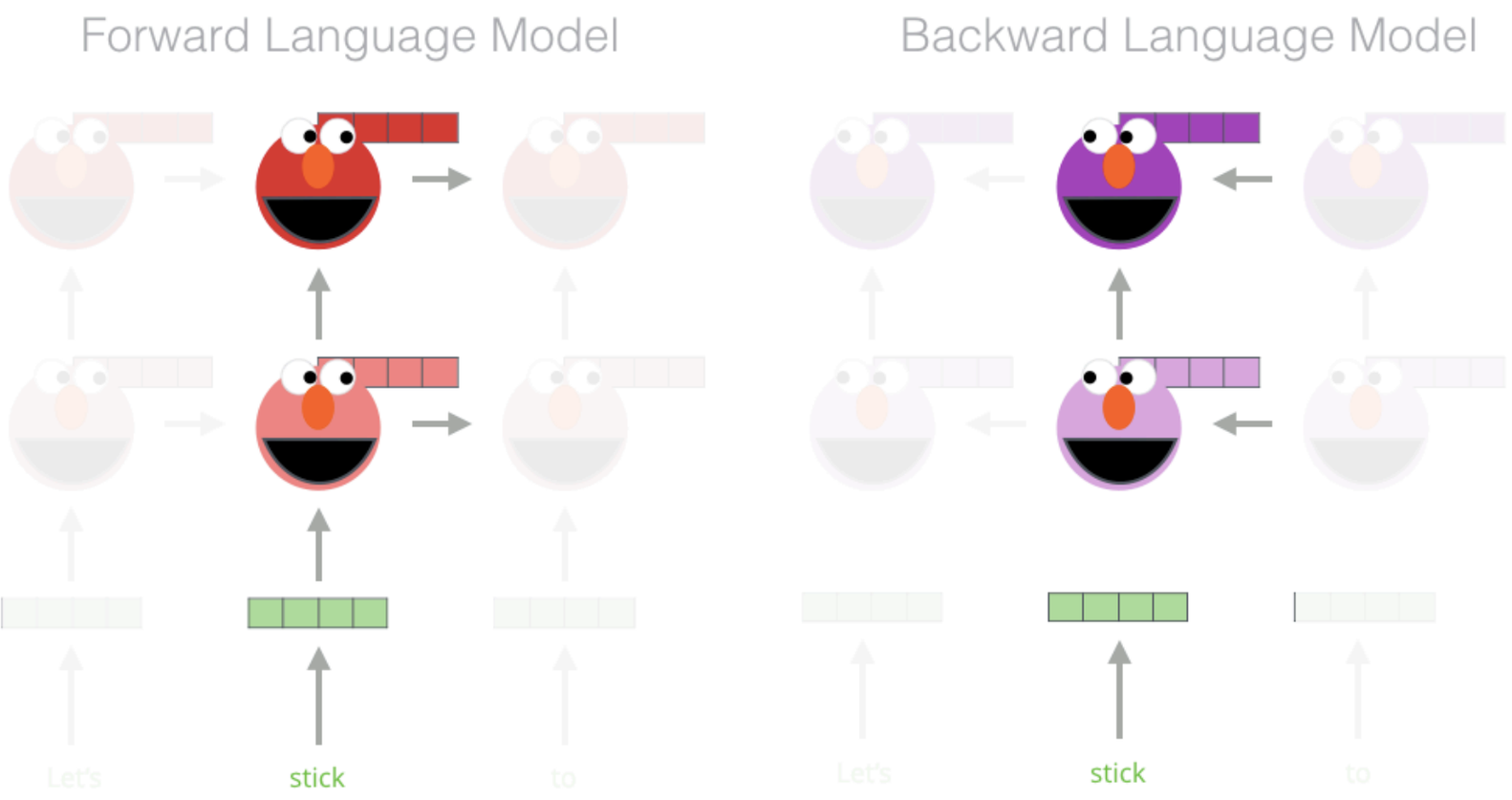


2- Multiply each vector by a weight based on the task

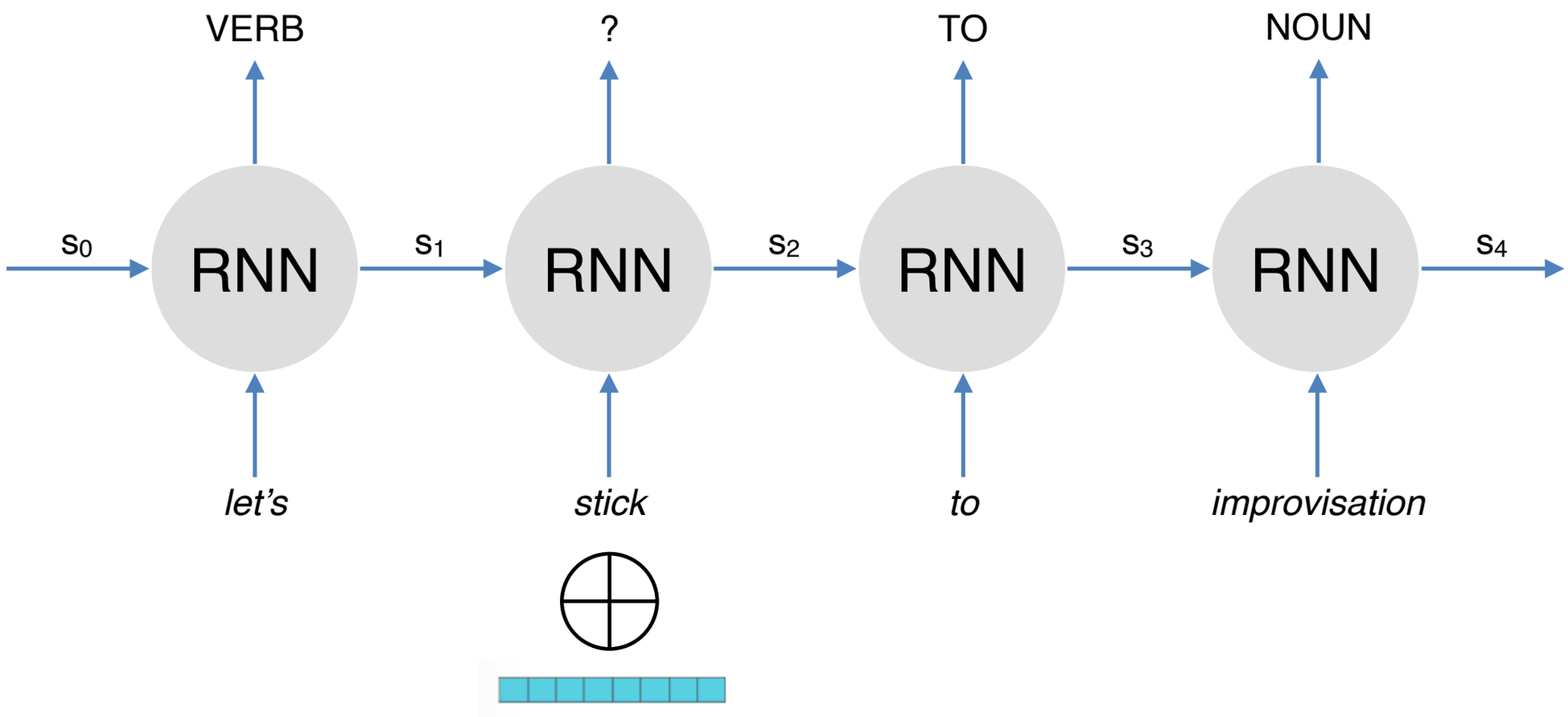


3- Sum the (now weighted) vectors

ELMo embedding of “stick” for this task in this context



Downstream Task: POS Tagging



$$s_i = \tanh(W_s s_{i-1} + (W_x x_i \oplus e_i) + b)$$

How Good is ELMO?

TASK	PREVIOUS SOTA		OUR BASELINE	ELMO + BASELINE	INCREASE (ABSOLUTE/ RELATIVE)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

- SQuAD: QA
- SNLI: textual entailment
- SRL: semantic role labelling
- Coref: coreference resolution
- NER: named entity recognition
- SST-5: sentiment analysis

Other Findings

- Lower layer representation = captures syntax
  - good for POS tagging, NER
- Higher layer representation = captures semantics
  - good for QA, textual entailment, sentiment analysis

Contextual vs. Non-contextual

	Source	Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder {...}	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway play for Garson {...}	{...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

Table 4: Nearest neighbors to "play" using GloVe and the context embeddings from a biLM.

What are the disadvantages of contextual embeddings?

- Difficult to do intrinsic evaluation (e.g. word similarity, analogy)
- Interpretability More diff to do this
- Computationally expensive to train large-scale contextual embeddings
- Only works for certain languages

PollEv.com/jeyhanlau569

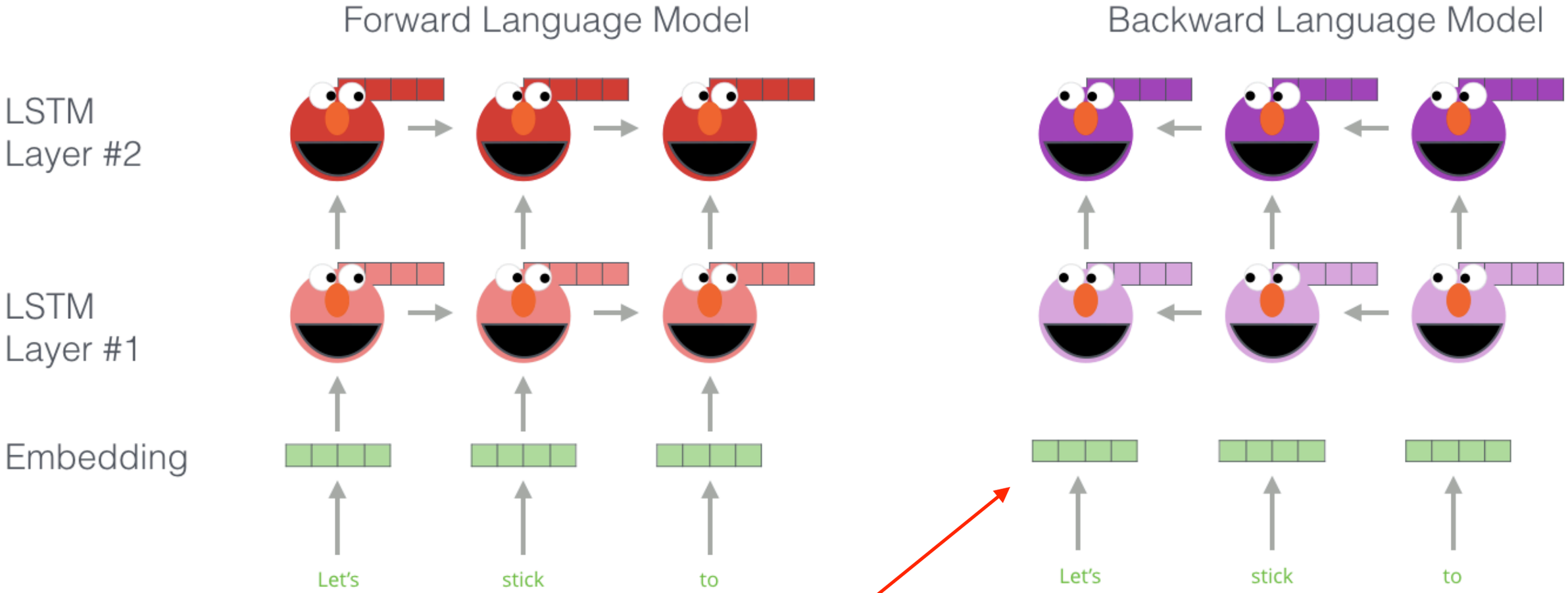
BERT

Disadvantages of RNNs

- Sequential processing: difficult to scale to very large corpus or models
- RNN language models run left to right (captures only one side of context)
- Bidirectional RNNs help, but they only capture surface bidirectional representations

Extracting Contextual Representation

Embedding of “stick” in “Let’s stick to” - Step #1

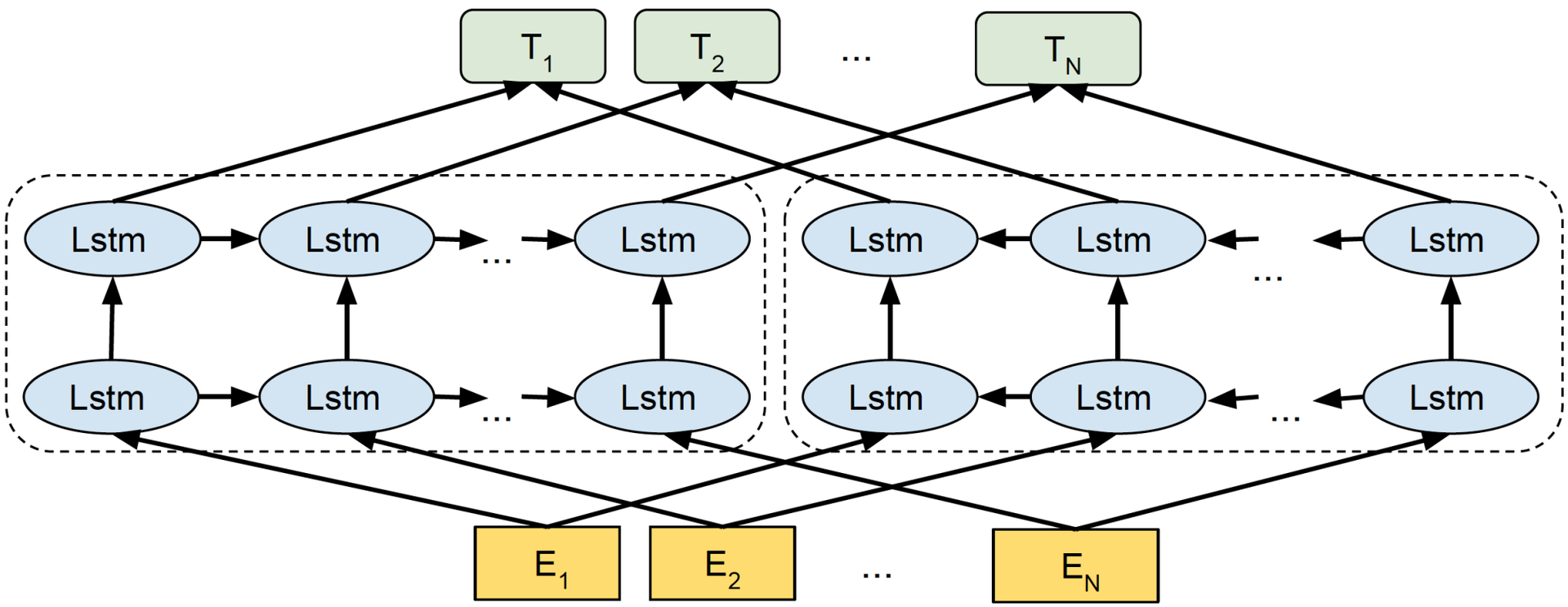


These two RNNS are run independently!
Information is aggregated after they have separately produced their hidden representations

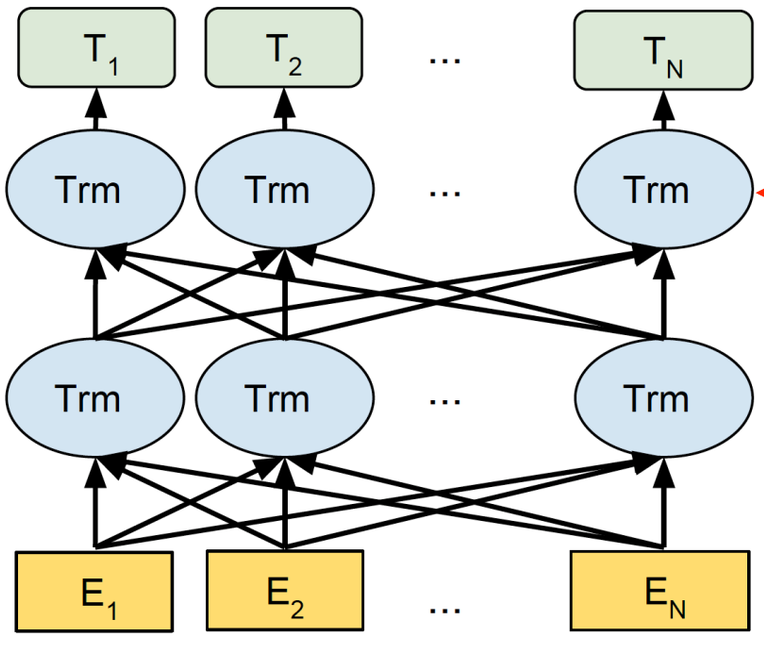
BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers

- Devlin et al. (2019): https://arxiv.org/abs/1810.04805
- Uses self-attention networks (aka **Transformers**) to capture dependencies between words
  - No sequential processing
- **Masked language model** objective to capture deep bidirectional representations
- Loses the ability to generate language
- Not an issue if the goal is to learn contextual representations

ELMo



BERT



?? We'll come back to describe Transformers in the last part of the lecture

Objective 1: Masked Language Model

- ‘Mask’ out k% of tokens at random
- Objective: predict the masked words

Objective 2: Next Sentence Prediction

- Learn relationships between sentences
- Predicts whether sentence B follows sentence A
- Useful pre-training objective for downstream applications that analyse sentence pairs (e.g. textual entailment)

Sentence A: *Today we have a lecture on NLP.*
Sentence B: *It is an interesting lecture.*
Label: IsNextSentence

Sentence A: *Today we have a lecture on NLP.*
Sentence B: *Polar bears are white.*
Label: NotNextSentence

Training/Model Details

- WordPiece (subword) Tokenisation
- Multiple layers of transformers to learn contextual representations
- BERT is pretrained on Wikipedia+BookCorpus
- Training takes multiple GPUs over several days

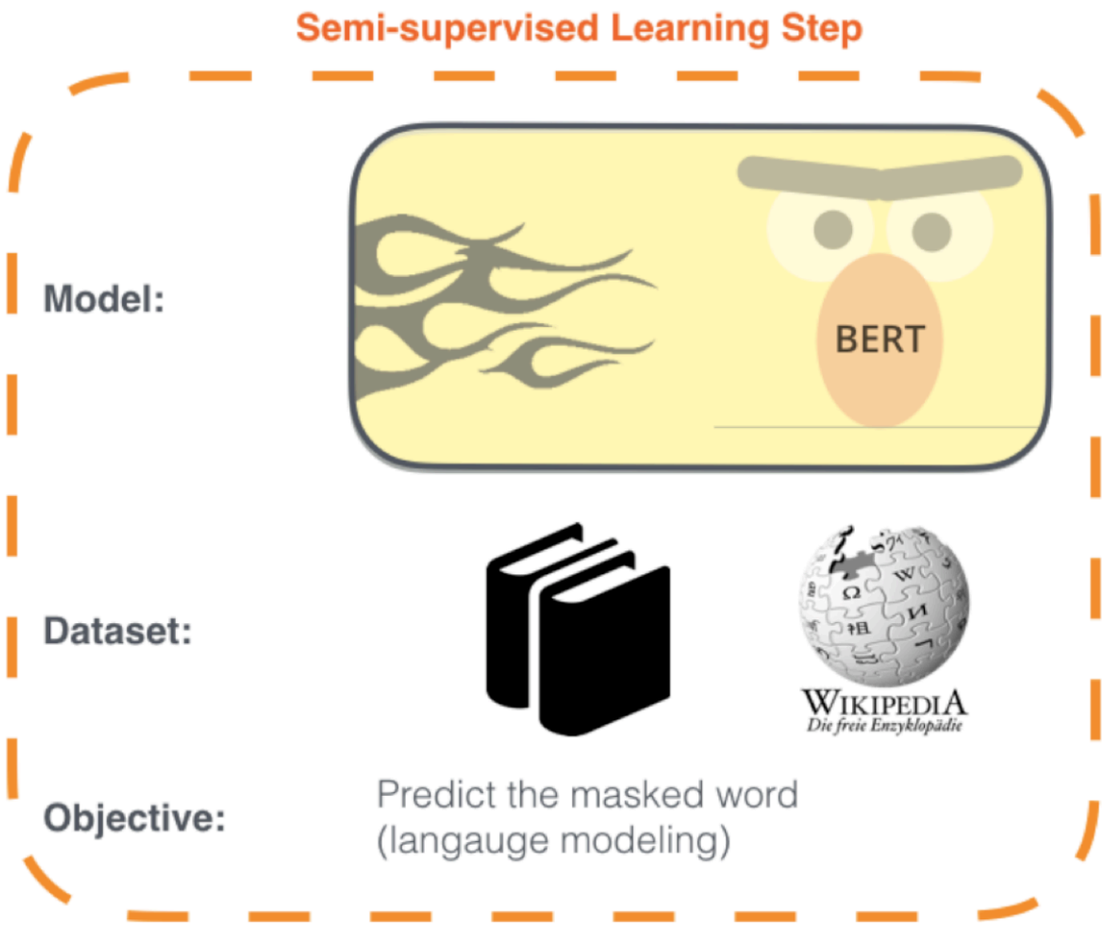
How To Use BERT?

- Given a pretrained BERT, continue training (i.e. fine-tune) it on downstream tasks
- But how to adapt it to downstream task?
  - Add a classification layer on top of the contextual representations

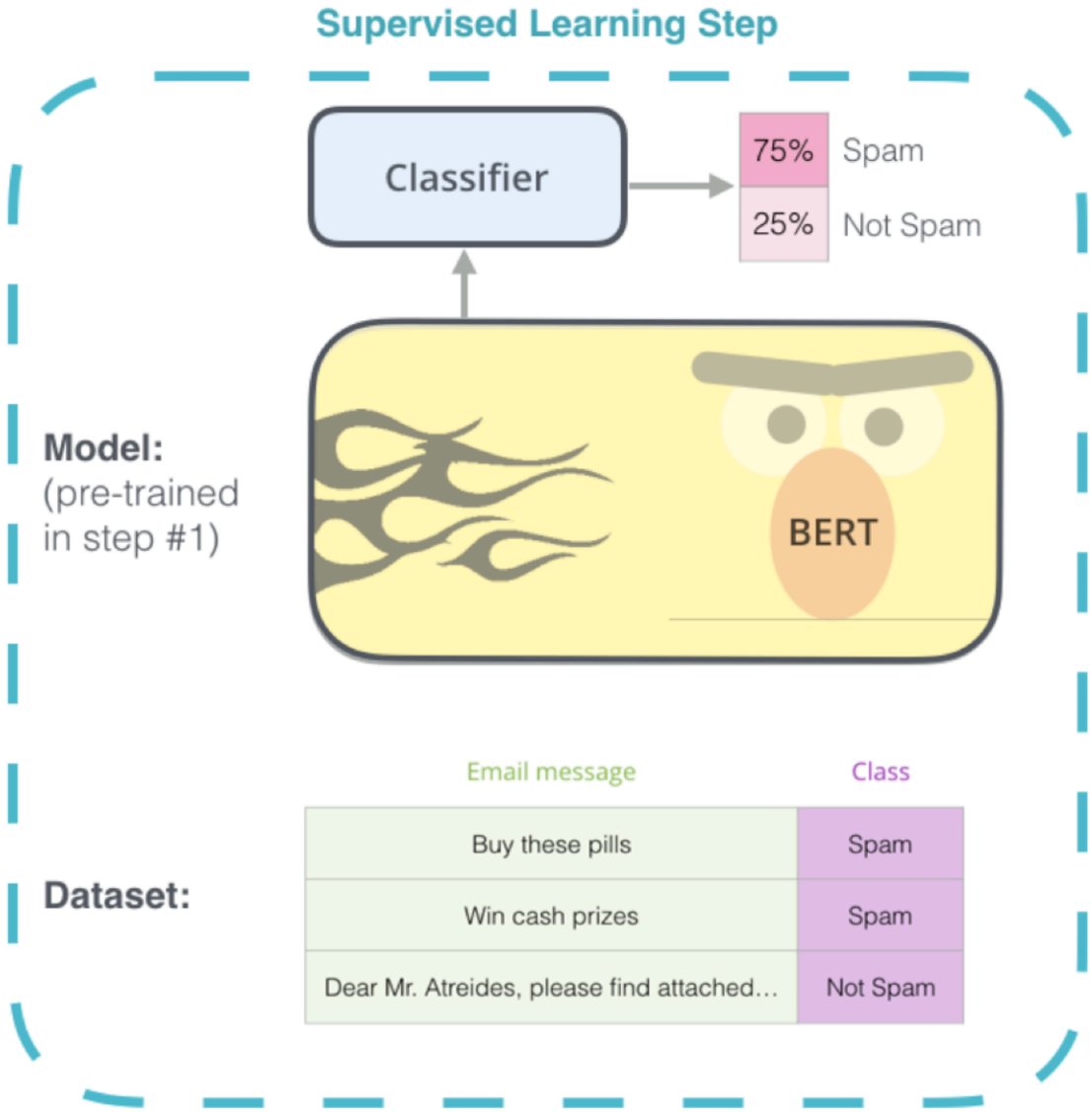
Training and Fine-Tuning

1 - Semi-supervised training on large amounts of text (books, wikipedia..etc).

The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.



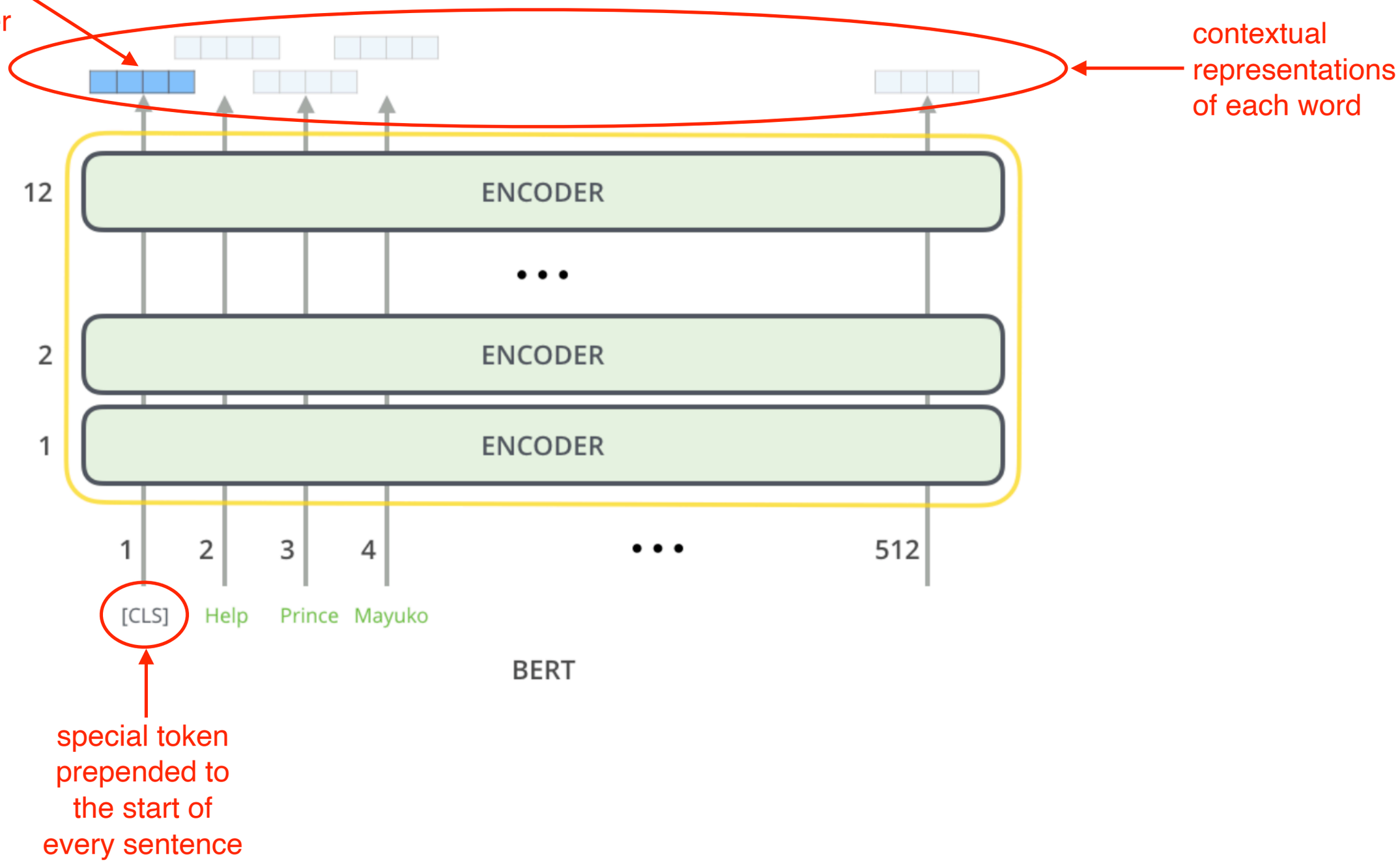
2 - Supervised training on a specific task with a labeled dataset.



Example: Spam Detection

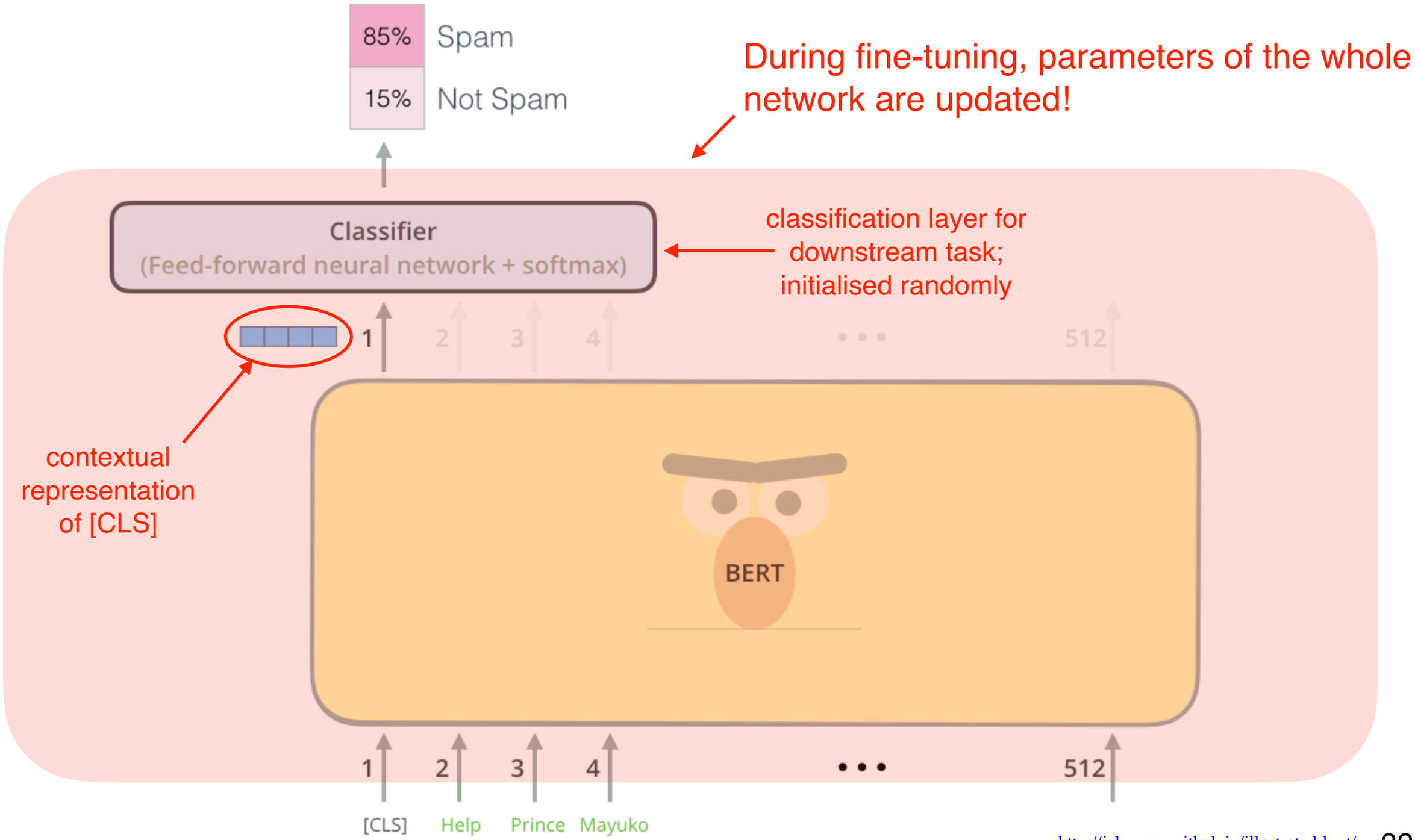
captures information of the whole sentence; input to new classification layer we're adding

contextual representations of each word



special token prepended to the start of every sentence

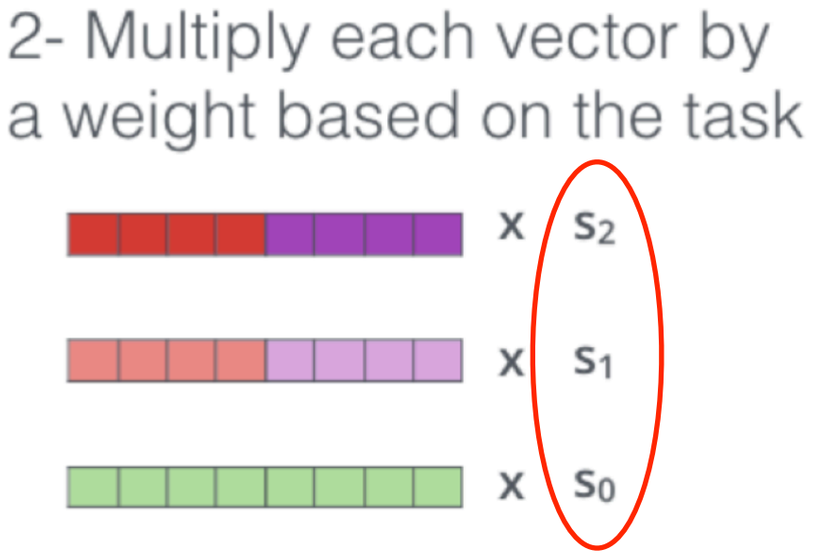
Example: Spam Detection



http://jalammar.github.io/illustrated-bert/

BERT vs. ELMo

- ELMo provides only the contextual representations
- Downstream applications has their own network architecture
- ELMo parameters are fixed when applied to downstream applications
  - Only the weights to combine states from different LSTM layers are learned



BERT vs. ELMo

- BERT adds a classification layer for downstream tasks
  - No task-specific model needed
- BERT updates all parameters during fine-tuning

How Good is BERT?

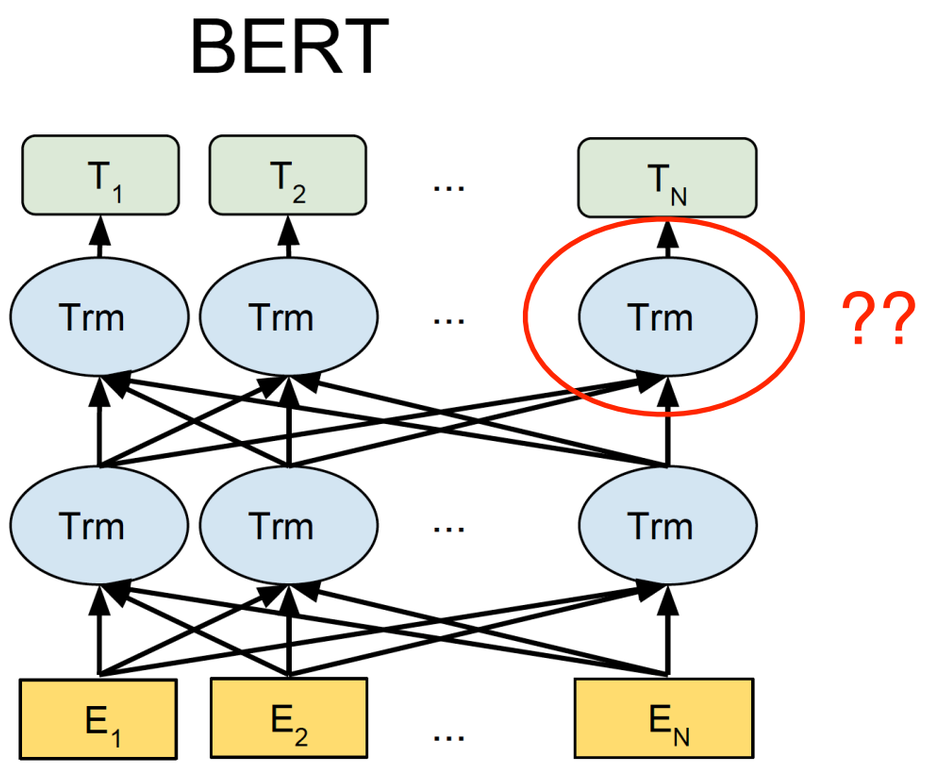
System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

- MNLI, RTE: textual entailment
- QQP, STS-B, MRPC: sentence similarity
- QNLI: answerability prediction
- SST-2: sentiment analysis
- COLA: sentence acceptability prediction

Transformers

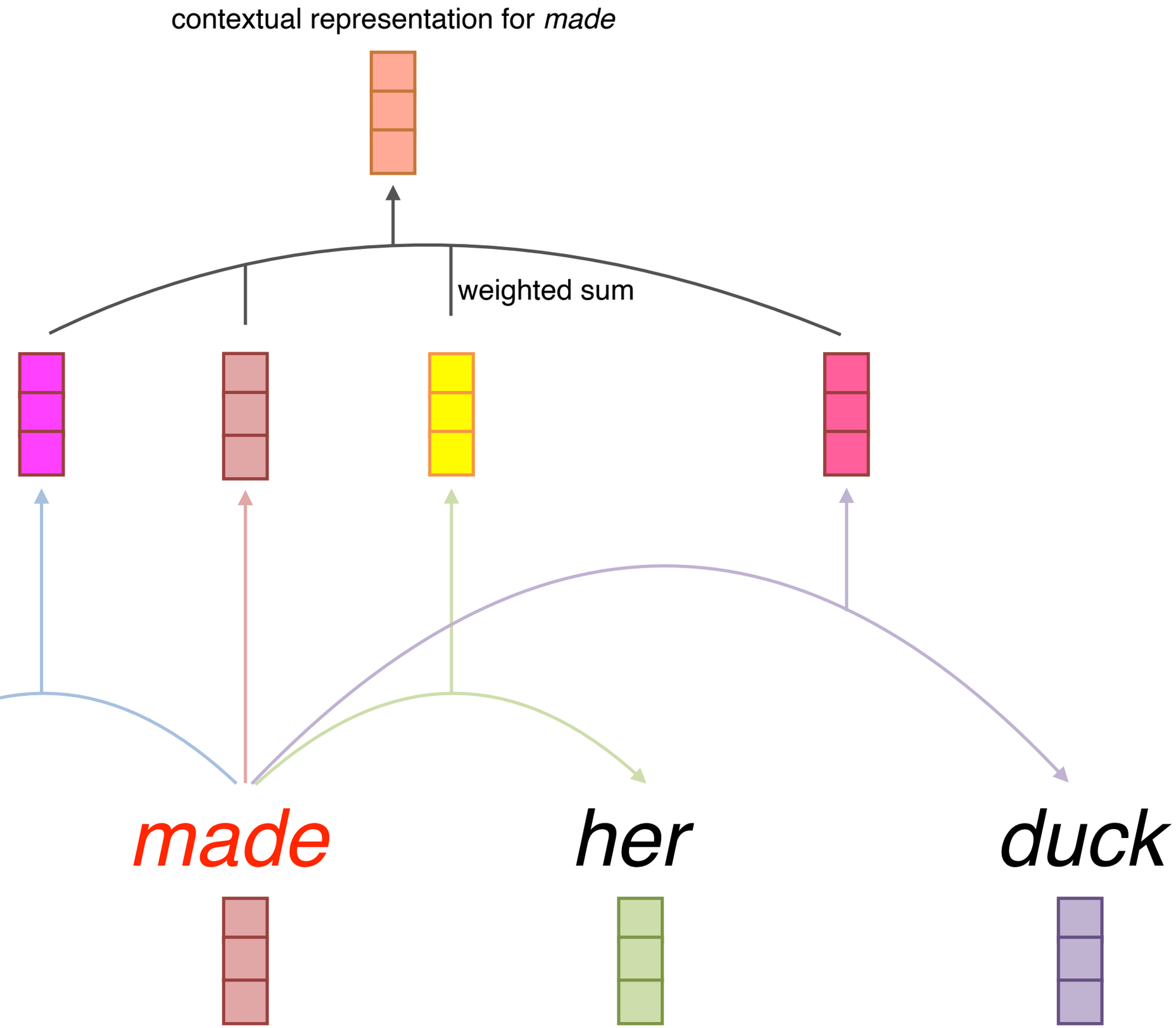
Transformers

- What are transformers, and how do they work?



Attention is All You Need

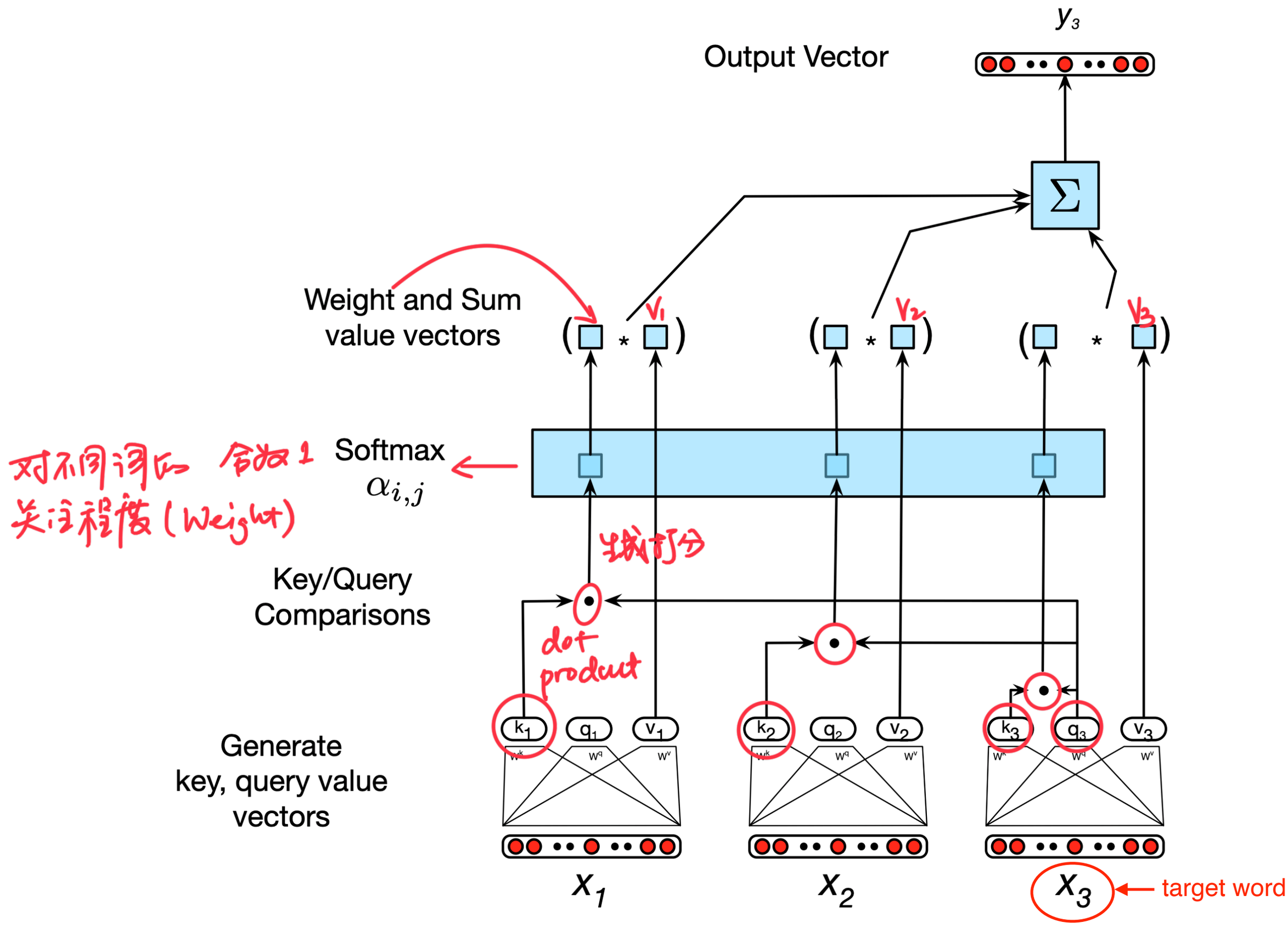
- Vaswani et al. (2017): https://arxiv.org/abs/1706.03762
- Use **attention** instead of using RNNs (or CNNs) to capture dependencies between words


contextual representation for *made*
weighted sum
I
made
her
duck

Self-Attention via Query, Key, Value

- Input:
  - query q (e.g. *made*) ← 待查询词
  - key k and value v (e.g. *her*)
- Query, key and value are all **vectors**
  - linear projections from embeddings
- Comparison between **query vector of target word** (*made*) and **key vectors of context words** to compute **weights**
- Contextual representation of target word = weighted sum of **value vectors of context words and target word**

$$c_{\text{made}} = 0.1v_{\text{I}} + 0.5v_{\text{made}} + 0.2v_{\text{her}} + 0.3v_{\text{duck}}$$


y_3
Output Vector
Σ
Weight and Sum
value vectors
v_1
v_2
v_3
对不同词Q 给权重
关注程度(Weight)
Softmax
$\alpha_{i,j}$
生成打分
Key/Query
Comparisons
dot
product
Generate
key, query value
vectors
k_1 q_1 v_1
k_2 q_2 v_2
k_3 q_3 v_3
W^k W^q W^v
x_1
x_2
x_3
target word

Self-Attention

$$A(q, K, V) = \sum_i \frac{e^{q \cdot k_i}}{\sum_j e^{q \cdot k_j}} \times v_i$$

Weight

query and key comparisons

softmax

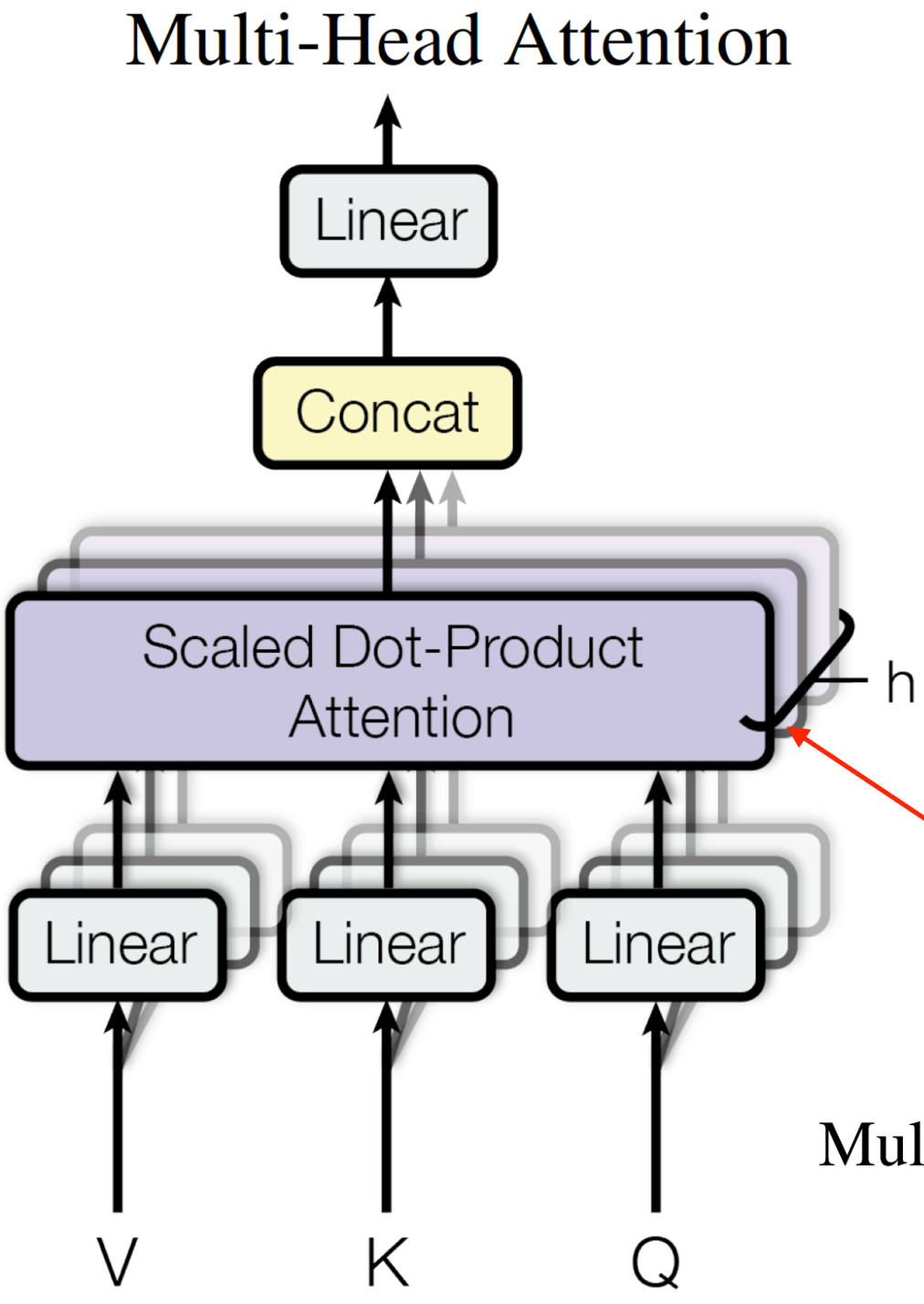
- Multiple queries, stack them in a matrix

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

- Uses **scaled dot-product** to prevent values from growing too large

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

dimension of query and key vectors



- Only one attention for each word pair
- Uses multi-head attention to allow multiple interactions

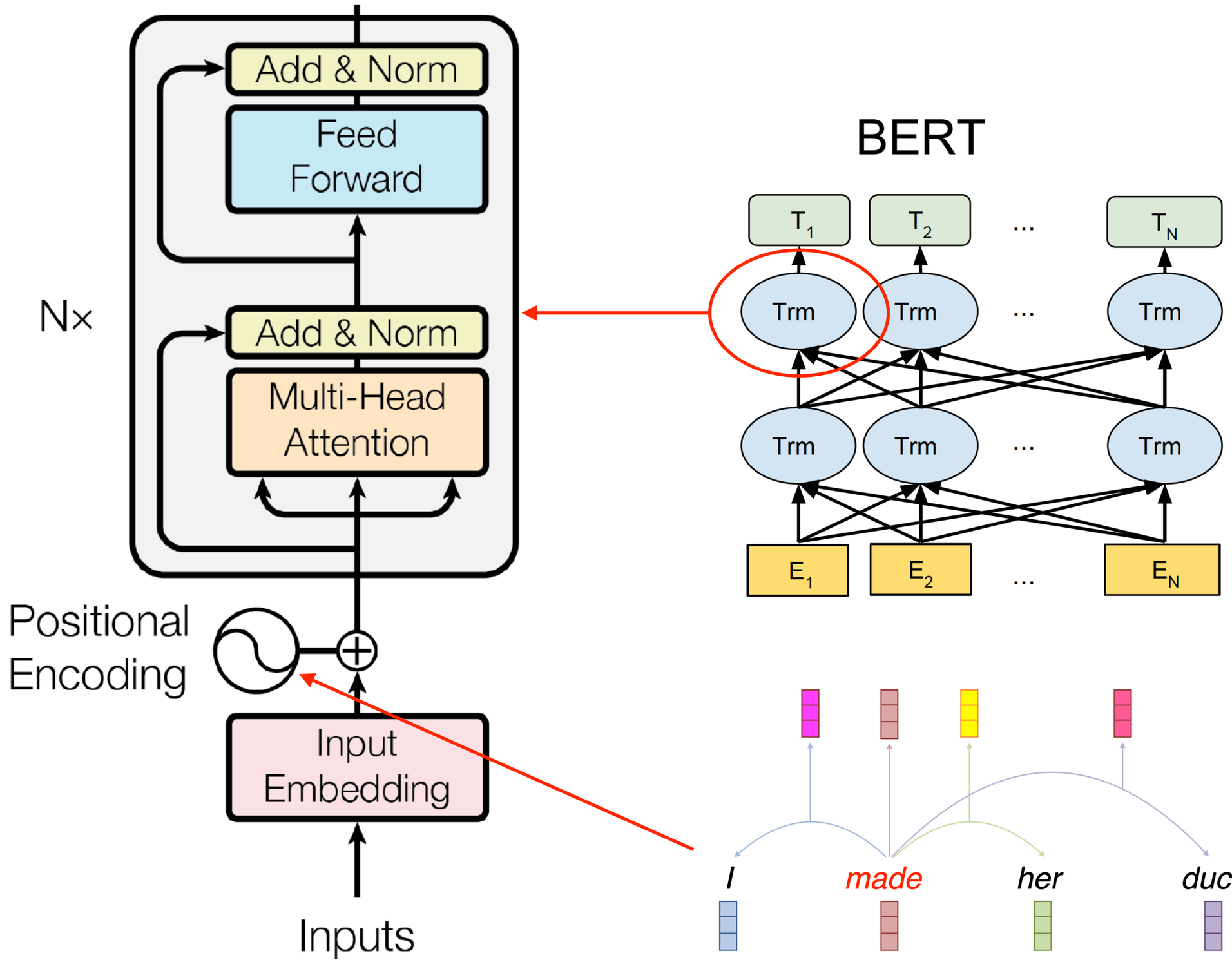
$$A(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$$\text{MultiHead}(Q, K, V) = \text{concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

linear projection of Q/K/V for each head

Transformer Block



A Final Word

- Contextual representations are very useful
- Pre-trained on very large corpus
  - Learned some knowledge about language
  - Uses unsupervised objectives
- When we use them for downstream tasks, we are no longer starting from “scratch”

Further Reading

- ELMo: https://arxiv.org/abs/1802.05365
- BERT: https://arxiv.org/abs/1810.04805
- Transformer: http://nlp.seas.harvard.edu/2018/04/03/attention.html