Neural Language Models and BERT

CS 485, Fall 2023
Applications of Natural Language Processing
https://people.cs.umass.edu/~brenocon/cs485_f23/

Brendan O'Connor
College of Information and Computer Sciences
University of Massachusetts Amherst

including slides from Mohit Iyyer and Emma Strubell

Language modeling as representation learning

- Train a skip-gram LM ->
  get useful *word embeddings*

- Today: train a (funny) LM ->
  get useful *token embeddings*

Why contextual embeddings?

BERT

- “Bidirectional… Transformers”
  - Transformer: a specific neural net architecture for token sequences, that uses attention and token embeddings
  - Bidirectional: The core model is a ***masked LM***, predicting missing word(s) from rest of words in sentence
- Usage
  - 1. "Pretrain": train it as a masked language model on a large corpus
    - It learns to infer useful *contextual word embeddings* per token
  - 2. "Fine-tune": apply it for your desired supervised learning task. (Further update the parameters to do well at your task.)
- BERT (+ variants) are incredibly successful
  - ... and it learns useful linguistic structure by itself! Rogers et al., 2020
  - ... and there are easy to use implementations: https://huggingface.co/docs/transformers/index

Transformers: Self-attention

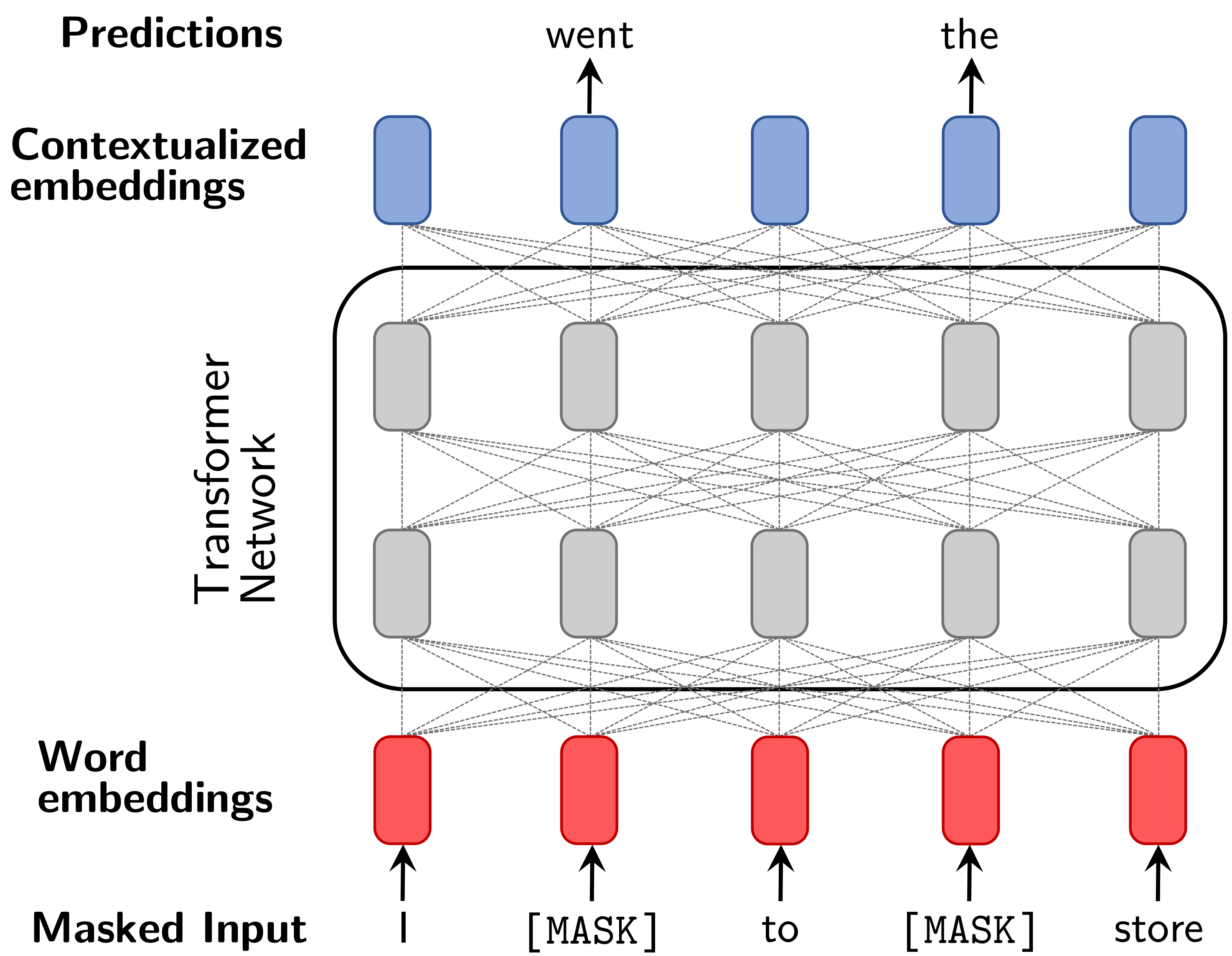


Fig. 3. A high-level illustration of BERT. Words in the input sequence are randomly masked out and then all words are embedded as vectors in $\mathbb{R}^d$. A Transformer network applies multiple layers of multiheaded attention to the representations. The final representations are used to predict the identities of the masked-out input words.

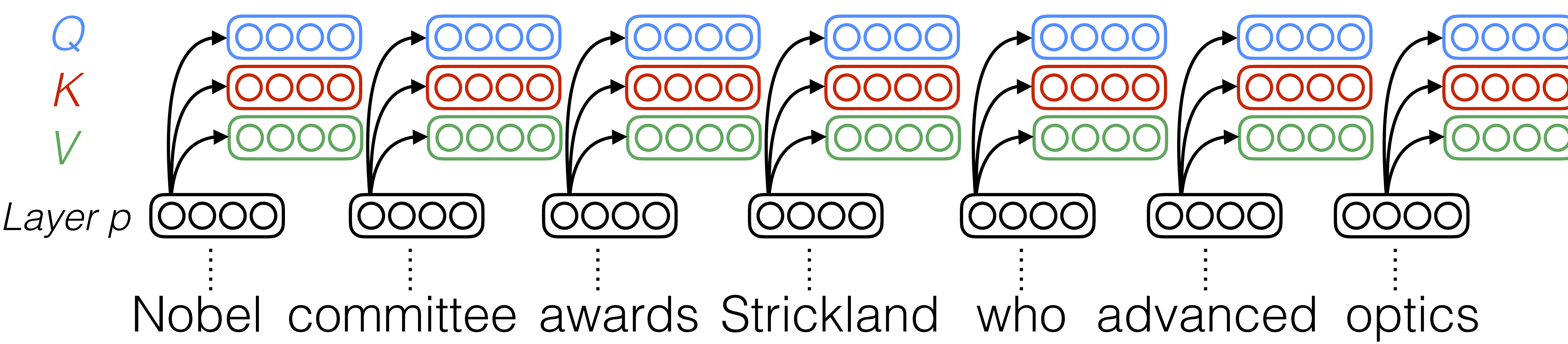
[Manning et al., 2020]

Attention for Masked LM

Self-attention

[Vaswani et al. 2017 original notation, slide: Emma Strubell]

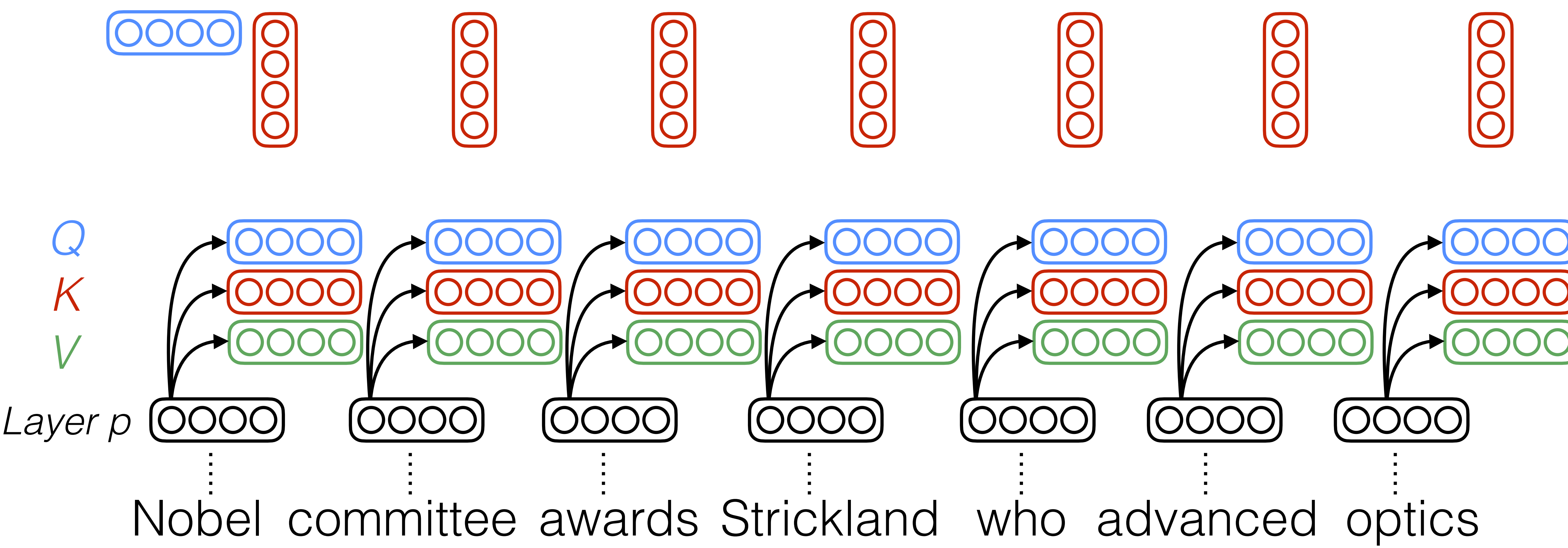
$$\mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



Self-attention

[Vaswani et al. 2017 original notation, slide: Emma Strubell]

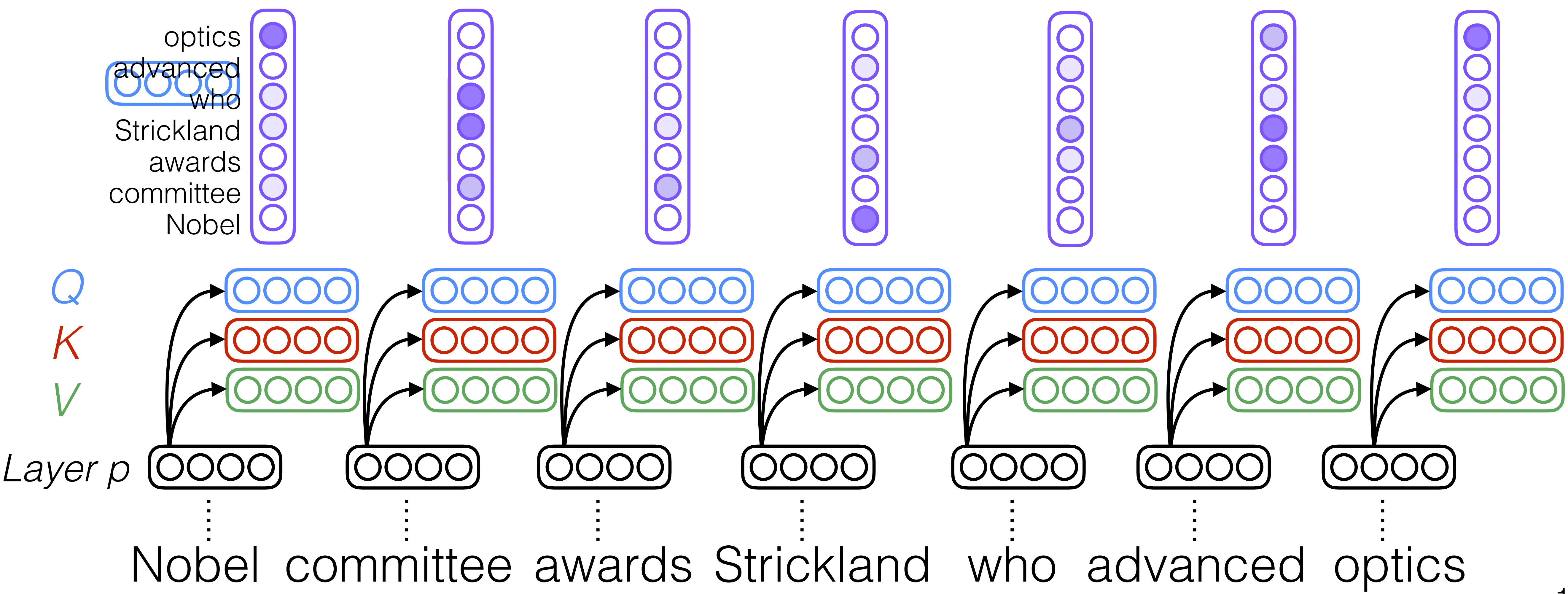
$$\mathrm{softmax}\left(\frac{\boxed{QK^T}}{\sqrt{d_k}}\right)V$$



Self-attention

[Vaswani et al. 2017
original notation,
slide: Emma Strubell]

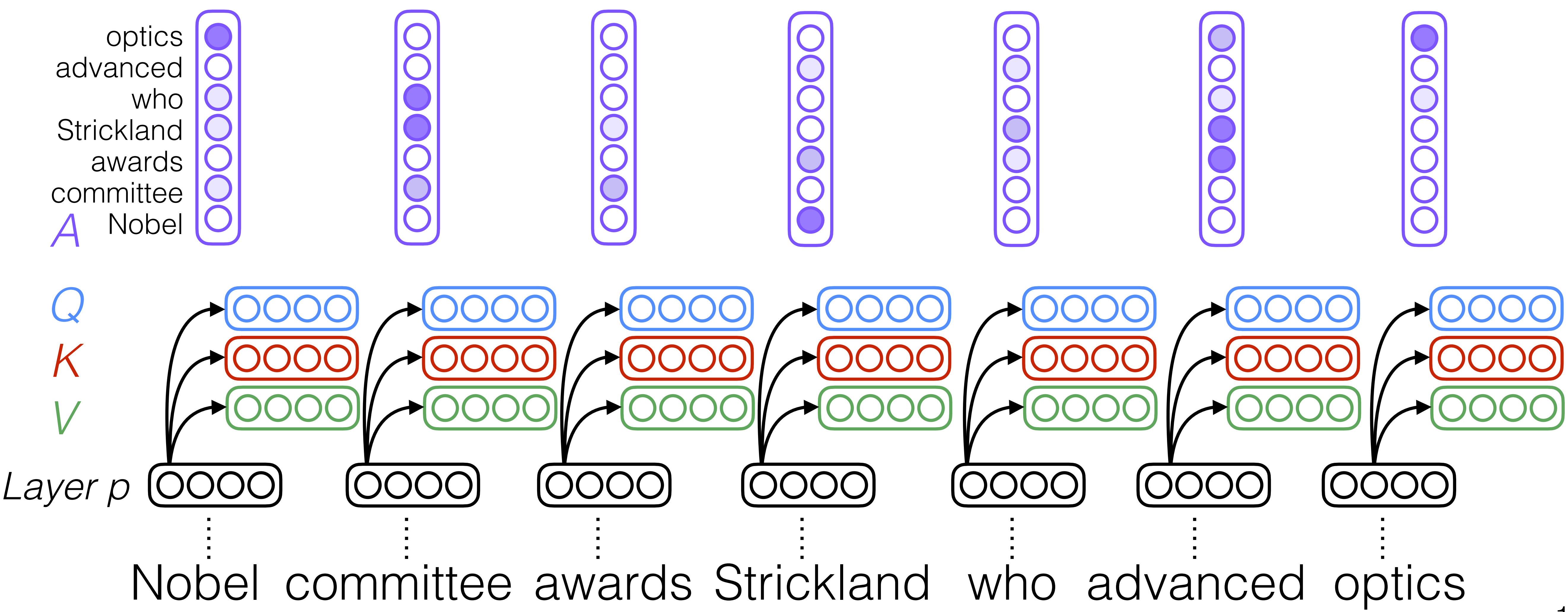
$$\boxed{\mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V}$$



Self-attention

[Vaswani et al. 2017 original notation, slide: Emma Strubell]

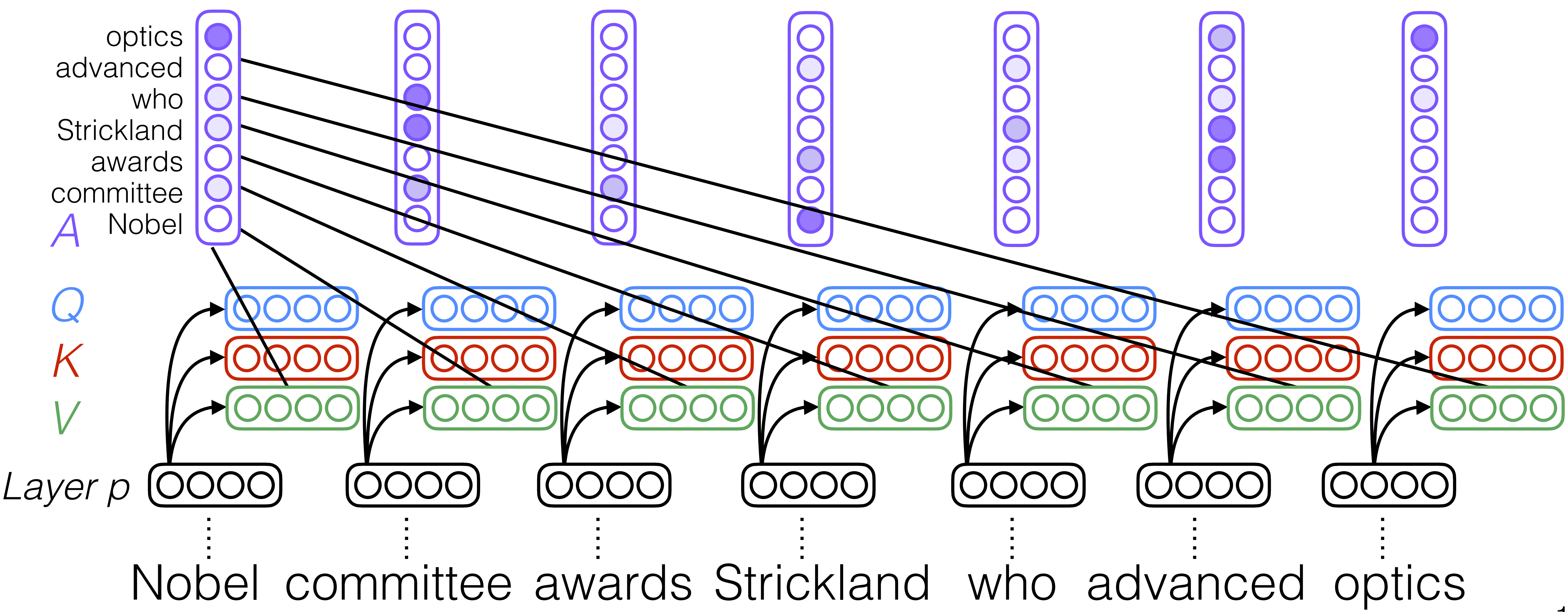
$$\boxed{\mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})}V$$



Self-attention

[Vaswani et al. 2017 original notation, slide: Emma Strubell]

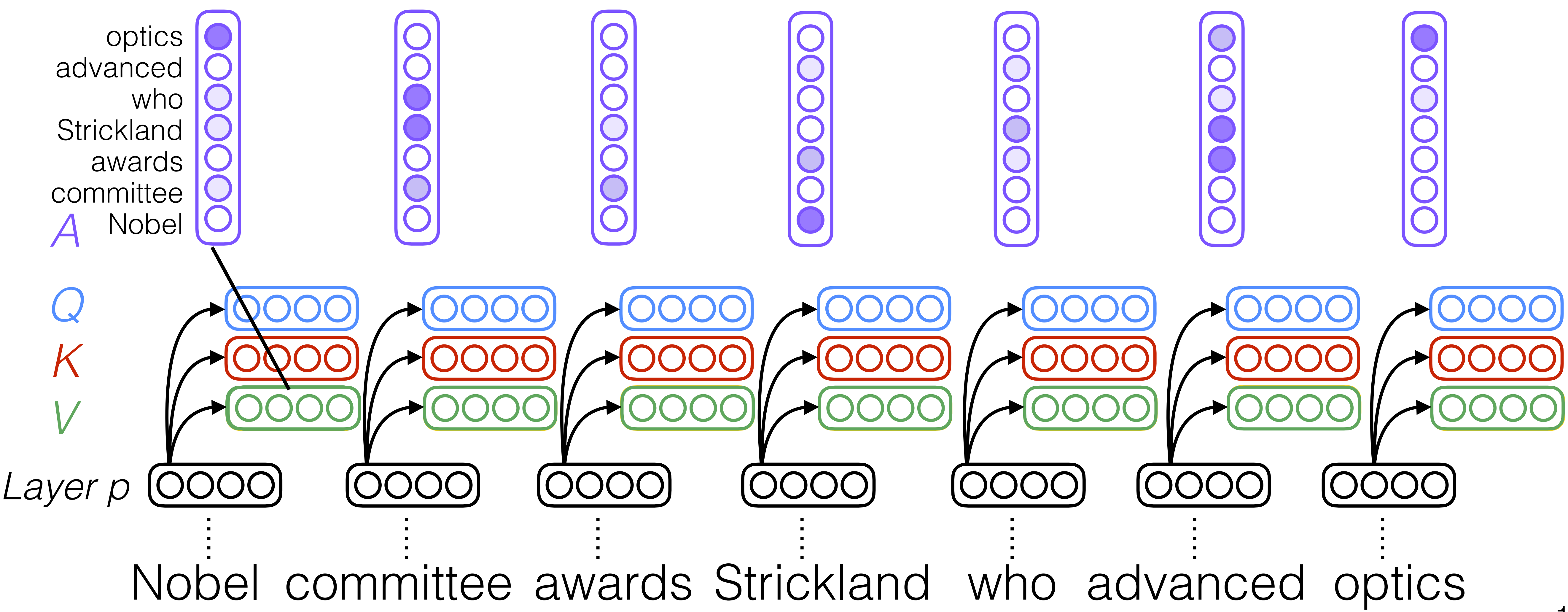
$$\boxed{\mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V}$$



Self-attention

[Vaswani et al. 2017 original notation, slide: Emma Strubell]

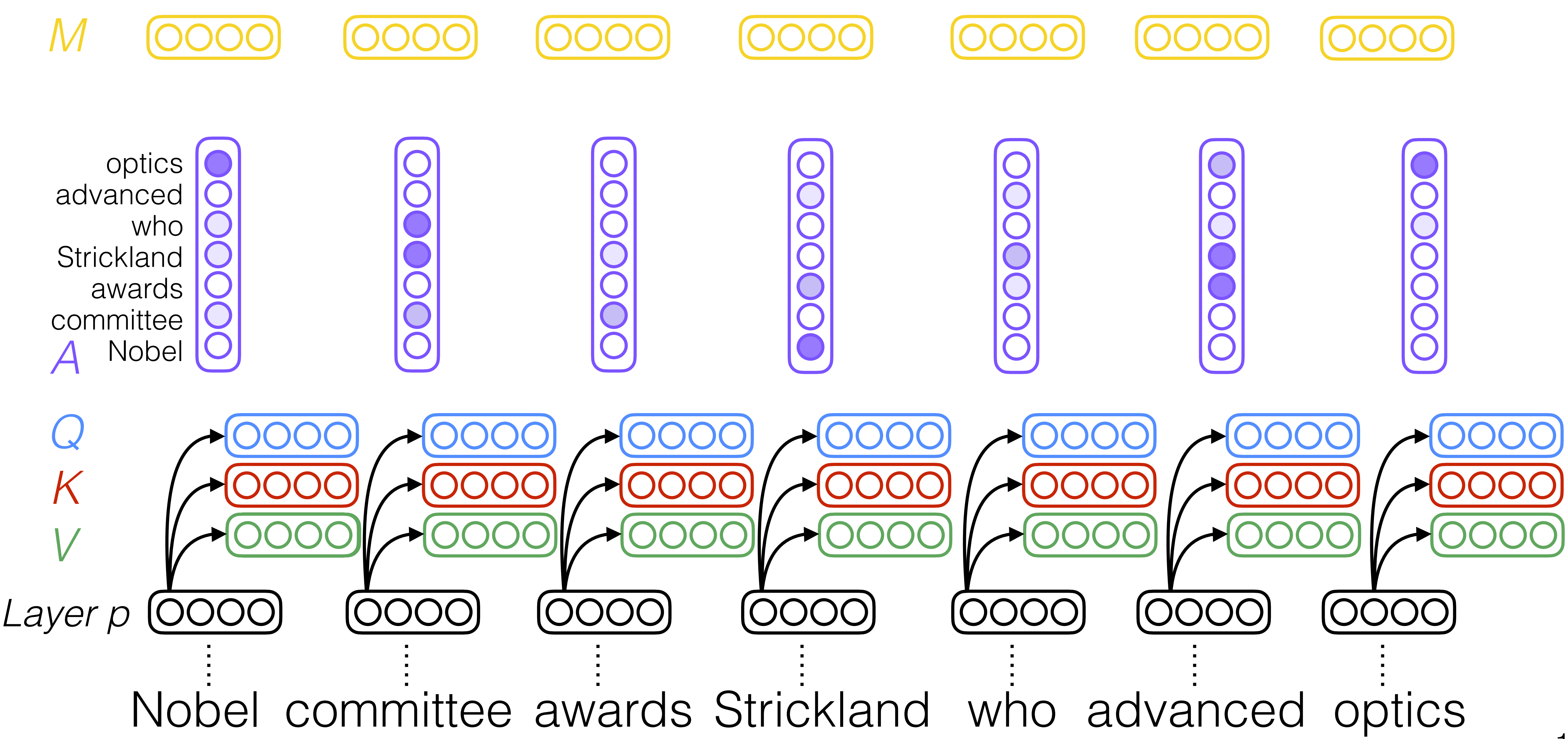
$$\mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



Self-attention

[Vaswani et al. 2017 original notation, slide: Emma Strubell]

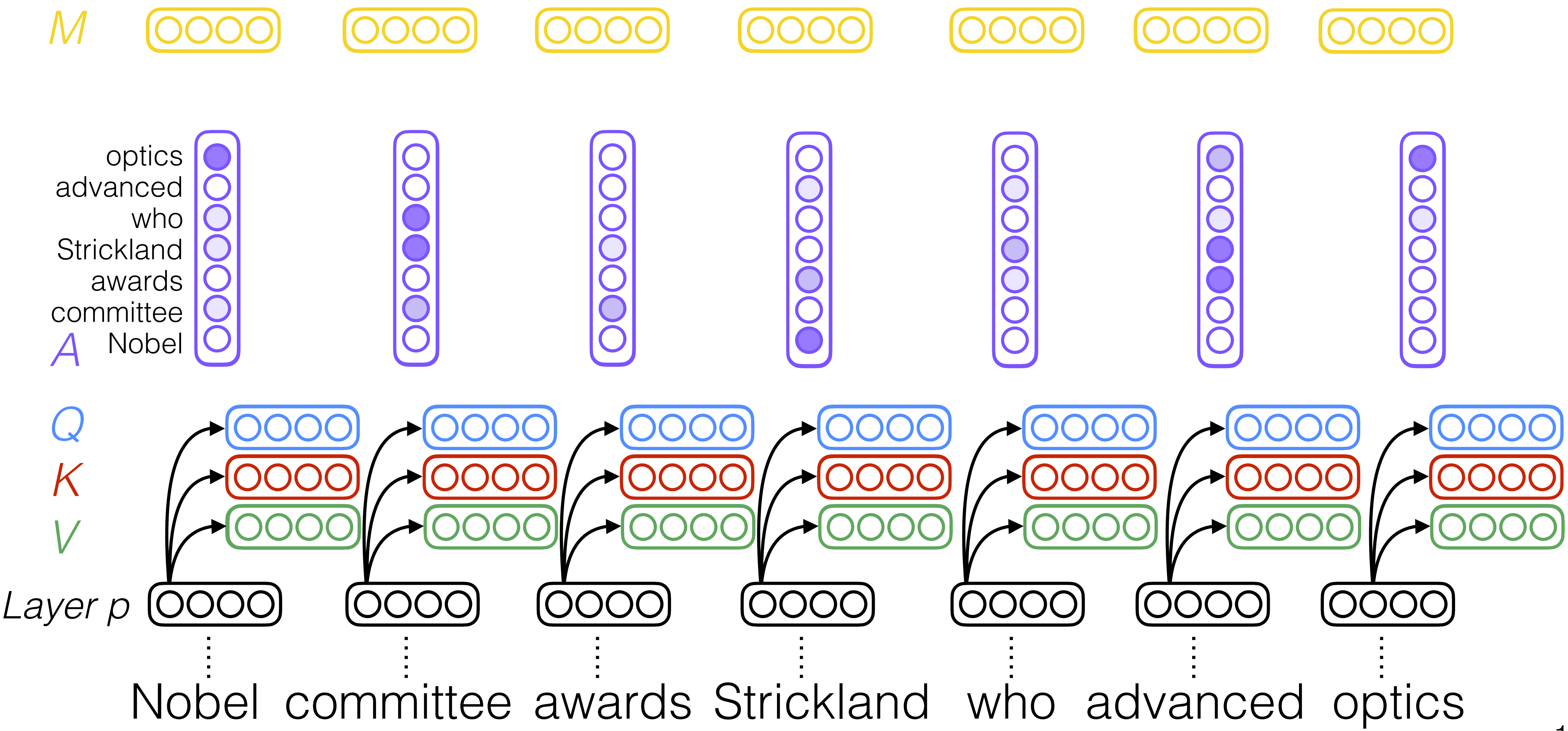
$$\boxed{\mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V}$$



Self-attention

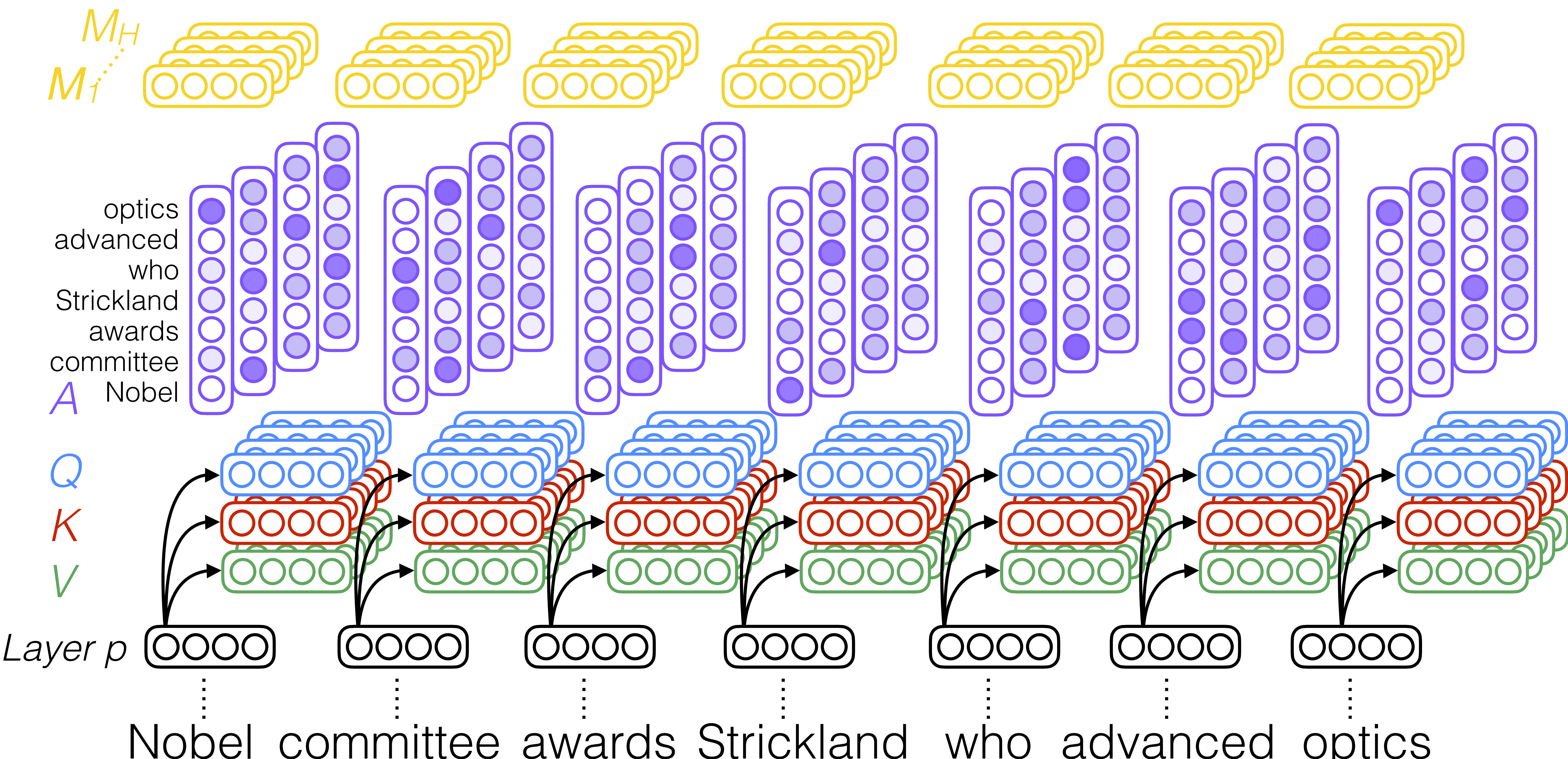
[Vaswani et al. 2017 original notation, slide: Emma Strubell]

$$\boxed{\mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V}$$



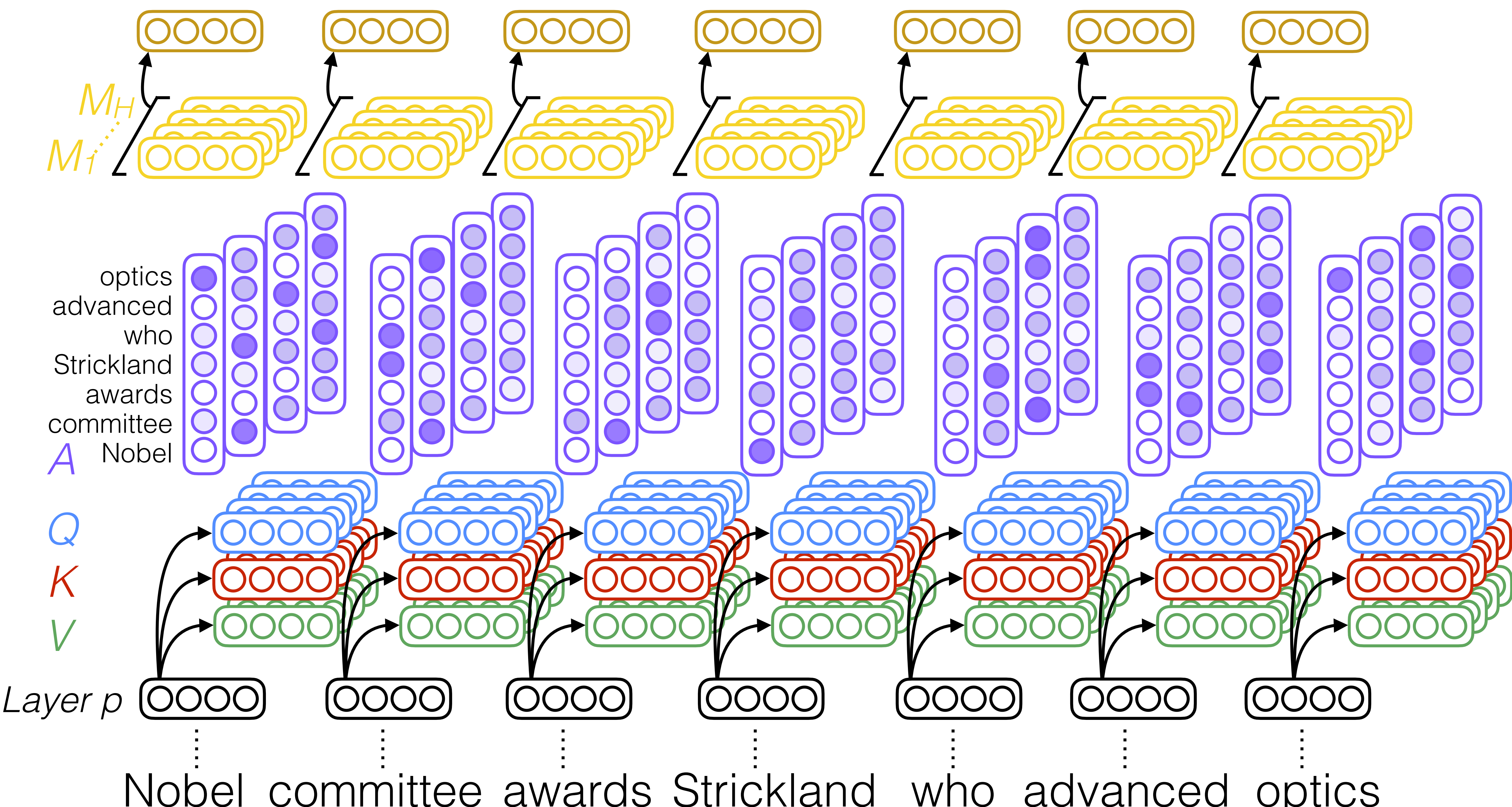
Multi-head self-attention

[Vaswani et al. 2017 original notation]



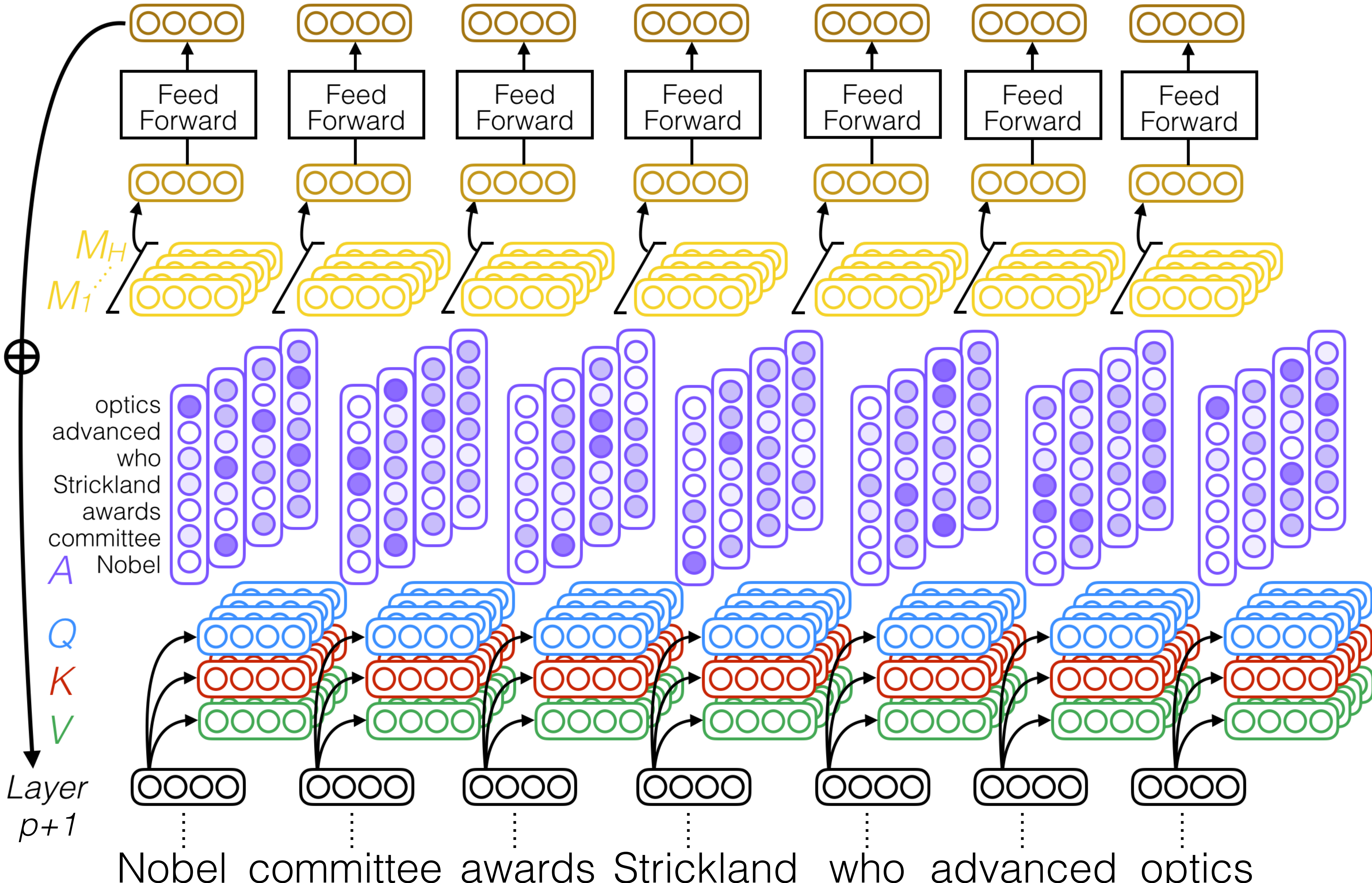
Multi-head self-attention

[Vaswani et al. 2017 original notation]



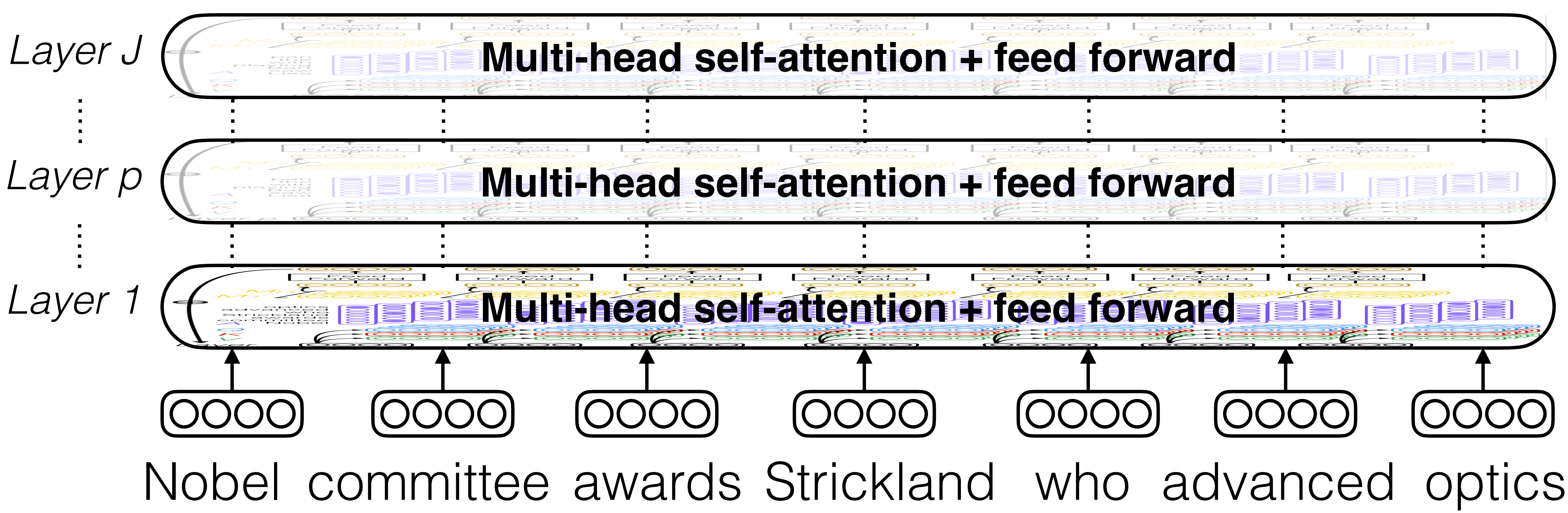
Multi-head self-attention

[Vaswani et al. 2017 original notation]



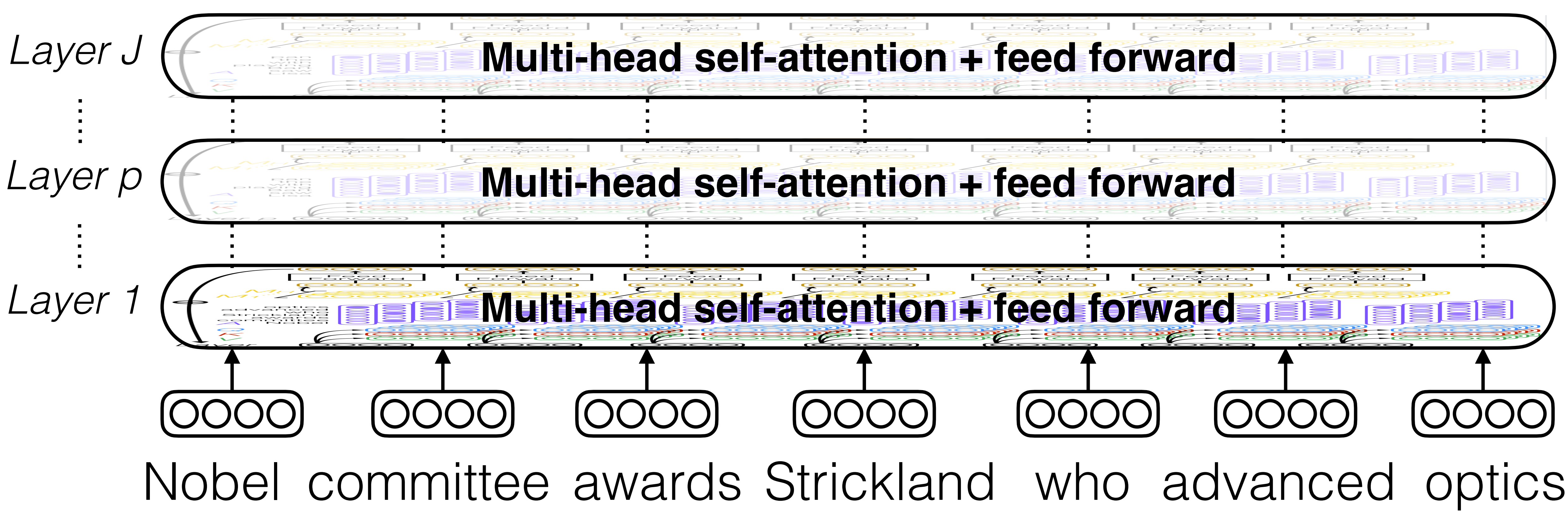
Multi-head self-attention

[Vaswani et al. 2017 original notation]

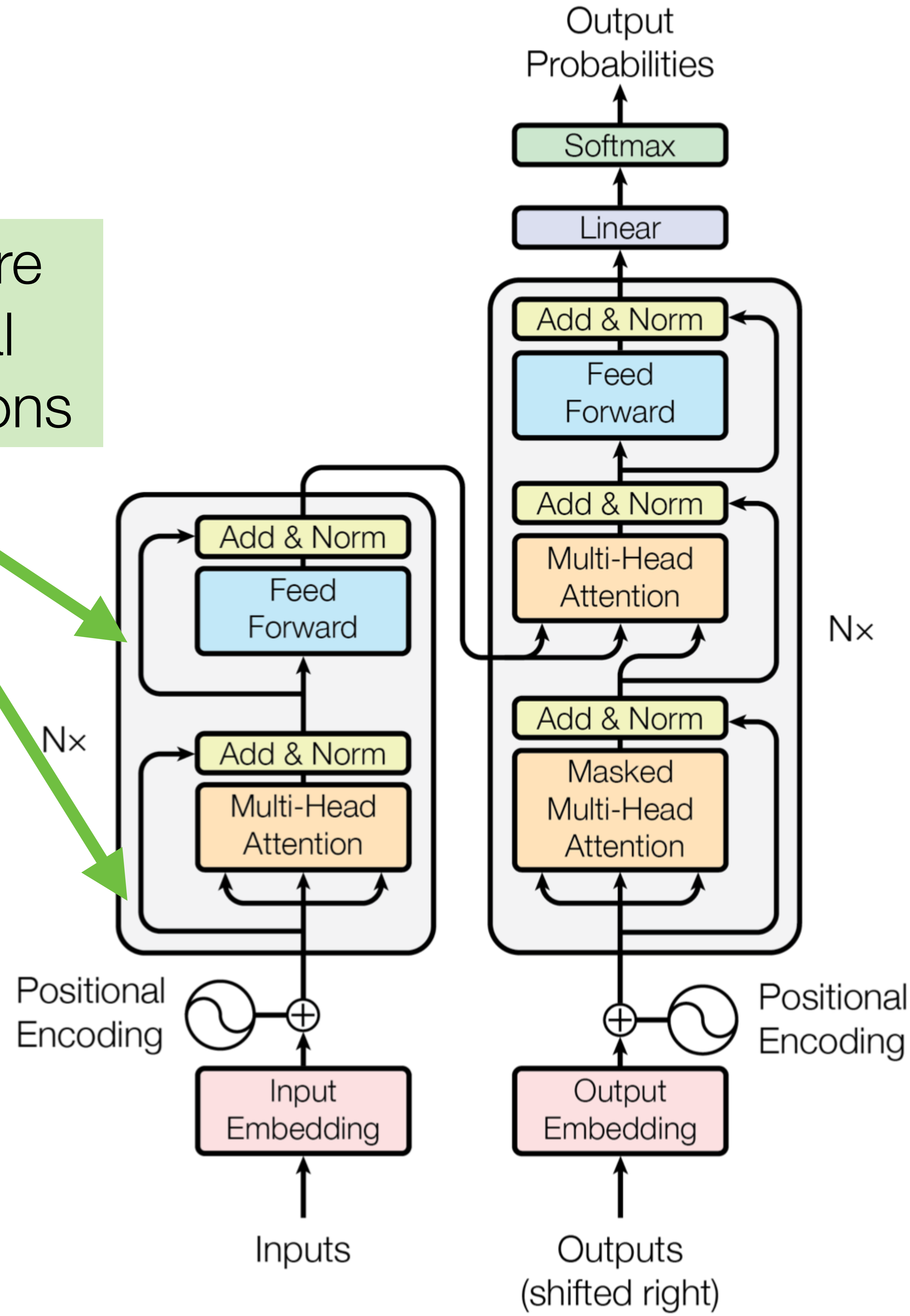


Multi-head self-attention

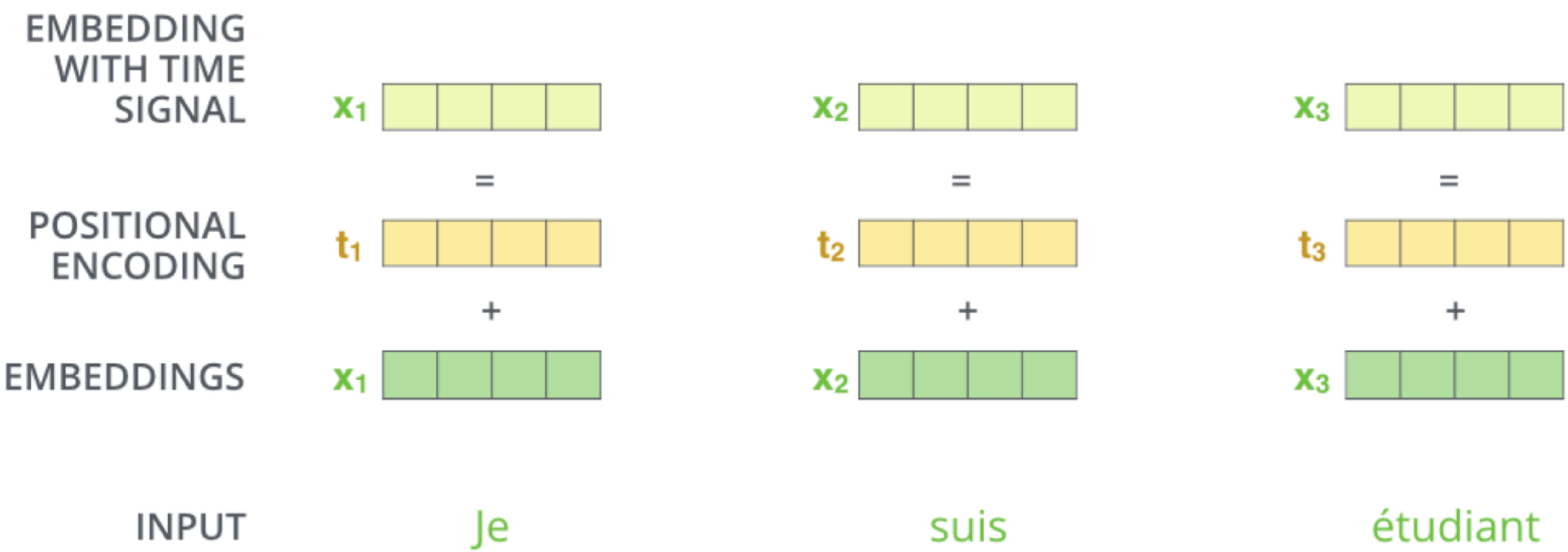
[Vaswani et al. 2017 original notation]



These are residual connections



Positional encoding

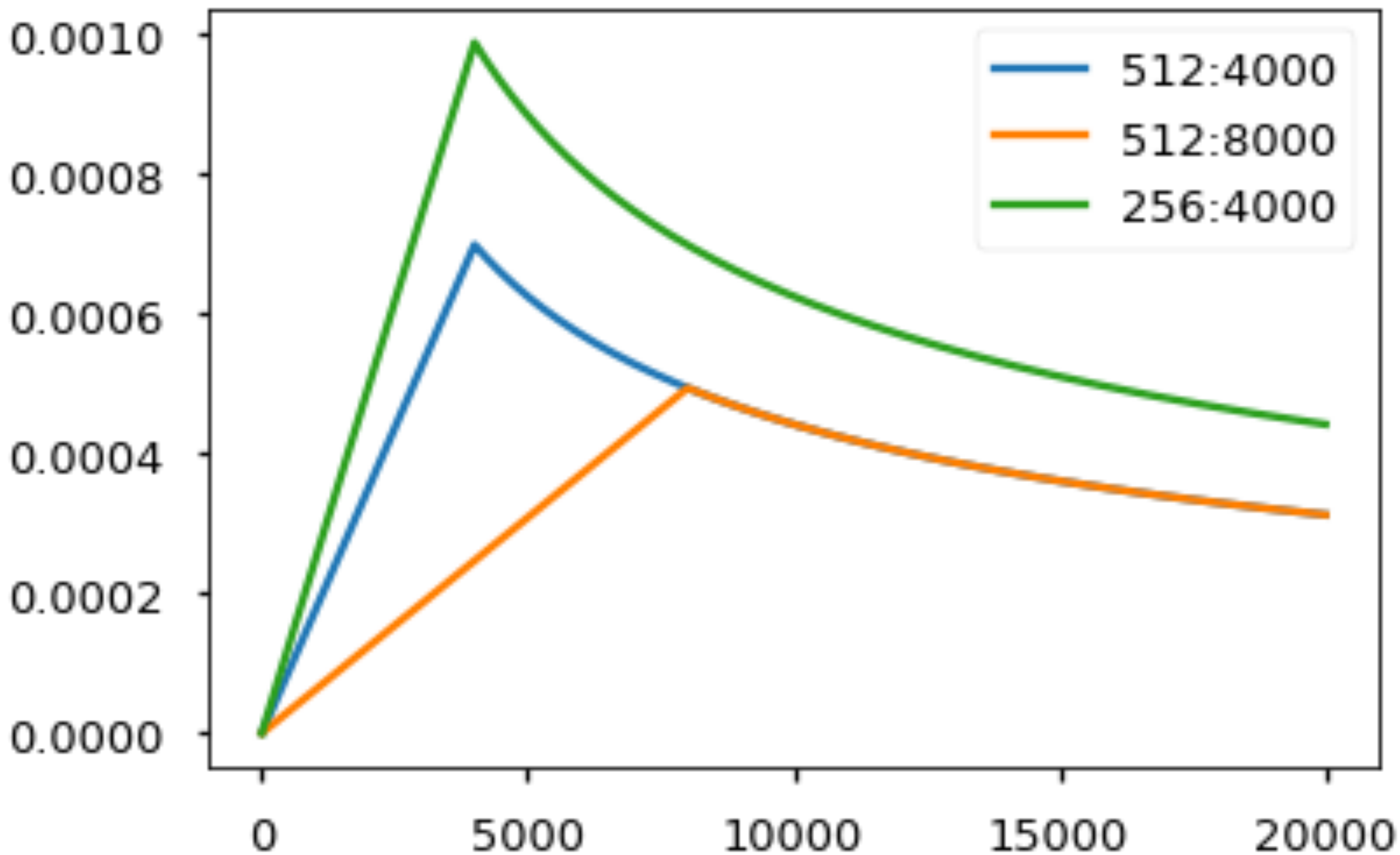


Hacks to get it to work:

Optimizer

We used the Adam optimizer (cite) with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-9}$. We varied the learning rate over the course of training, according to the formula: $lrate = d_{\text{model}}^{-0.5} \cdot \min(step\_num^{-0.5}, step\_num \cdot warmup\_steps^{-1.5})$ This corresponds to increasing the learning rate linearly for the first $warmup_steps$ training steps, and decreasing it thereafter proportionally to the inverse square root of the step number. We used $warmup_steps = 4000$.

> *Note: This part is very important. Need to train with this setup of the model.*



Label Smoothing

During training, we employed label smoothing of value $\epsilon_{ls} = 0.1$ (cite). This hurts perplexity, as the model learns to be more unsure, but improves accuracy and BLEU score.

> *We implement label smoothing using the KL div loss. Instead of using a one-hot target distribution, we create a distribution that has* `confidence` *of the correct word and the rest of the* `smoothing` *mass distributed throughout the vocabulary.*

I went to class and took ___

cats	TV	notes	took	sofa
0	0	1	0	0
0.025	0.025	0.9	0.025	0.025

with label smoothing

Byte pair encoding (BPE)

- Deal with rare words / large vocabulary by using *subword* tokenization
  - Initial analysis step iteratively merges frequent character n-grams to form the vocabulary
  - Confusing name comes from data compression literature - not actually about bytes for us

system	sentence
source	health research institutes
reference	Gesundheitsforschungsinstitute
WDict	Forschungsinstitute
C2-50k	Fo|rs|ch|un|gs|in|st|it|ut|io|ne|n
BPE-60k	Gesundheits|forsch|ungsinstitu|ten
BPE-J90k	Gesundheits|forsch|ungsin|stitute
source	asinine situation
reference	dumme Situation
WDict	asinine situation → UNK → asinine
C2-50k	as|in|in|e situation → As|in|en|si|tu|at|io|n
BPE-60k	as|in|ine situation → A|in|line-|Situation
BPE-J90K	as|in|ine situation → As|in|in-|Situation

Sennrich et al., ACL 2016

What does BERT learn?

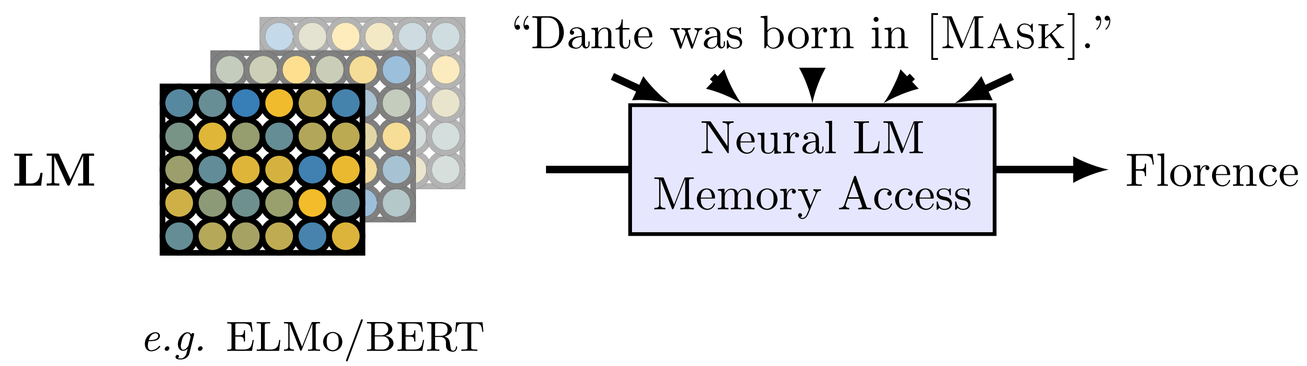


Figure 2: BERT world knowledge (Petroni et al., 2019).

[Rogers et al., 2020]

What does BERT learn?

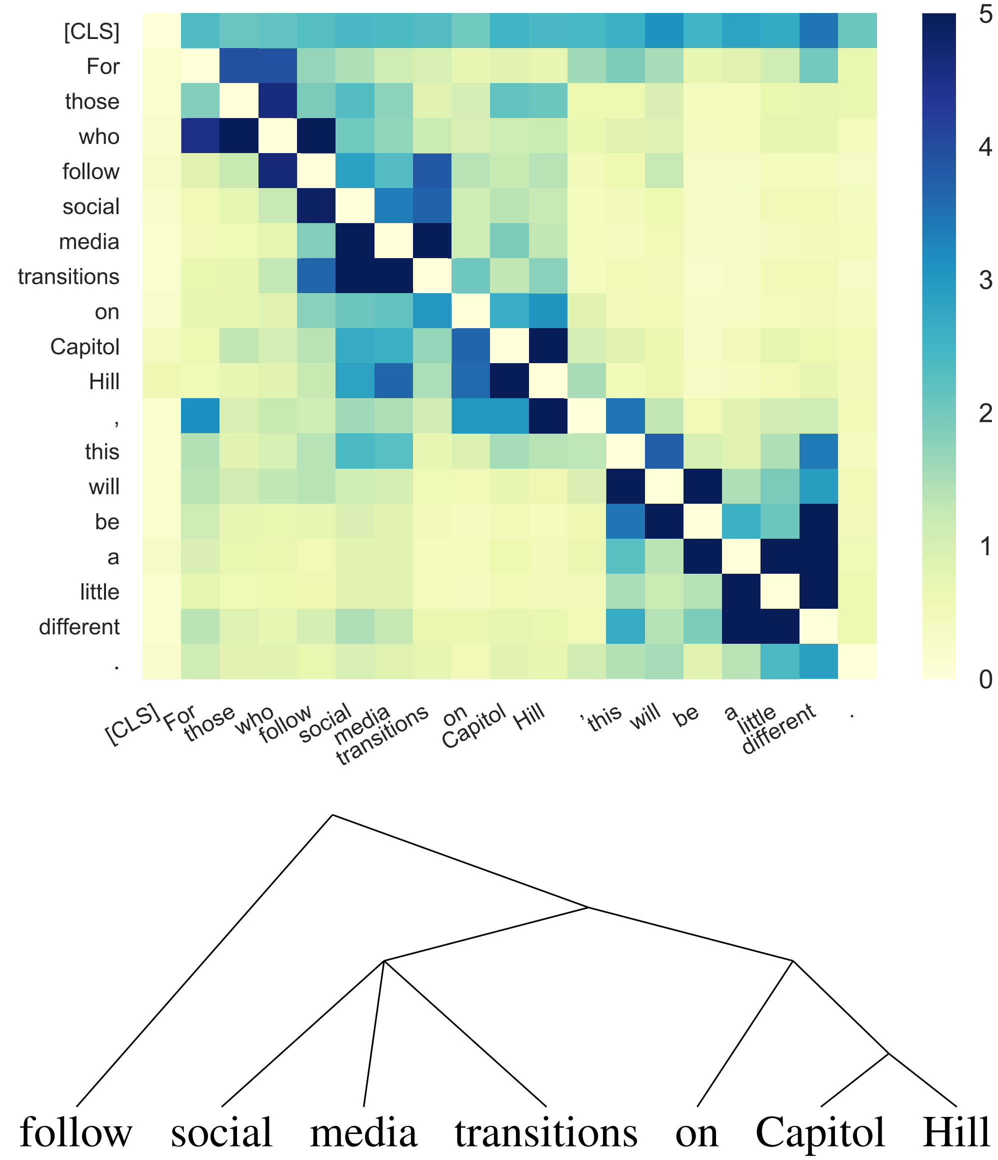


Figure 1: Parameter-free probe for syntactic knowledge: words sharing syntactic subtrees have larger impact on each other in the MLM prediction (Wu et al., 2020).

[Rogers et al., 2020]

What does BERT learn?

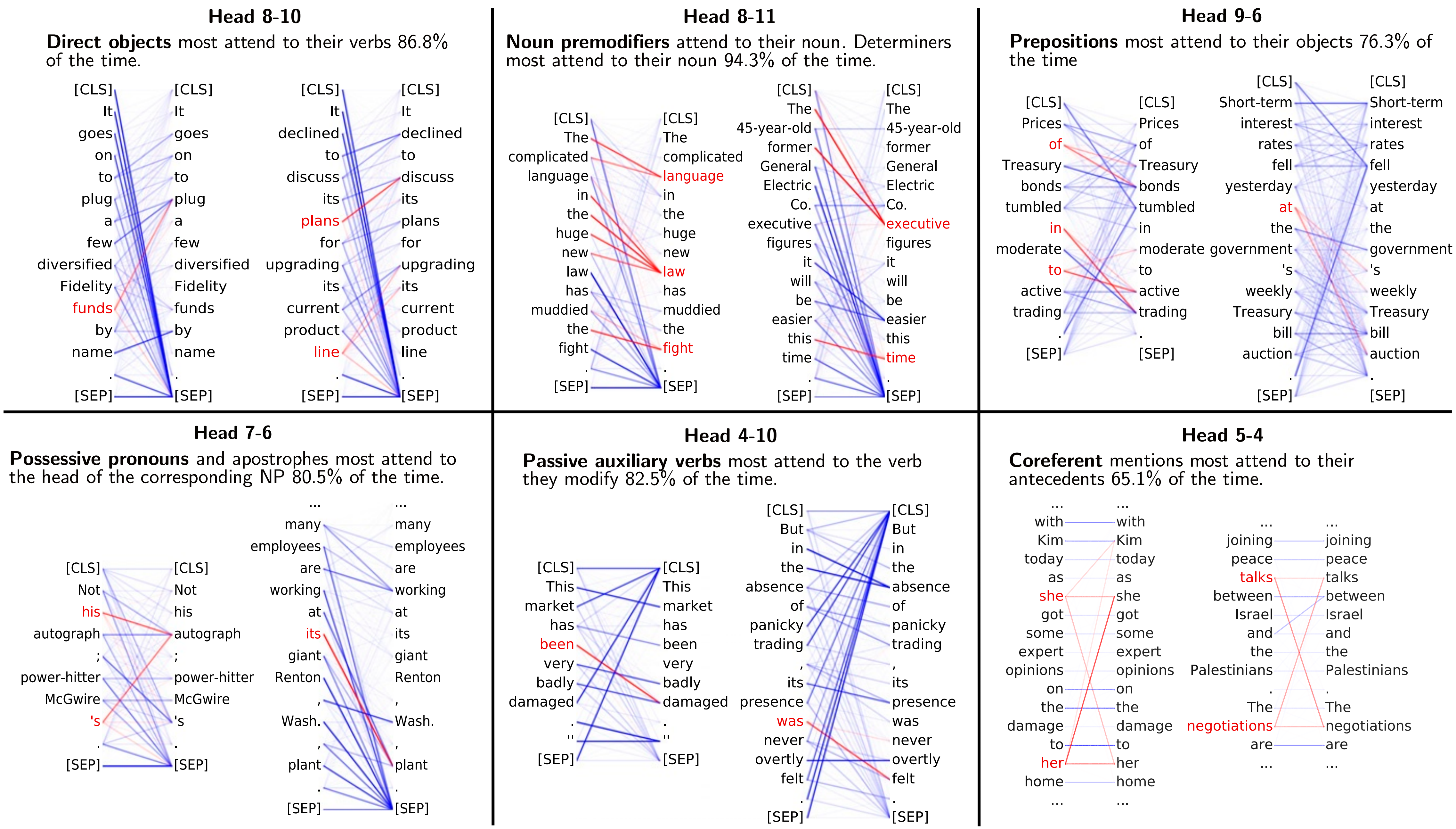


Fig. 6. Some BERT attention heads that appear sensitive to linguistic phenomena, despite not being explicitly trained on linguistic annotations. In the example attention maps, the darkness of a line indicates the size of the attention weight. All attention to/from red words is colored red; these words are chosen to highlight certain of the attention heads' behaviors. [CLS] (classification) and [SEP] (separator) are special tokens BERT adds to the input during preprocessing. Attention heads are numbered by their layer and index in BERT. Reprinted with permission from ref. 59, which is licensed under CC BY 4.0.

[Manning et al., 2020]

Using BERT

- You get
  - Per-token embeddings
  - Multiple layers at each
  - Embedding for per-sentence “[CLS]” symbol
- Use as input for tasks. Two learning approaches
  - “Frozen”: use them as input features
  - Fine-tuning: backprop through the actual BERT model itself (better)

Using BERT for classif.

- Many pretrained BERT or BERT-like models are available (especially for English and other high-resource languages…)
- Check out HuggingFaces' examples
    - https://huggingface.co/transformers/examples.html