Transformers and transfer learning

CS 585, Fall 2019

Mohit Iyyer
College of Information and Computer Sciences
University of Massachusetts Amherst

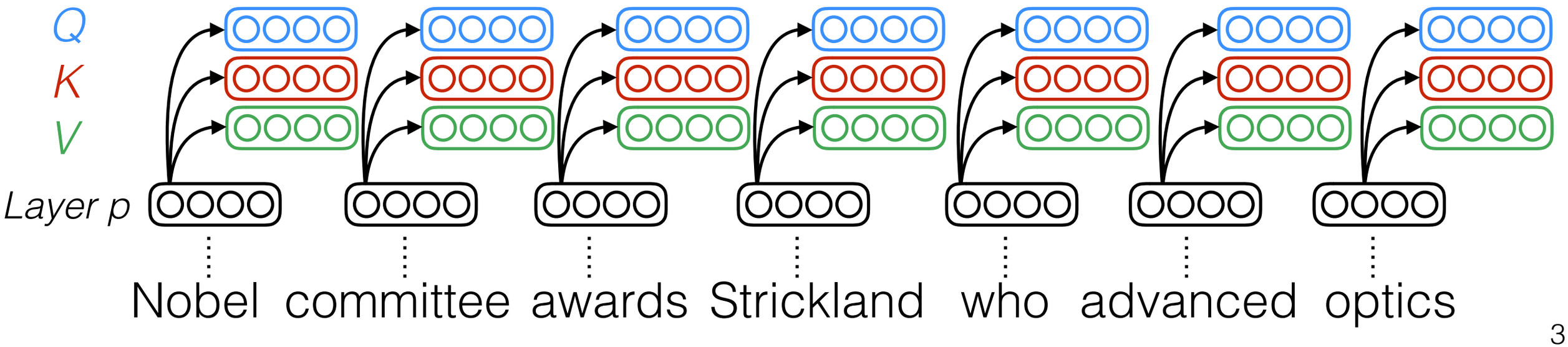
some slides from Emma Strubell, Matt Peters, and Jacob Devlin

stuff from last time

- HW2 out now! please start early, as it is fairly long and may prove difficult to implement
- can you repeat questions asked during class? ok

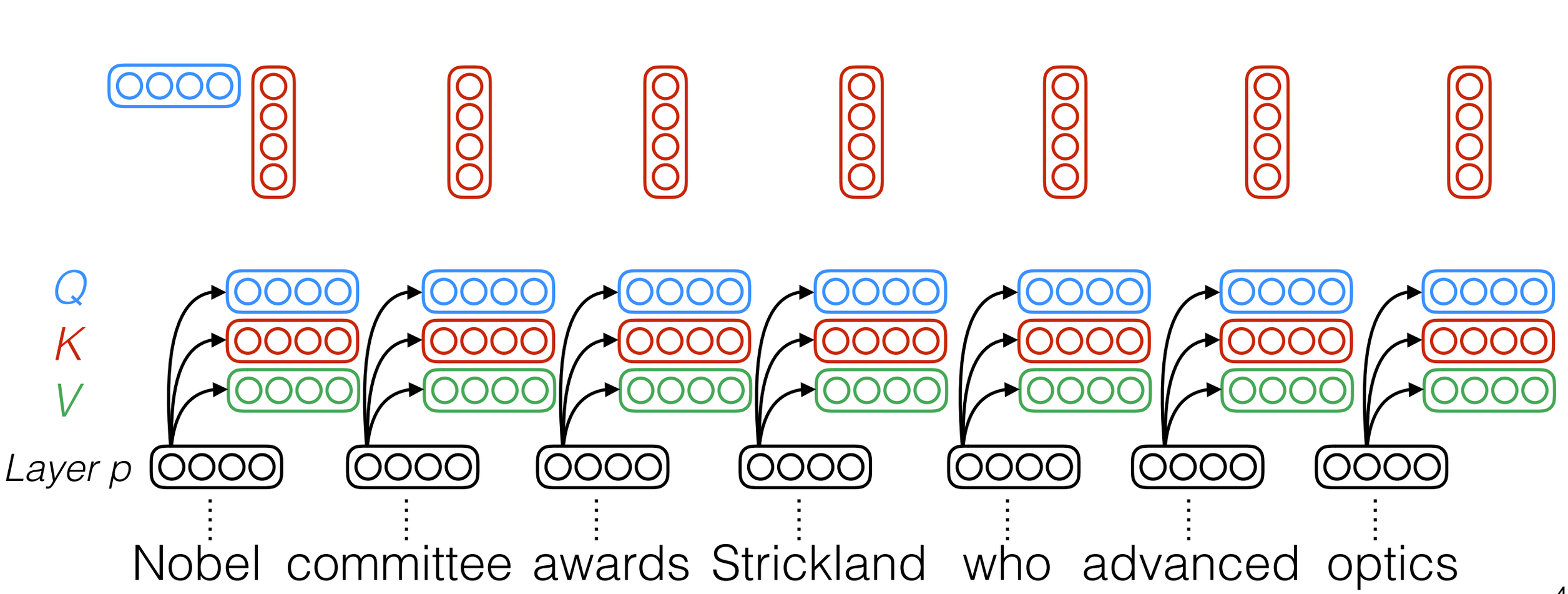
[Vaswani et al. 2017]

Self-attention



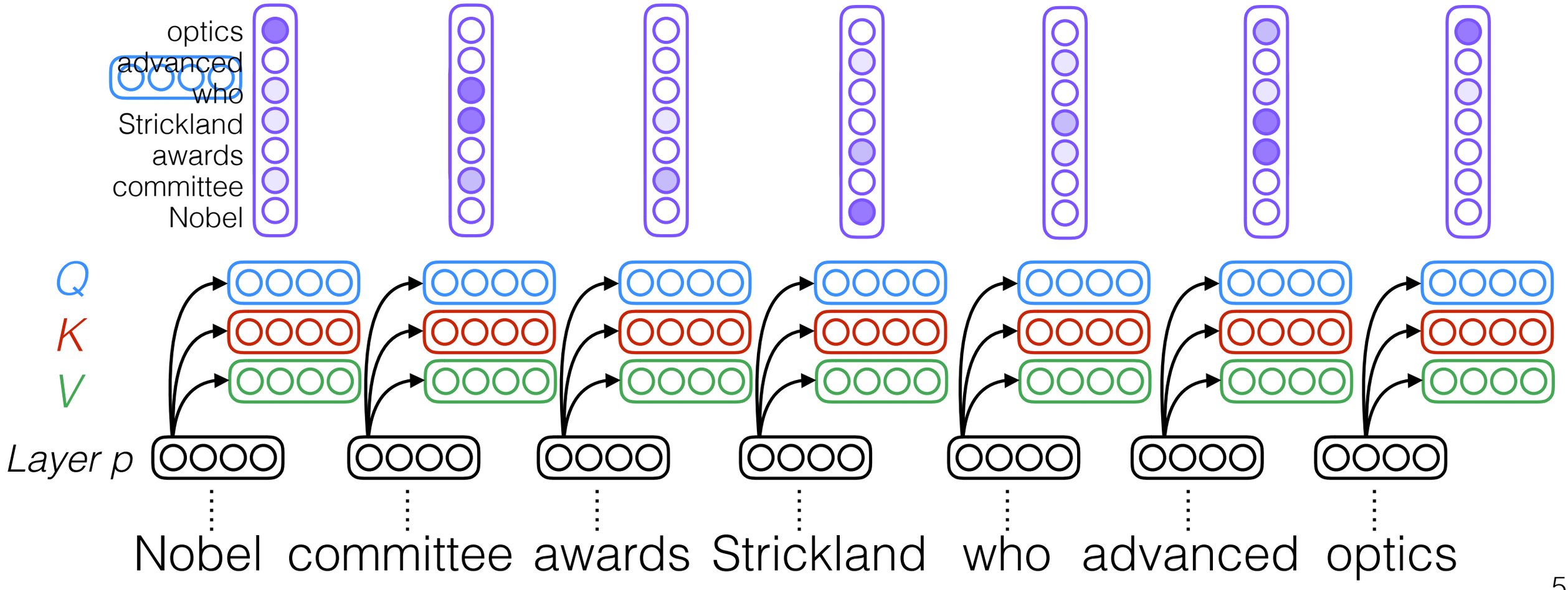
[Vaswani et al. 2017]

Self-attention



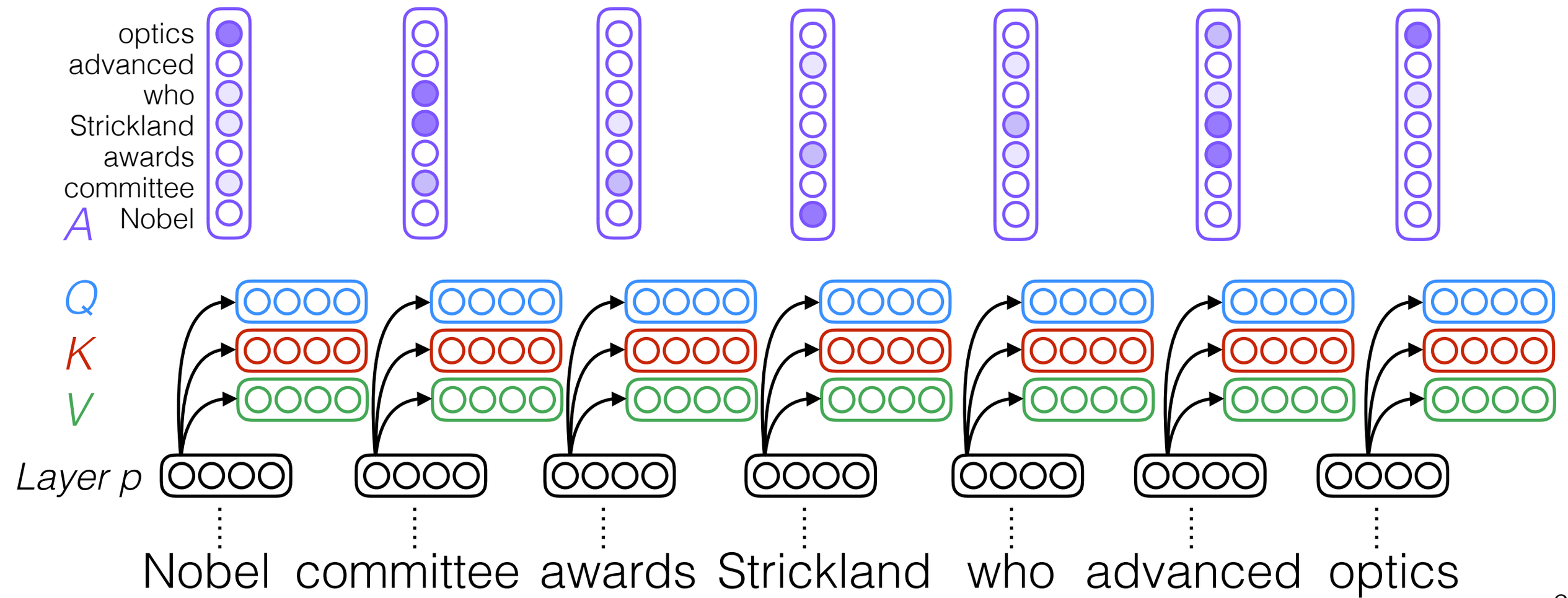
[Vaswani et al. 2017]

Self-attention



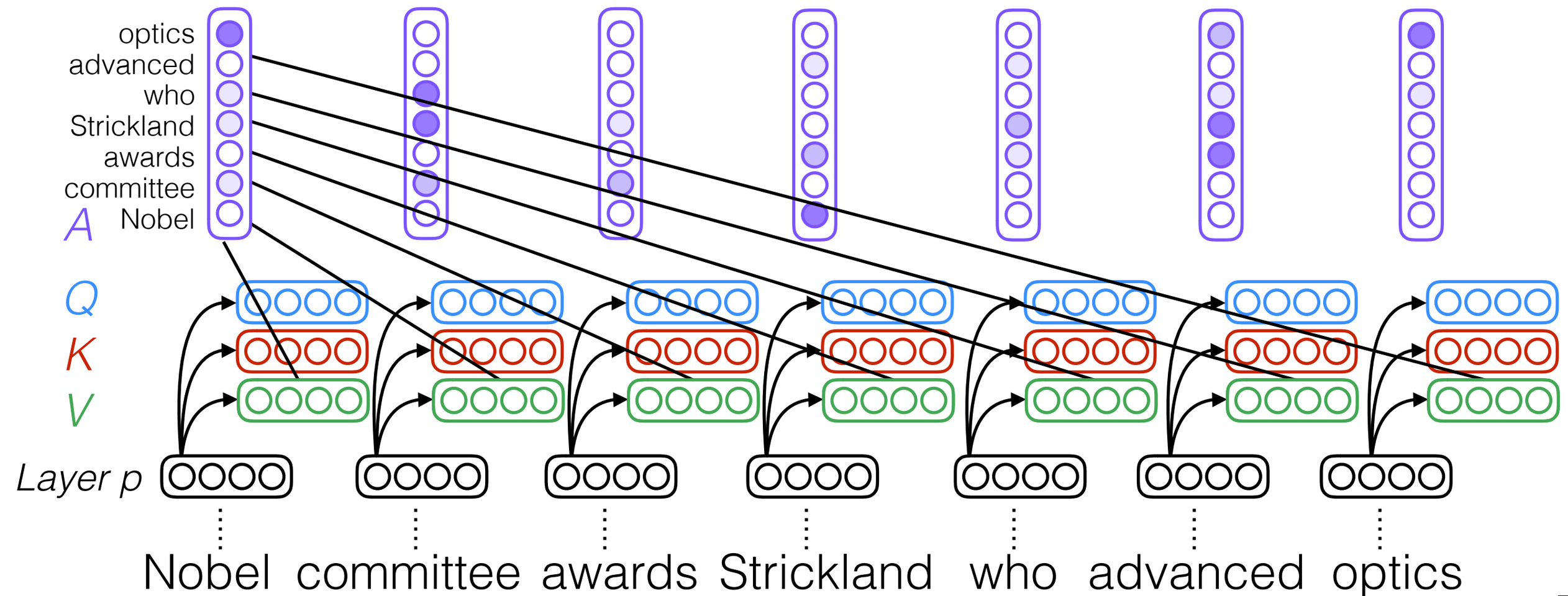
[Vaswani et al. 2017]

Self-attention



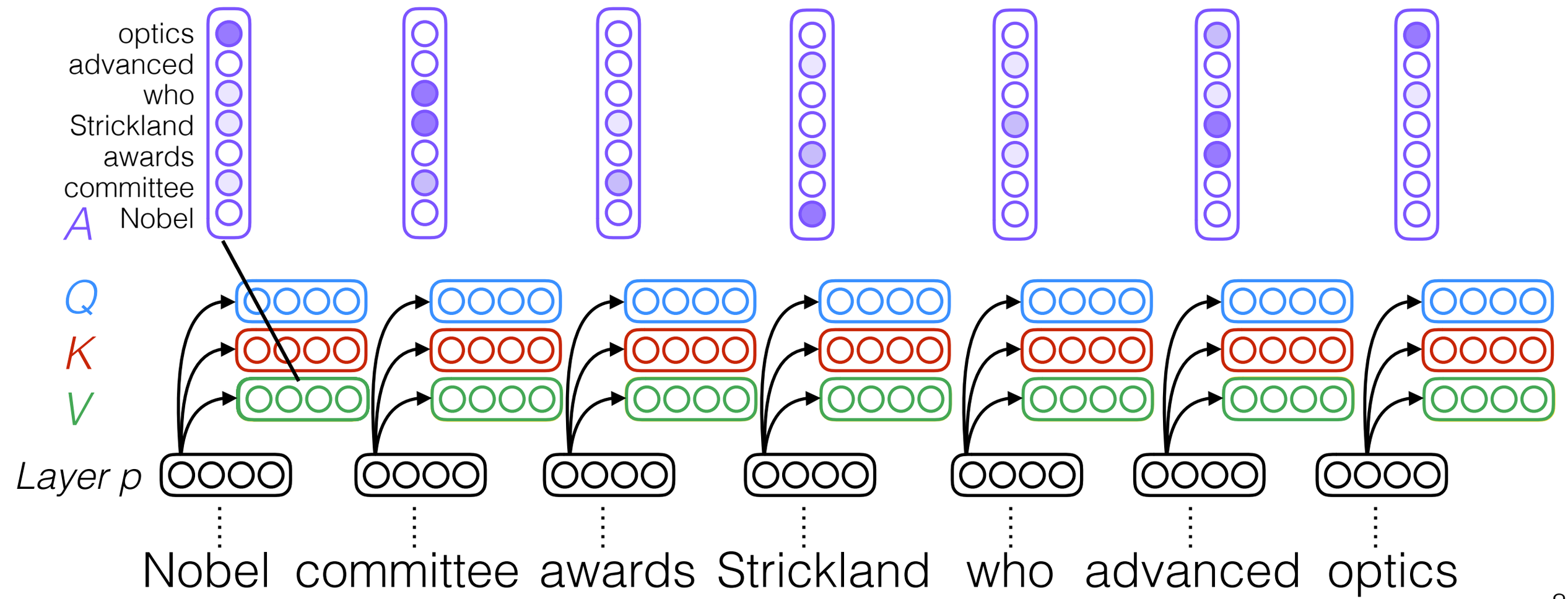
[Vaswani et al. 2017]

Self-attention



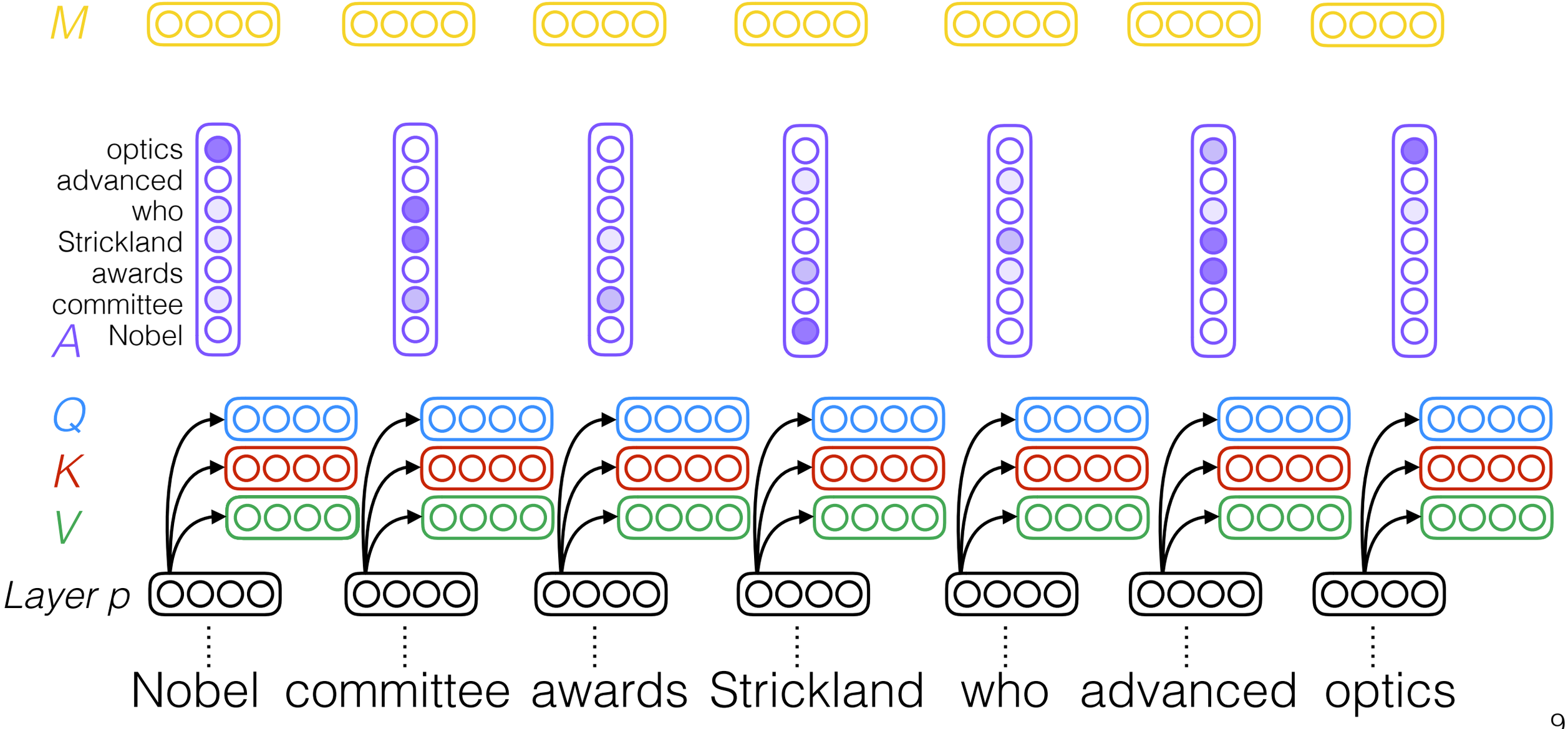
[Vaswani et al. 2017]

Self-attention



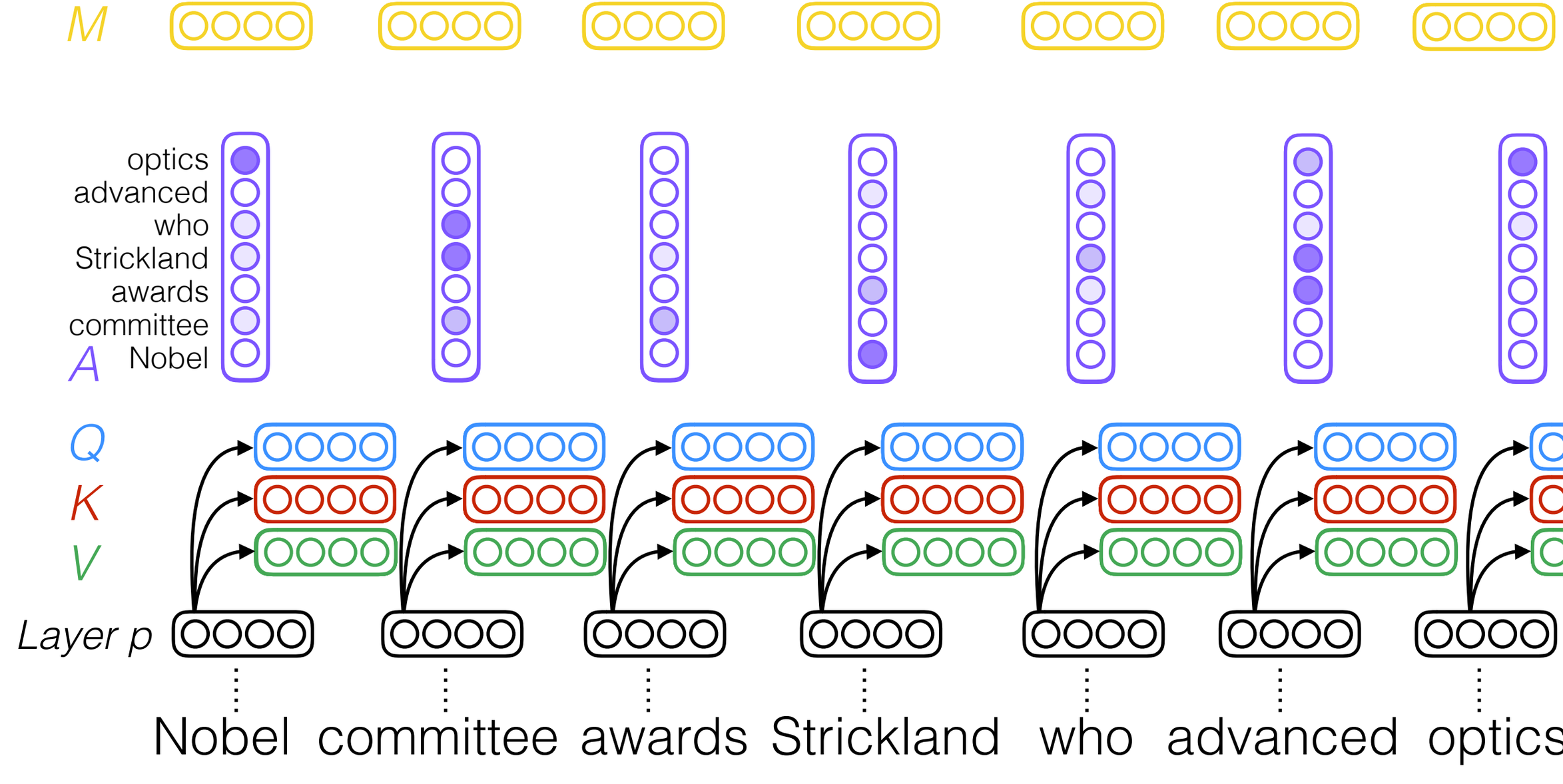
[Vaswani et al. 2017]

Self-attention



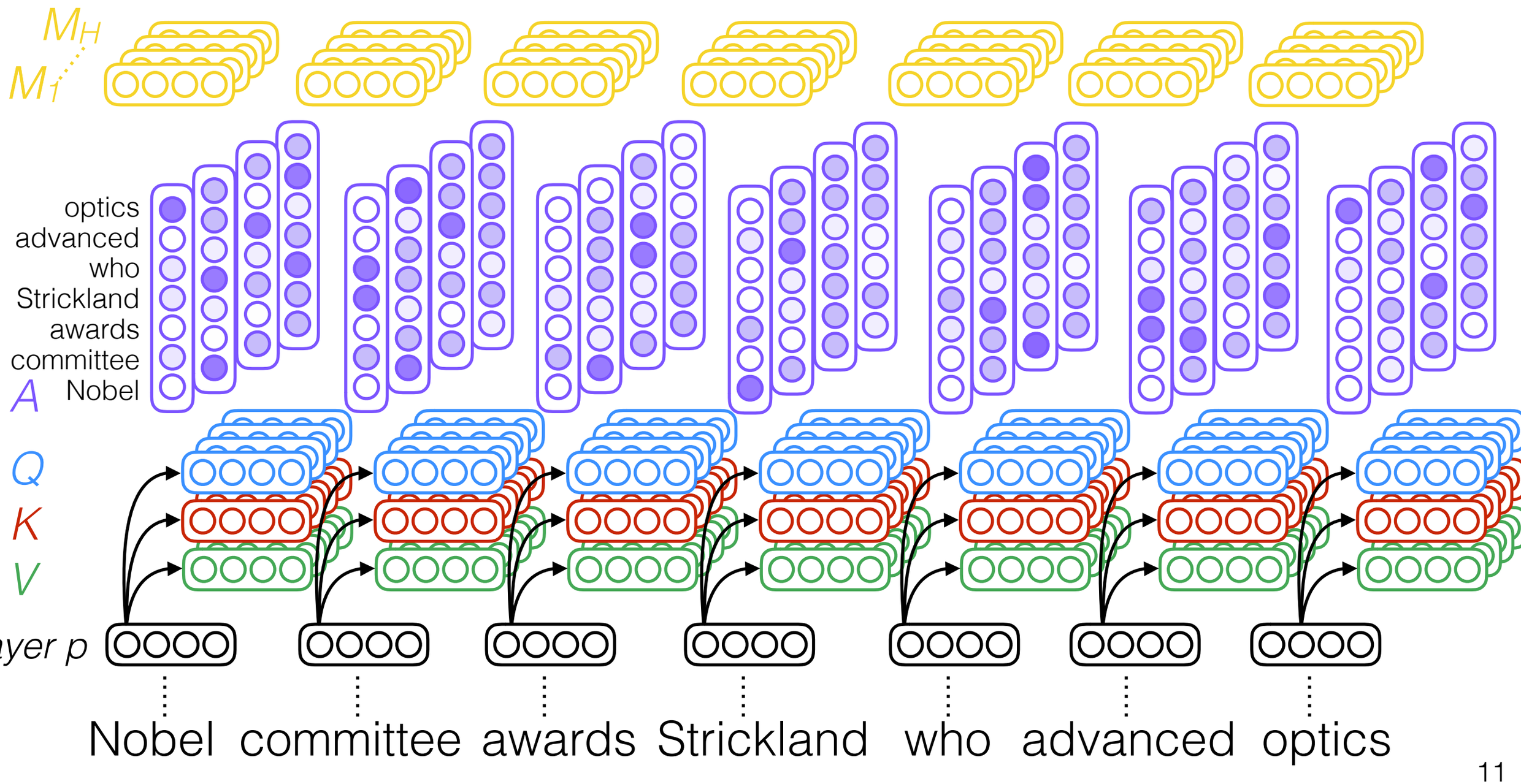
[Vaswani et al. 2017]

Self-attention



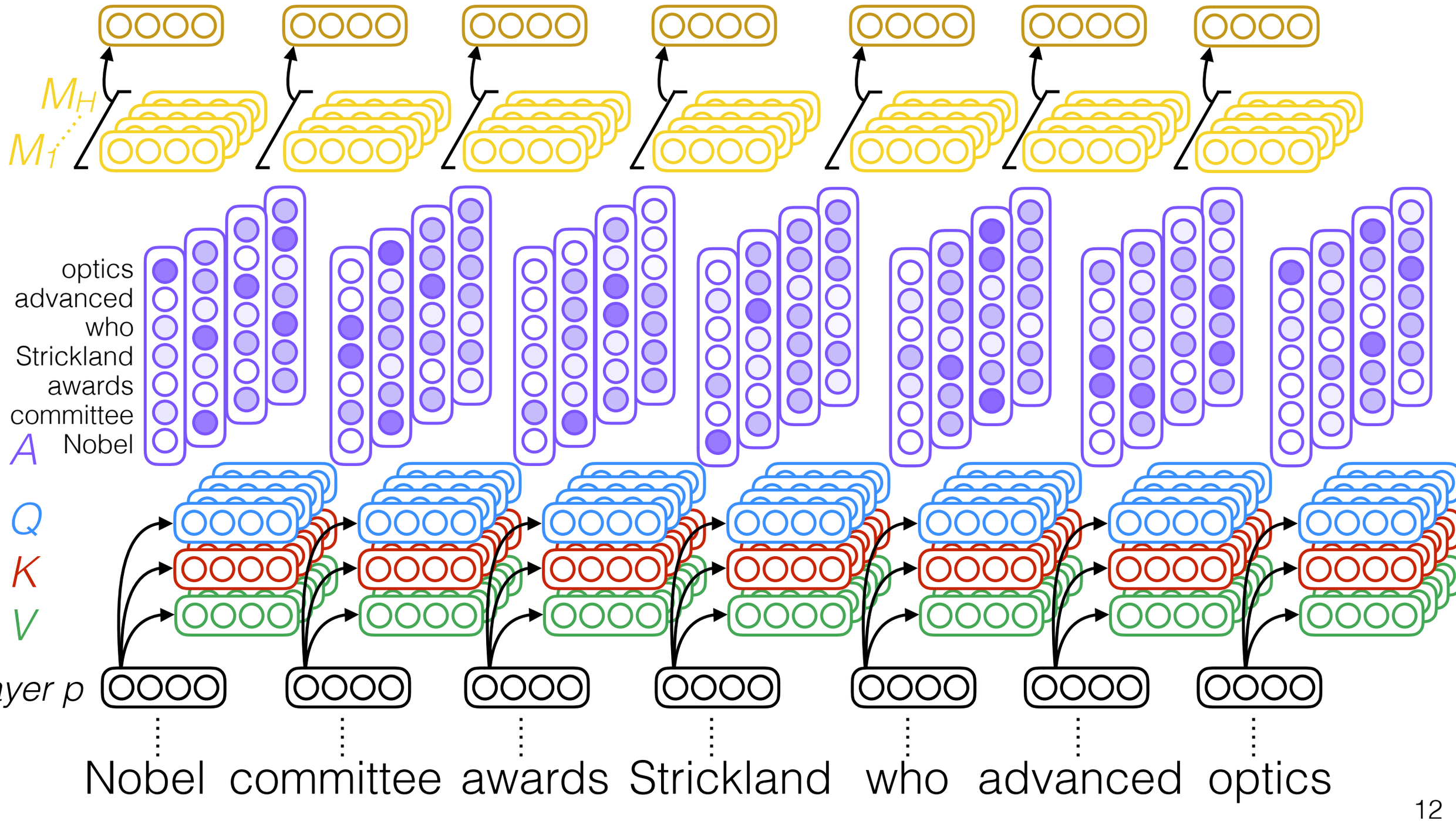
[Vaswani et al. 2017]

Multi-head self-attention



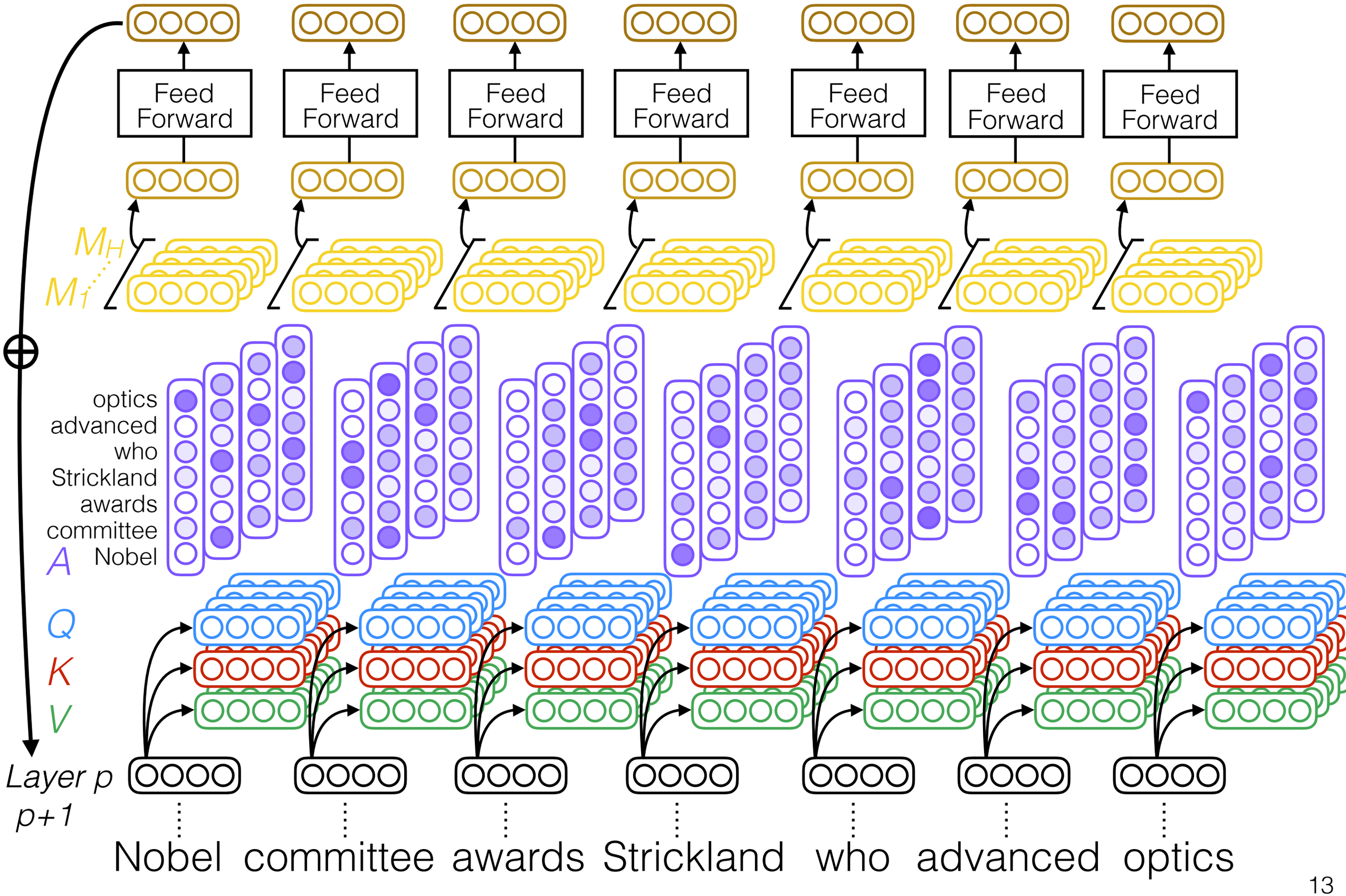
[Vaswani et al. 2017]

Multi-head self-attention



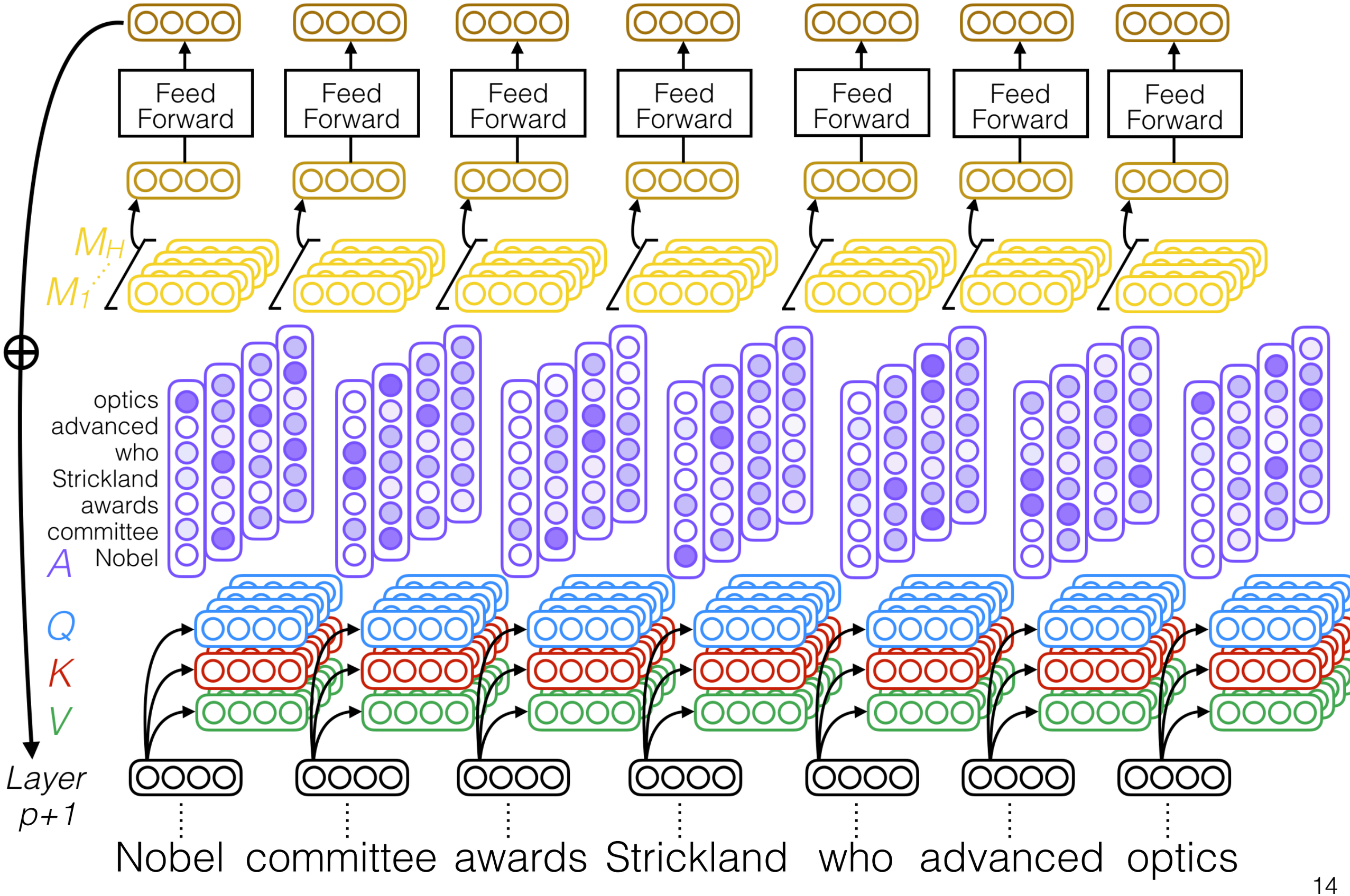
[Vaswani et al. 2017]

Multi-head self-attention



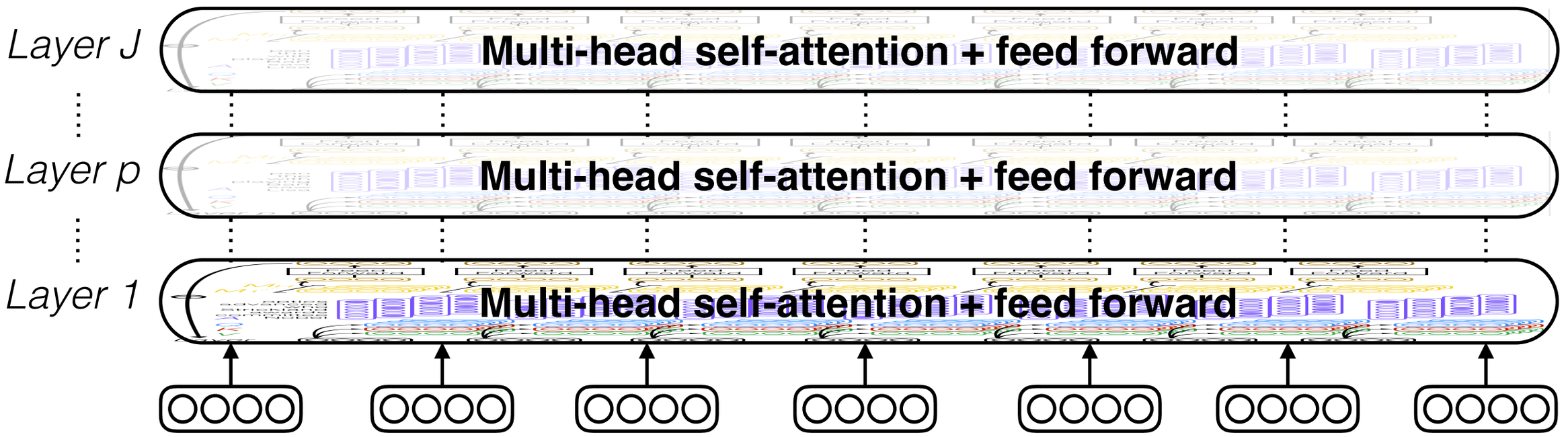
[Vaswani et al. 2017]

Multi-head self-attention

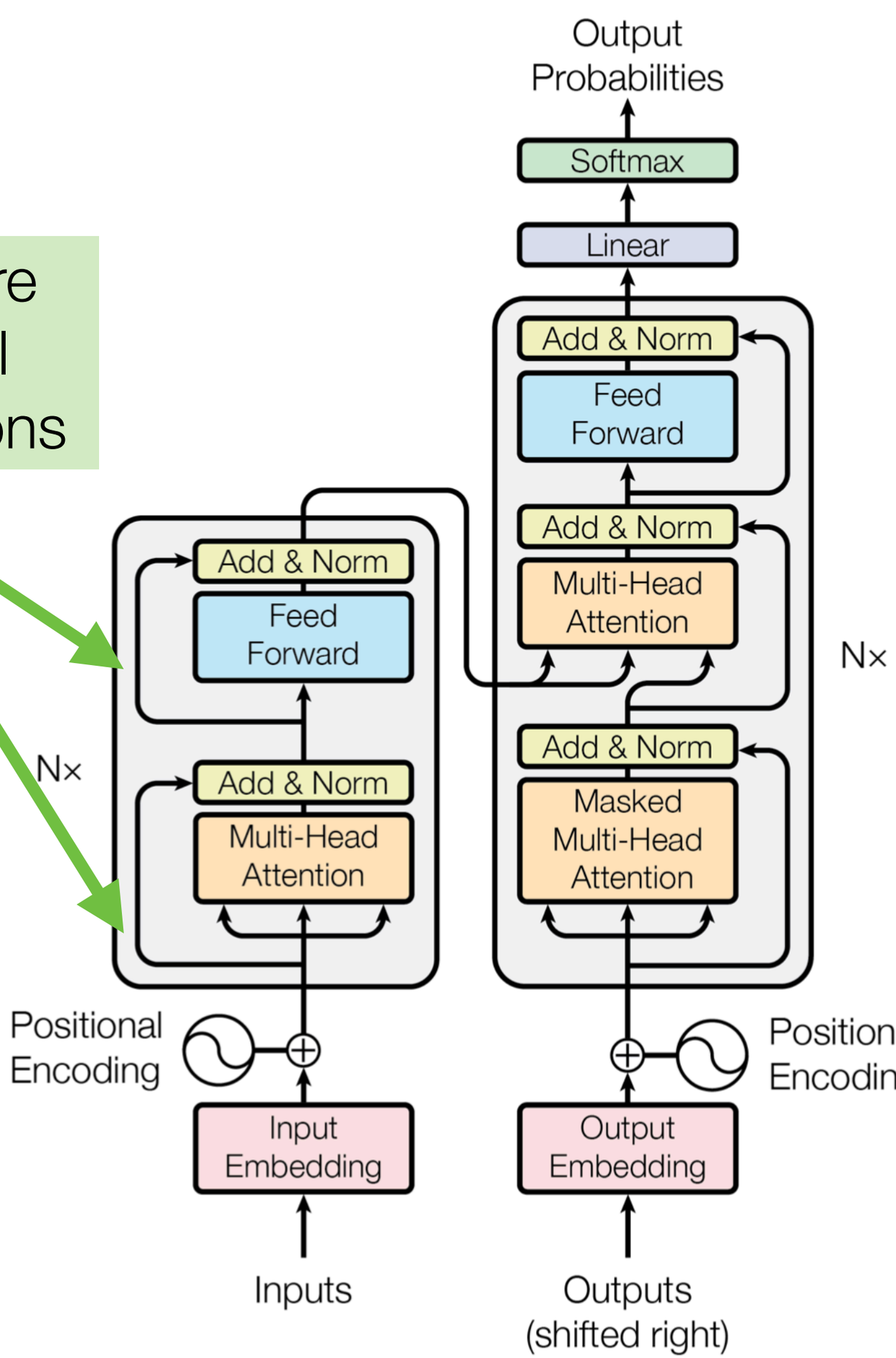


[Vaswani et al. 2017]

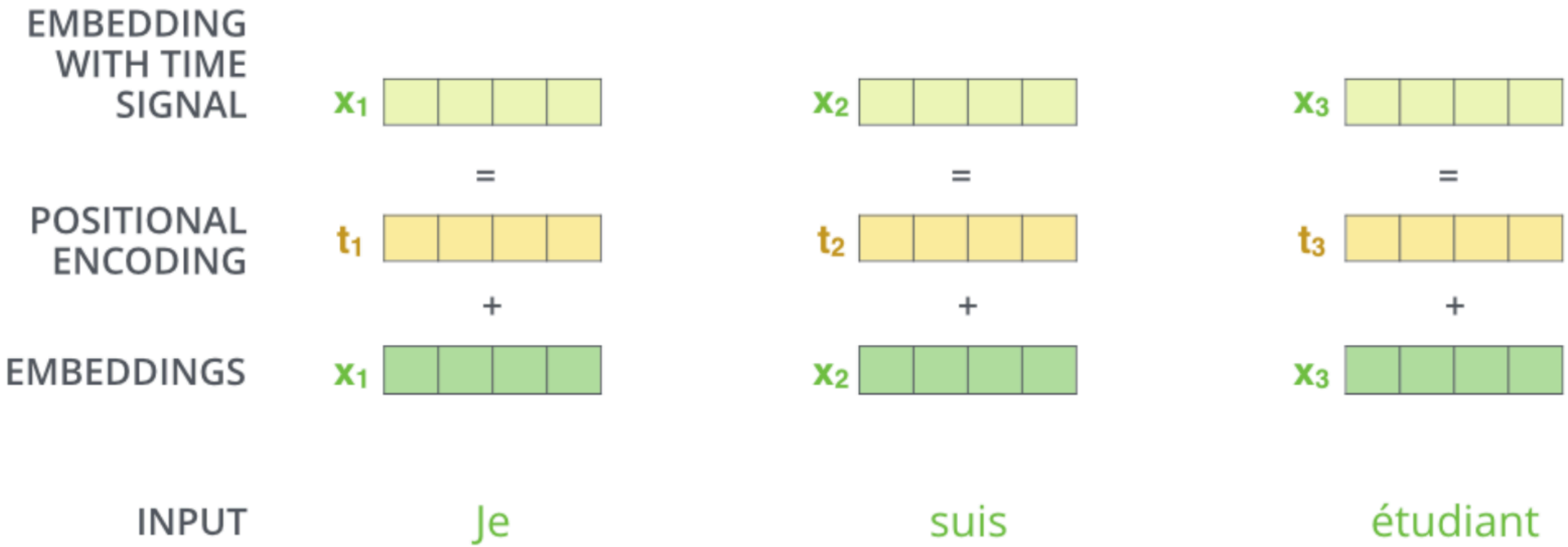
Multi-head self-attention



These are residual connections



Positional encoding



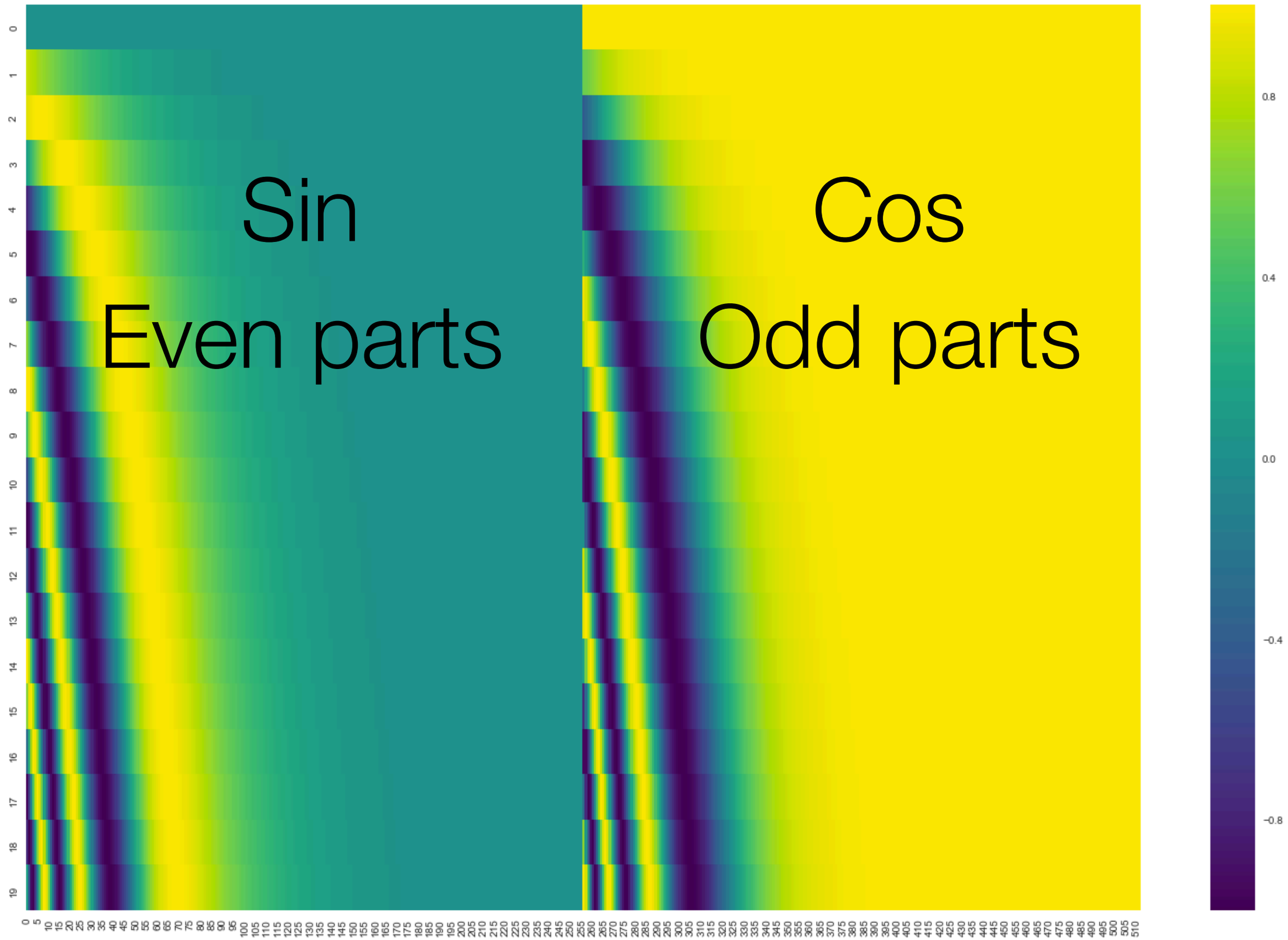
Positional encoding

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

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

Positional encoding is a 512d vector
i = a particular dimension of this vector
pos = position of the word in the text
d_model = 512

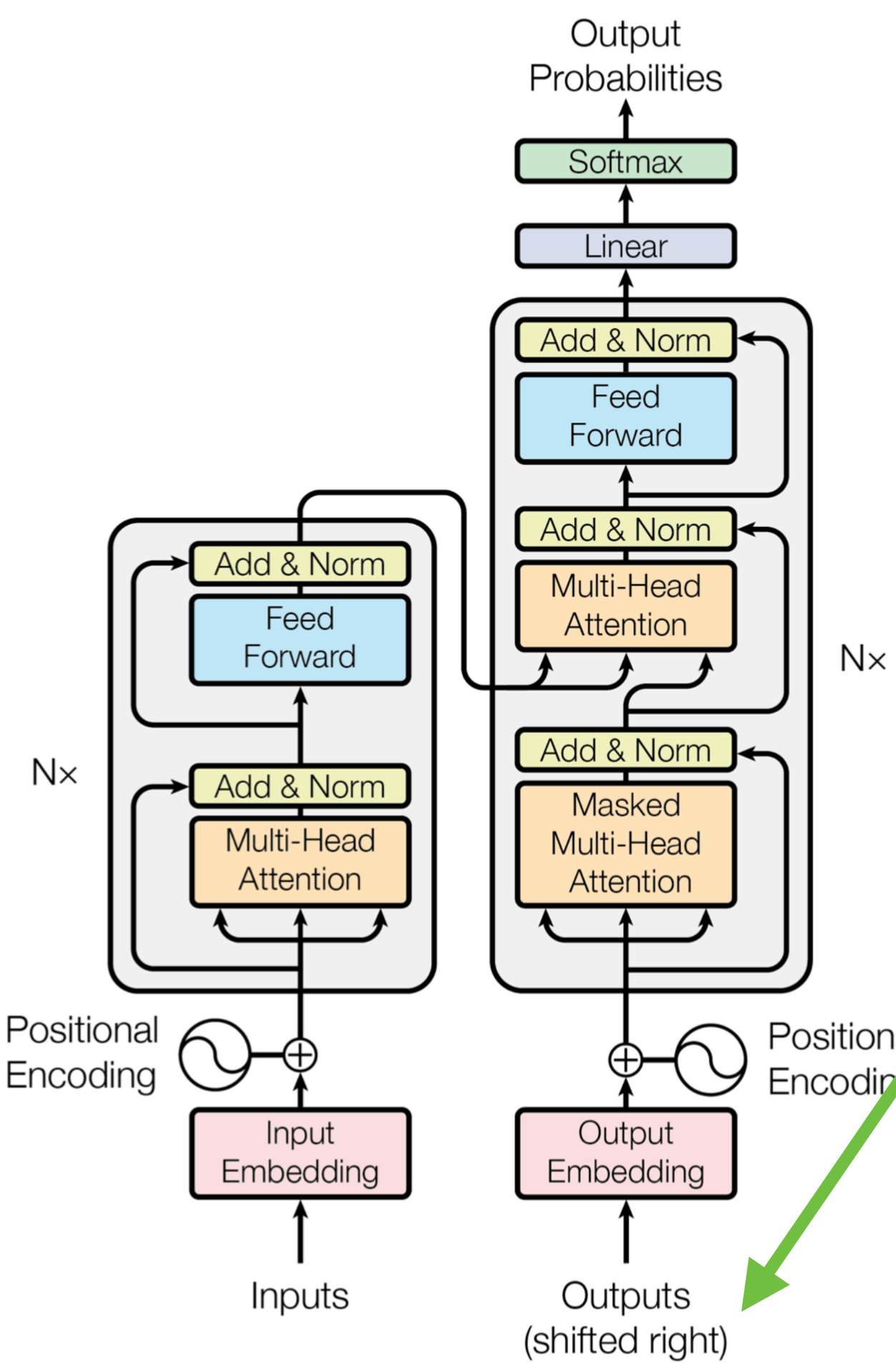
What does this look like?*



Absolute vs relative difference?

Model	Position Information	EN-DE BLEU	EN-FR BLEU
Transformer (base)	Absolute Position Representations	26.5	38.2
Transformer (base)	Relative Position Representations	**26.8**	**38.7**
Transformer (big)	Absolute Position Representations	27.9	41.2
Transformer (big)	Relative Position Representations	**29.2**	**41.5**

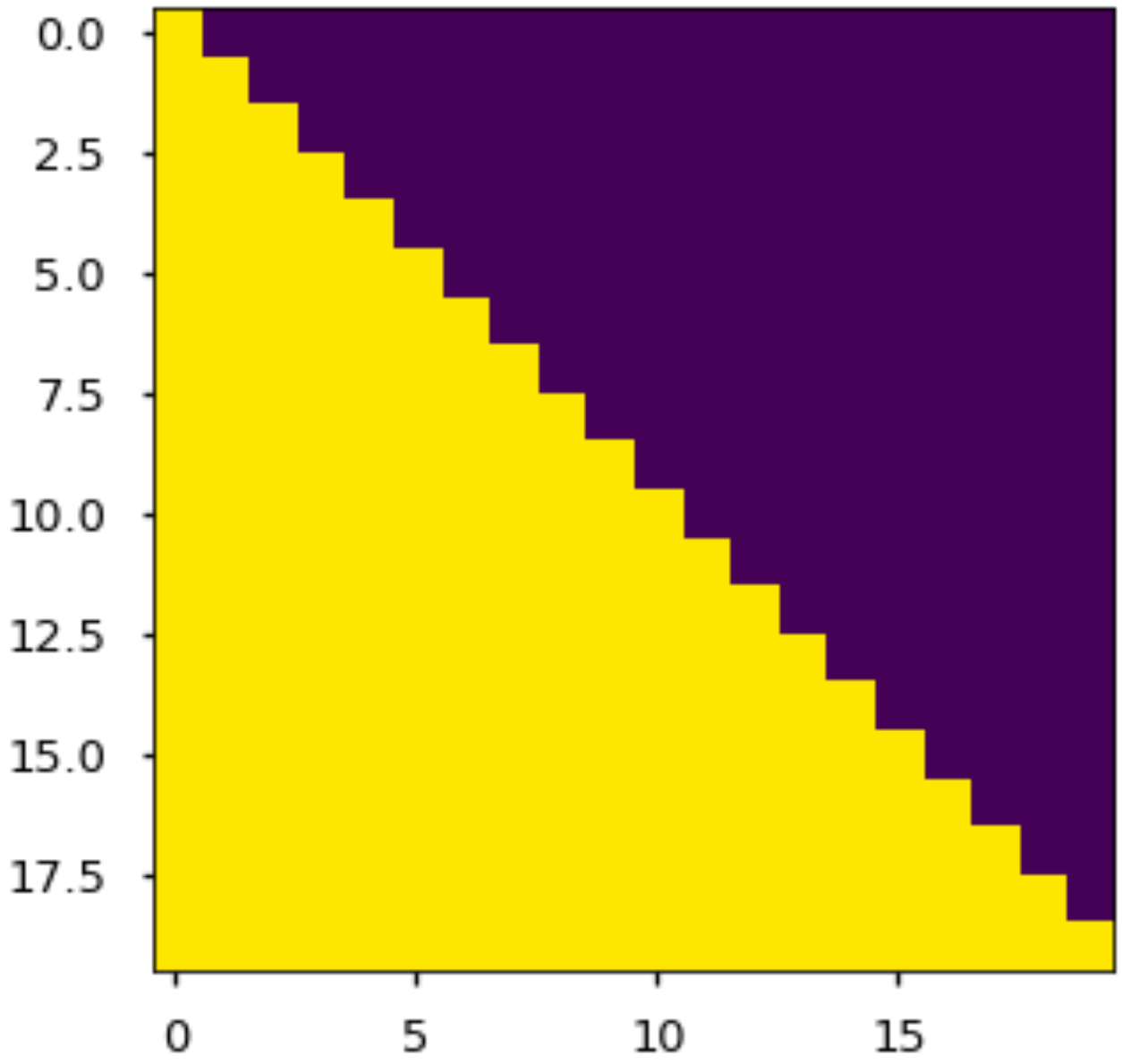
Shaw et al., NAACL 2018



What's going on here?

Last major missing piece:

- Decoder self-attention masking



Ablations

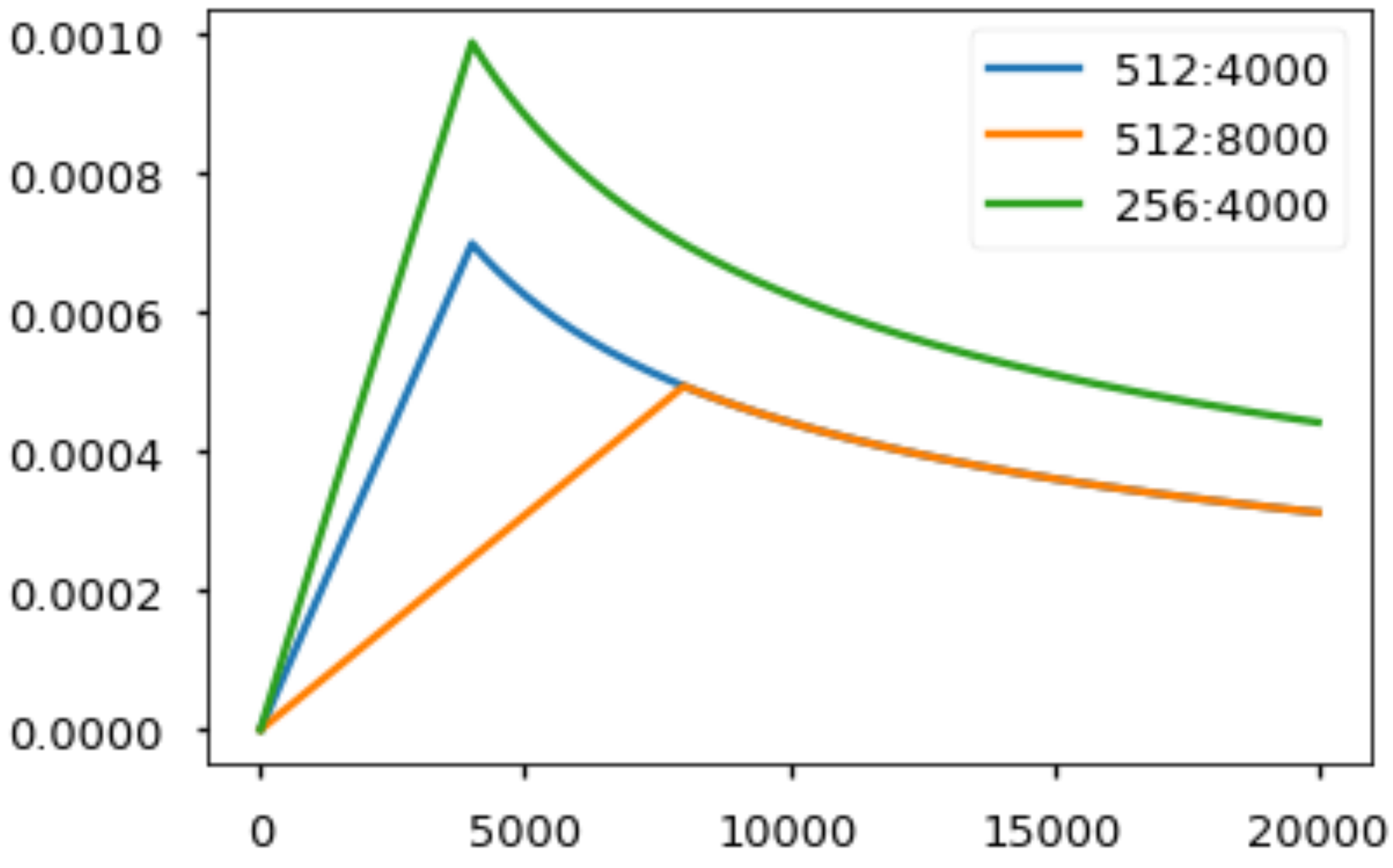
	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

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 $warmup_steps = 4000$.

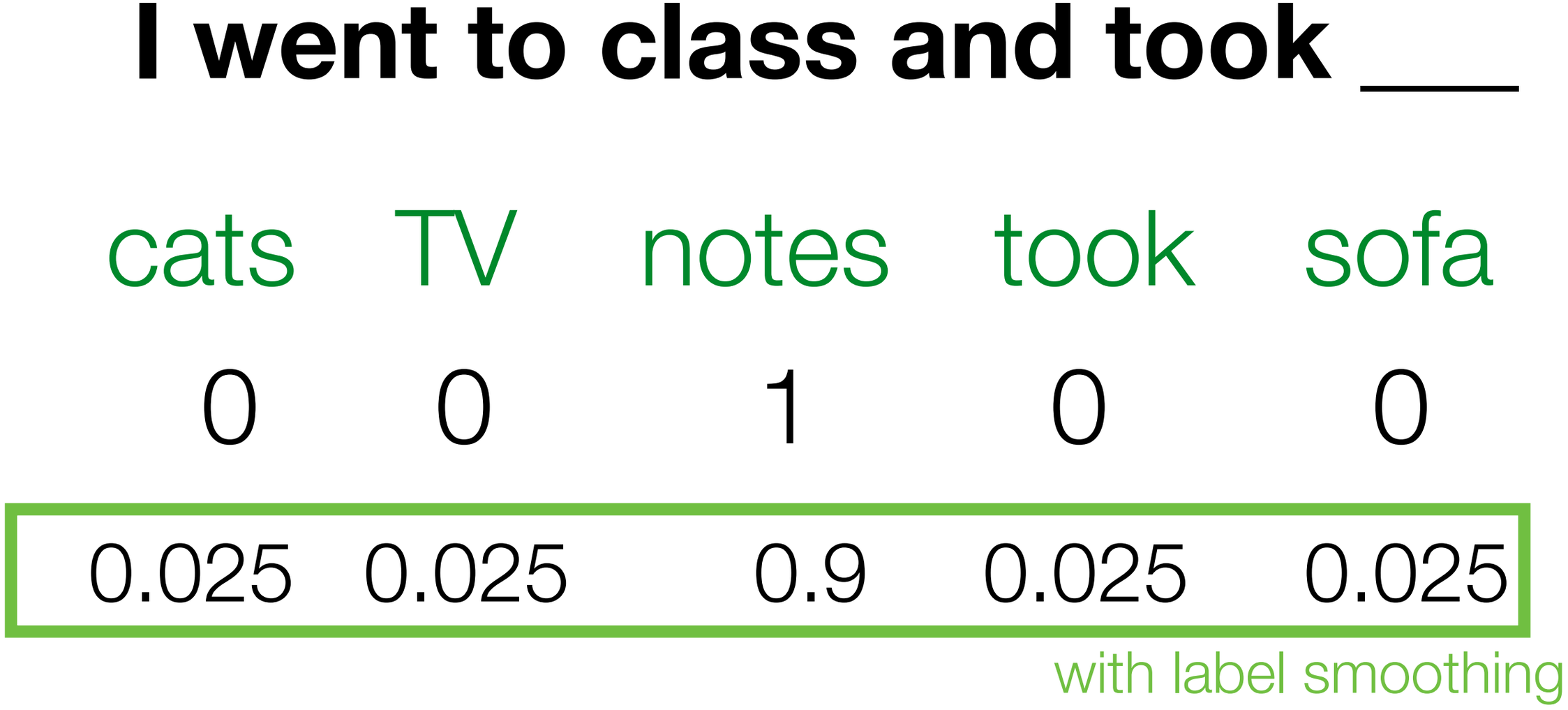
> *Note: This part is very important. Need to train with this setup of the model.*



Label Smoothing

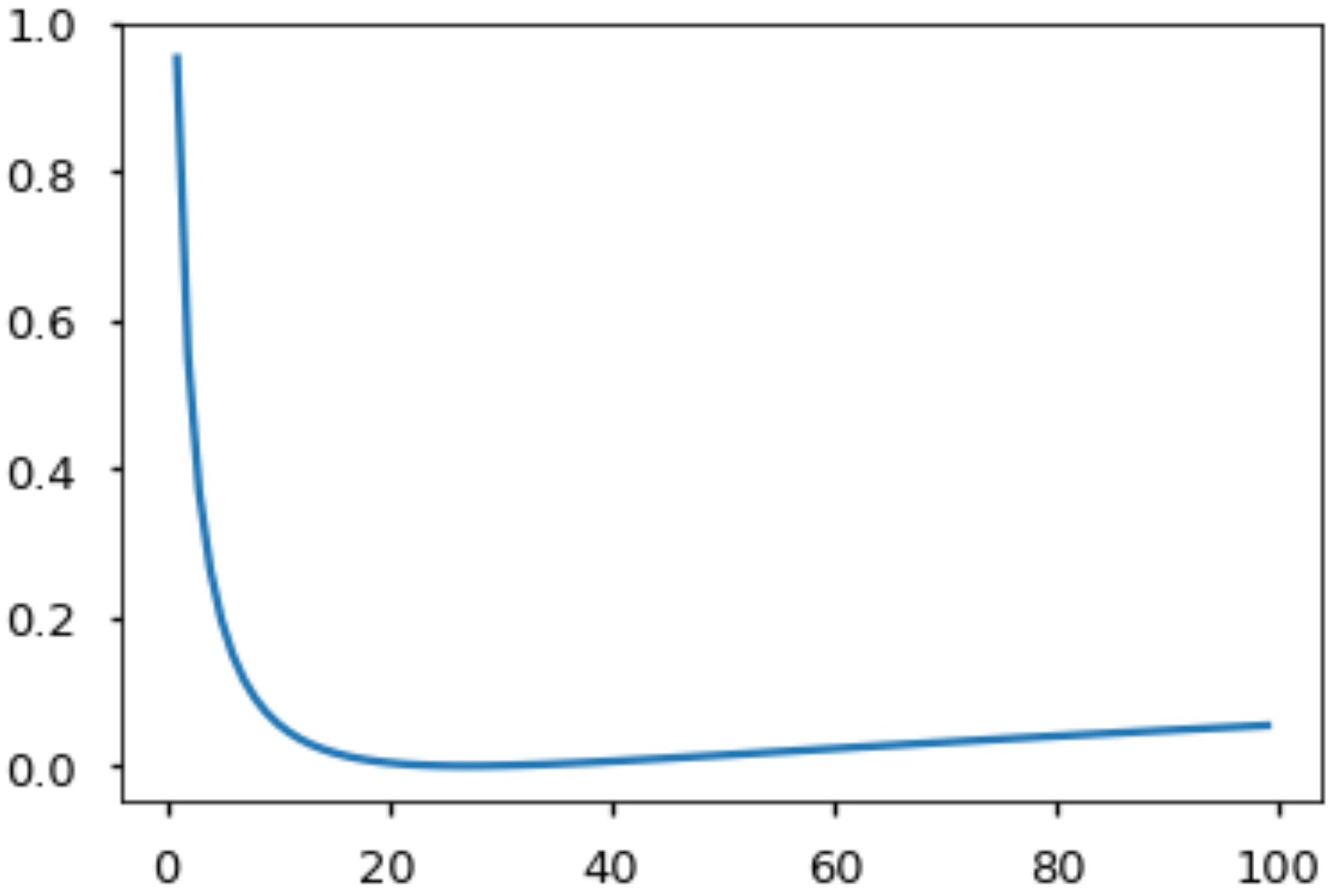
During training, we employed label smoothing of value $\epsilon_{ls} = 0.1$ (cite). This hurts perplexity, as the model learns to be more unsure, but improves accuracy and BLEU score.

> *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.*



Get penalized for overconfidence!

Loss



Target word confidence

Byte pair encoding (BPE)

- Deal with rare words / large vocabulary by instead using *subword* tokenization

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

exercise

transfer learning

Do NNs really need millions of labeled examples?

- Can we leverage *unlabeled* data to cut down on the number of labeled examples we need?

What is transfer learning?

- **In our context:** take a network trained on a task for which it is easy to generate labels, and adapt it to a different task for which it is harder.
- **In computer vision:** train a CNN on ImageNet, transfer its representations to every other CV task
- **In NLP:** train a really big language model on billions of words, transfer to every NLP task!

can we use language models to produce word embeddings?

Deep contextualized word representations. Peters et al., NAACL 2018

Word vectors are ubiquitous

Most if not all current state-of-the-art NLP systems use pre-trained word embeddings*
(as of 2018)

* With the exception of data-rich tasks like machine translation

word2vec represents each word as a **single vector**

play =[0.2, -0.1, 0.5, ...]

bank =[-0.3, 1.4, 0.7, ...]

run =[-0.5, -0.3, -0.1, ...]

Single vector per word

The new-look ***play*** area is due to be completed by early spring 2010 .

Single vector per word

Gerrymandered congressional districts favor representatives who ***play*** to the party base .

Single vector per word

The freshman then completed the three-point ***play*** for a 66-63 lead .

Nearest neighbors

play = `[0.2, -0.1, 0.5, ...]`

Nearest Neighbors

playing	plays
game	player
games	Play
played	football
players	multiplayer

Multiple senses entangled

play =[0.2, -0.1, 0.5, ...]

Nearest Neighbors

playing
game
games
played
players

VERB

plays
player
Play
football
multiplayer

Multiple senses entangled

play =`[0.2, -0.1, 0.5, ...]`

Nearest Neighbors

playing
game
games
played
players

VERB

NOUN

plays
player
Play
football
multiplayer

Multiple senses entangled

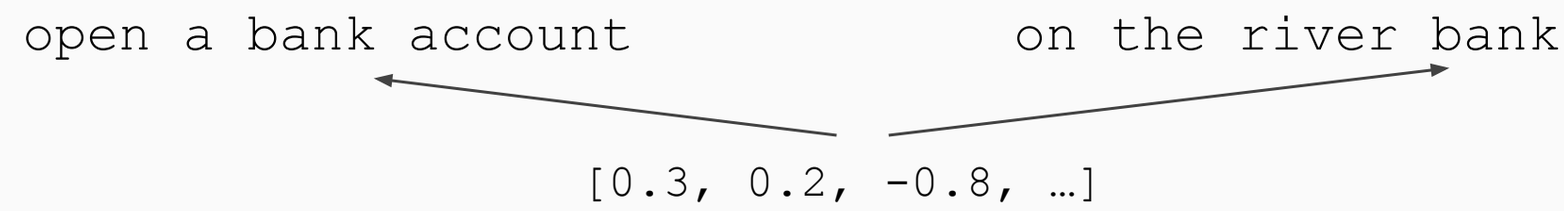
play = `[0.2, -0.1, 0.5, ...]`

Nearest Neighbors

playing	**VERB**	plays
game	**NOUN**	player
games		Play
played		football
players	**ADJ**	multiplayer

Contextual Representations

- **Problem**: Word embeddings are applied in a context free manner

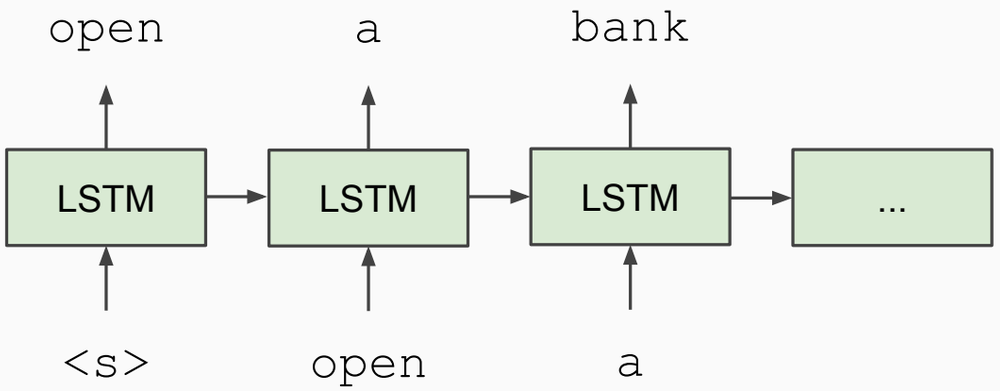


- **Solution**: Train *contextual* representations on text corpus

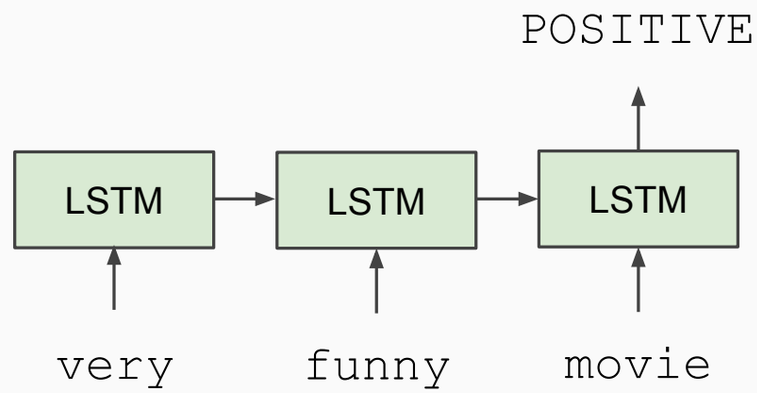
History of Contextual Representations

- *Semi-Supervised Sequence Learning*, Google, 2015

Train LSTM Language Model



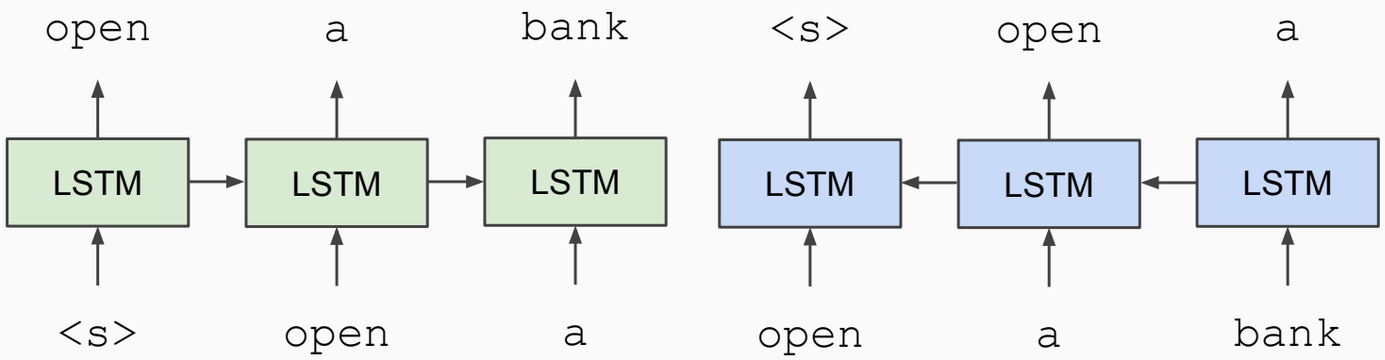
Fine-tune on Classification Task



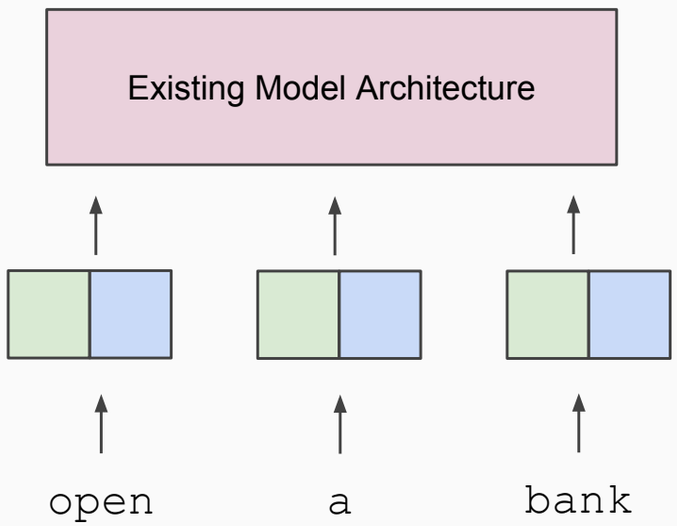
History of Contextual Representations

- *ELMo: Deep Contextual Word Embeddings*, AI2 & University of Washington, 2017

Train Separate Left-to-Right and Right-to-Left LMs



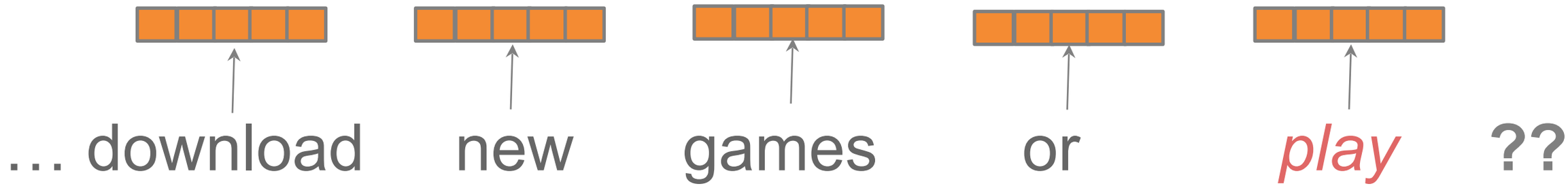
Apply as "Pre-trained Embeddings"



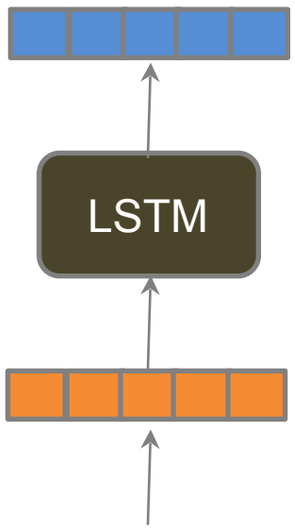
Deep bidirectional language model

… download new games or *play* ??

Deep bidirectional language model

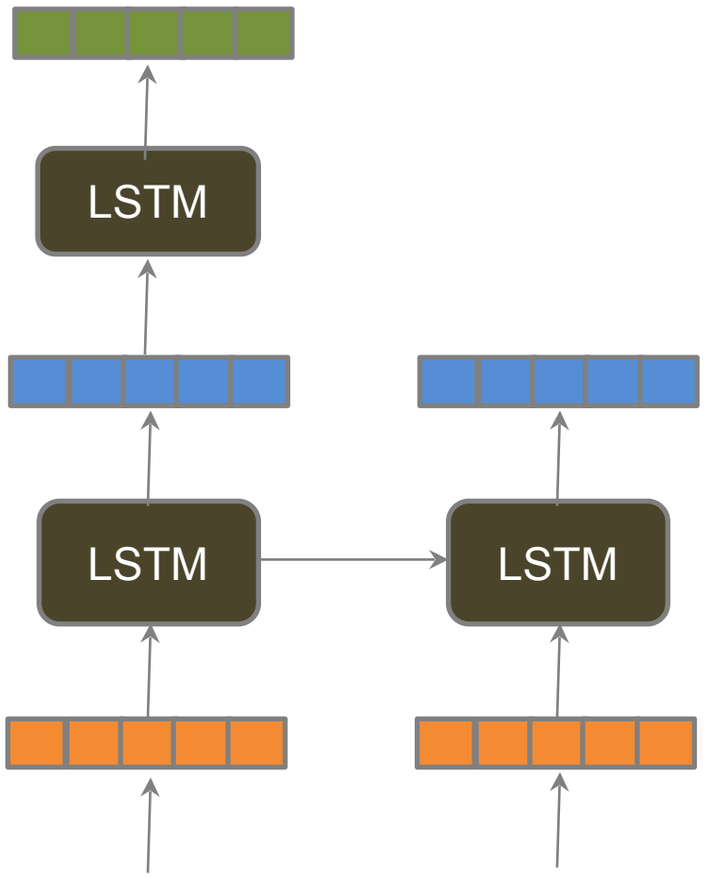


Deep bidirectional language model



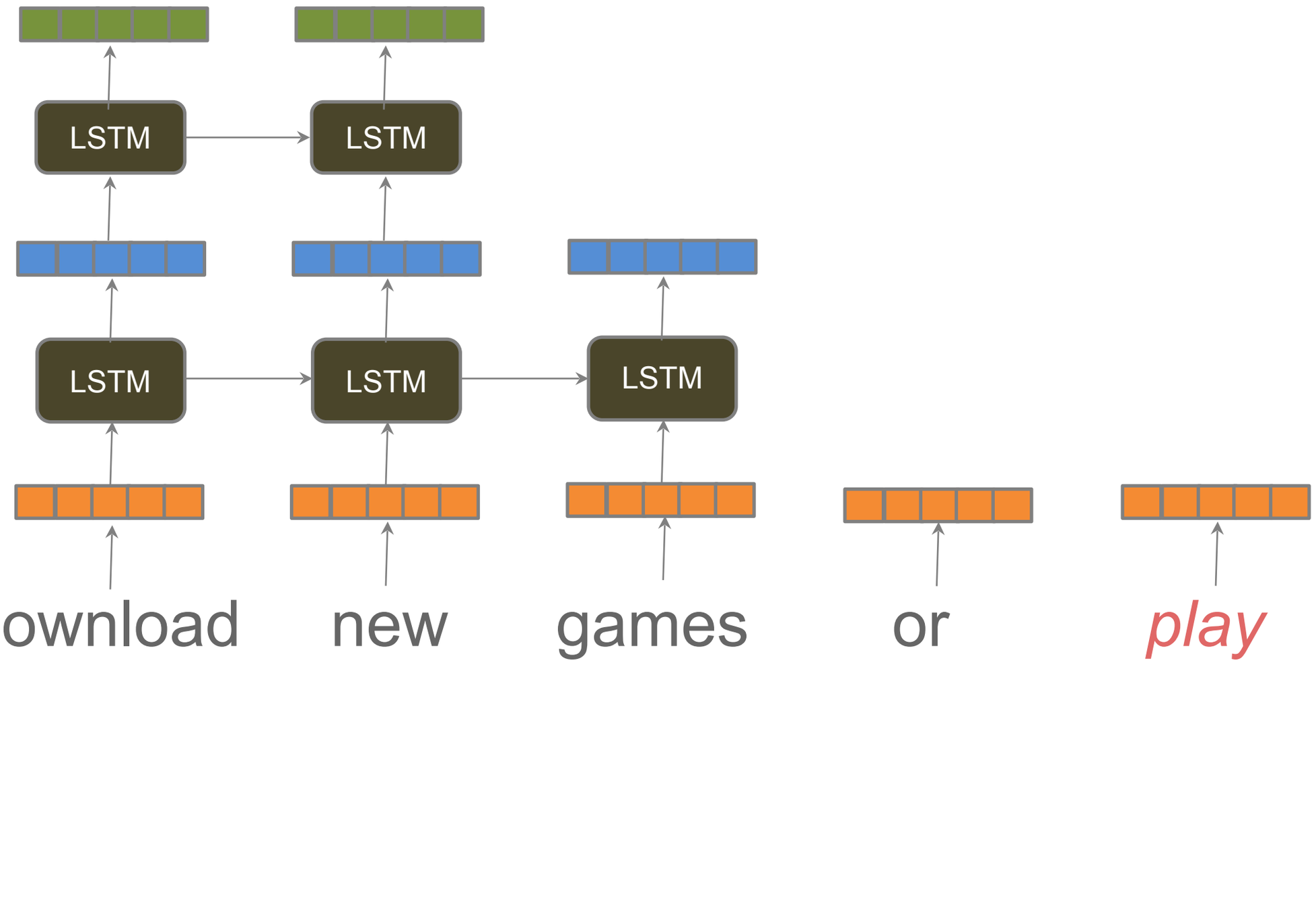
… download new games or *play* ??

Deep bidirectional language model

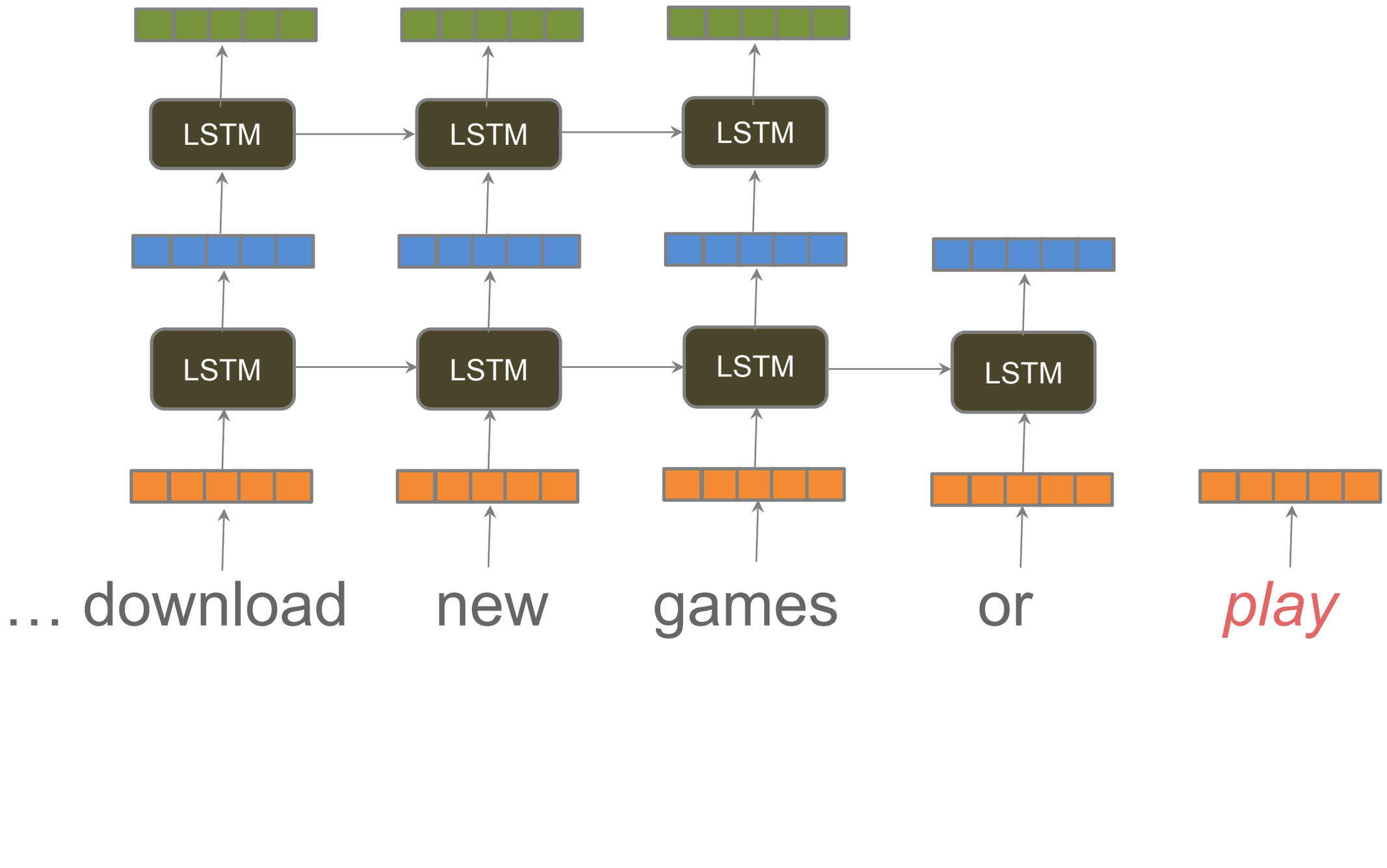


… download new games or *play* ??

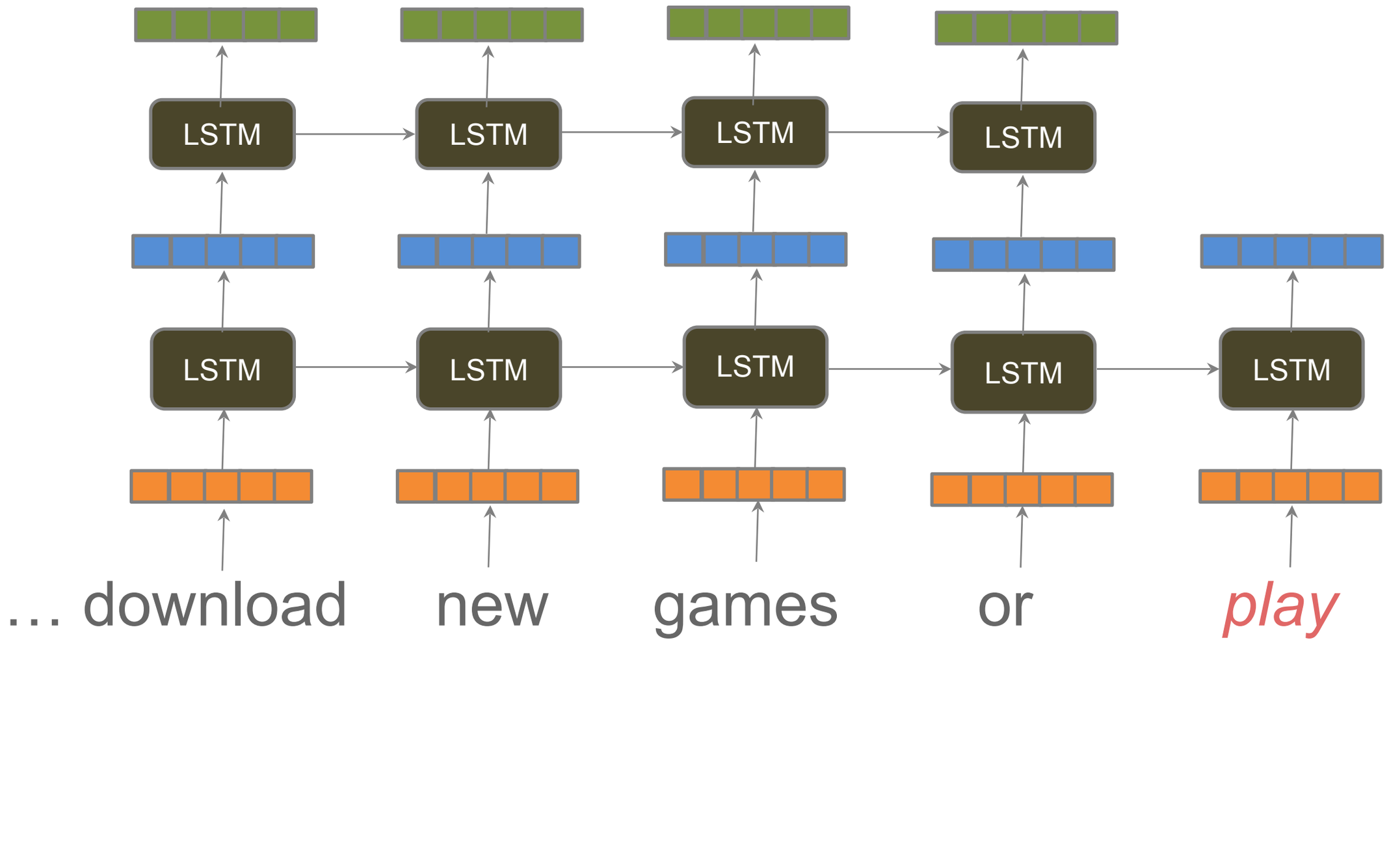
Deep bidirectional language model



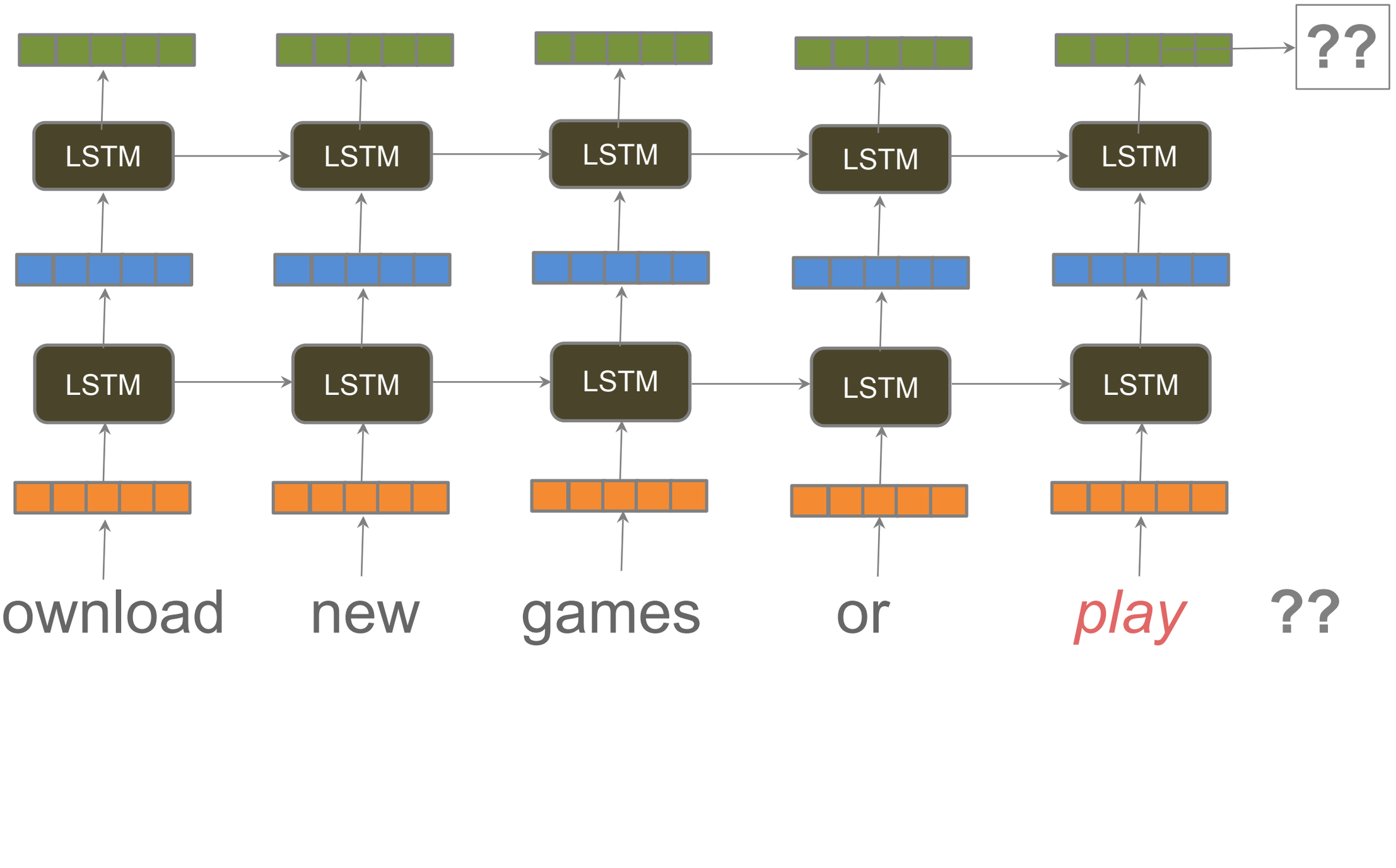
Deep bidirectional language model



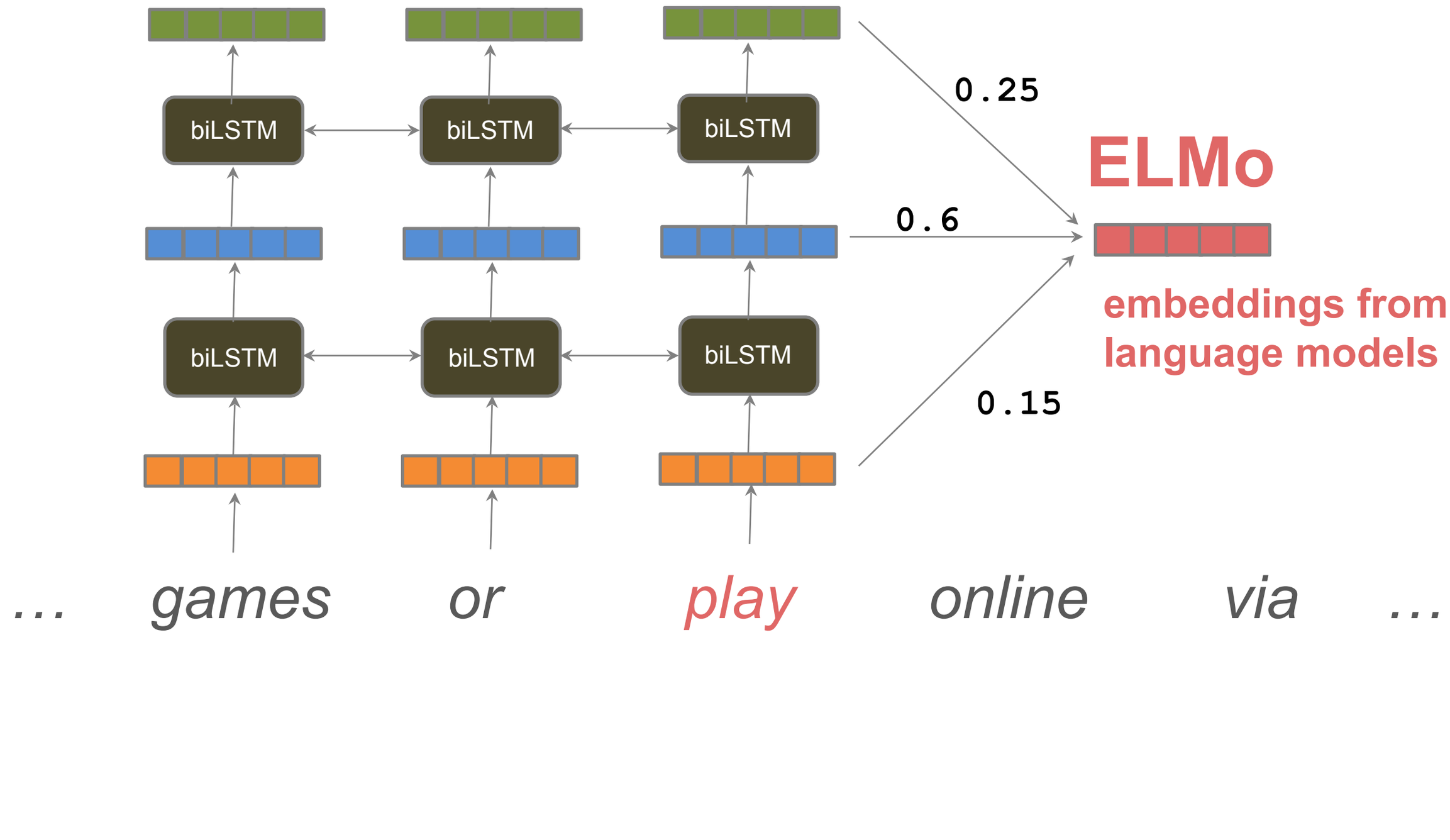
Deep bidirectional language model



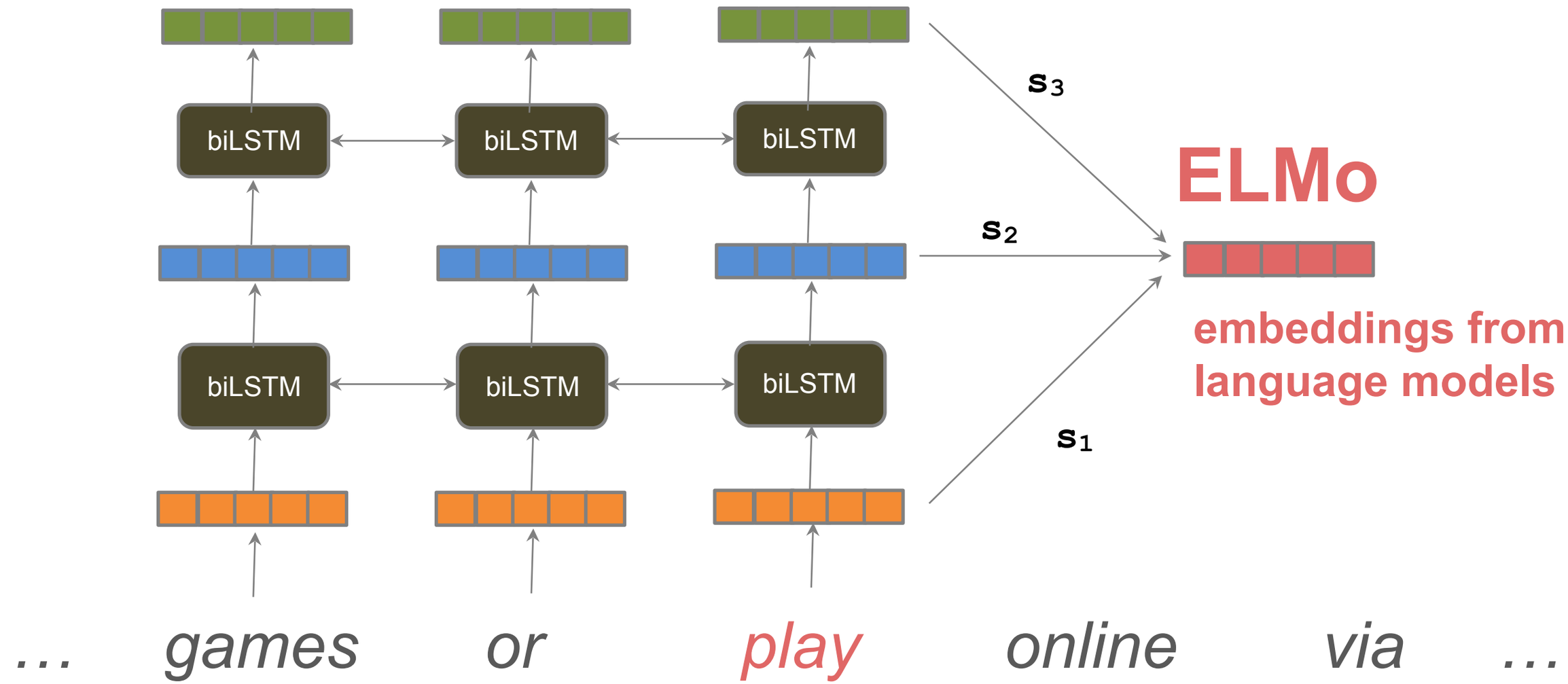
Deep bidirectional language model



Use all layers of language model



Learned task-specific combination of layers



The biLM produces $2L + 1$ intermediate representations:

$$\begin{aligned} R_k =& \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ =& \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\} \end{aligned}$$

where $\mathbf{h}_{k,0}^{LM} = \mathbf{x}_k^{LM}$ is the token layer and $\mathbf{h}_{k,j}^{LM} = [\overrightarrow{\mathbf{h}}_{k,j}^{LM}; \overleftarrow{\mathbf{h}}_{k,j}^{LM}]$, for each biLSTM layer.

ELMo: A task specific combination of these features:

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}.$$

where $\mathbf{s}^{task}$ are softmax-normalized weights and γ^{task} is a scaling parameter.

The biLM produces $2L + 1$ intermediate representations:

$$\begin{aligned} R_k =& \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ =& \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\} \end{aligned}$$

where $\mathbf{h}_{k,0}^{LM} = \mathbf{x}_k^{LM}$ is the token layer and $\mathbf{h}_{k,j}^{LM} = [\overrightarrow{\mathbf{h}}_{k,j}^{LM}; \overleftarrow{\mathbf{h}}_{k,j}^{LM}]$, for each biLSTM layer.

ELMo: A task specific combination of these features:

layer weights

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}.$$

where $\mathbf{s}^{task}$ are softmax-normalized weights and γ^{task} is a scaling parameter.

Contextual representations

ELMo representations are **contextual** – they depend on the entire sentence in which a word is used.

how many different embeddings does ELMo compute for a given word?

ELMo improves NLP tasks

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%

Large-scale recurrent neural language models learn contextual representations that capture basic elements of semantics and syntax

Adding ELMo to existing state-of-the-art models provides significant performance improvement on all NLP tasks.

TensorFlow ™

```
elmo = hub.Module("https://tfhub.dev/google/elmo/1", trainable=True)
embeddings = elmo(
    ["the cat is on the mat", "dogs are in the fog"],
    signature="default",
    as_dict=True)["elmo"]
```

AllenNLP

FROM

TO

Problem with Previous Methods

- **Problem**: Language models only use left context *or* right context, but language understanding is bidirectional.
- Why are LMs unidirectional?

Problem with Previous Methods

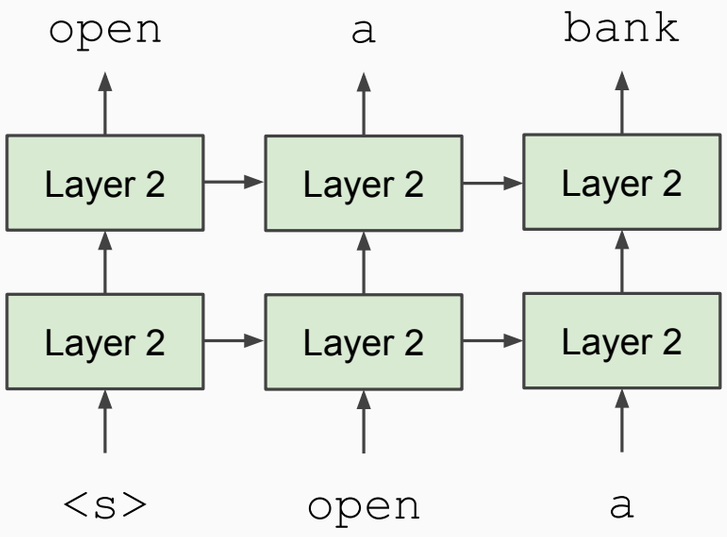
- **Problem**: Language models only use left context *or* right context, but language understanding is bidirectional.
- Why are LMs unidirectional?
- Reason 1: Directionality is needed to generate a well-formed probability distribution.
  - We don't care about this. Why not?

Problem with Previous Methods

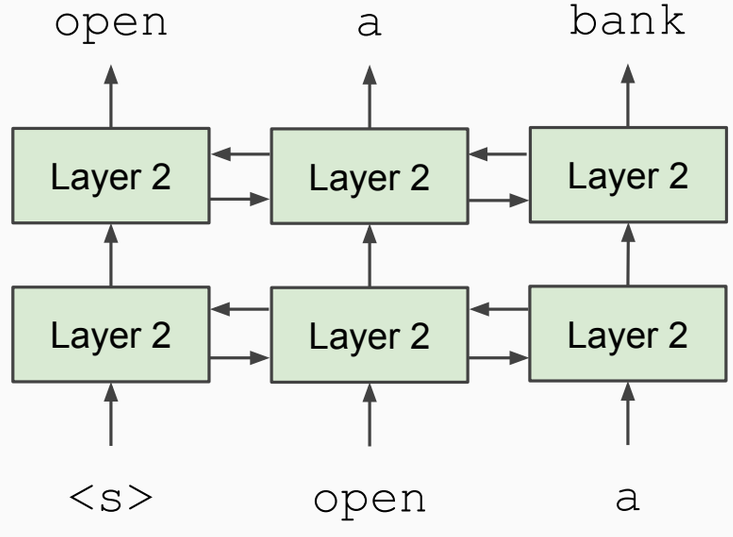
- **Problem**: Language models only use left context *or* right context, but language understanding is bidirectional.
- Why are LMs unidirectional?
- Reason 1: Directionality is needed to generate a well-formed probability distribution.
  - We don't care about this.
- Reason 2: Words can "see themselves" in a bidirectional encoder.

Unidirectional vs. Bidirectional Models

Unidirectional context
Build representation incrementally

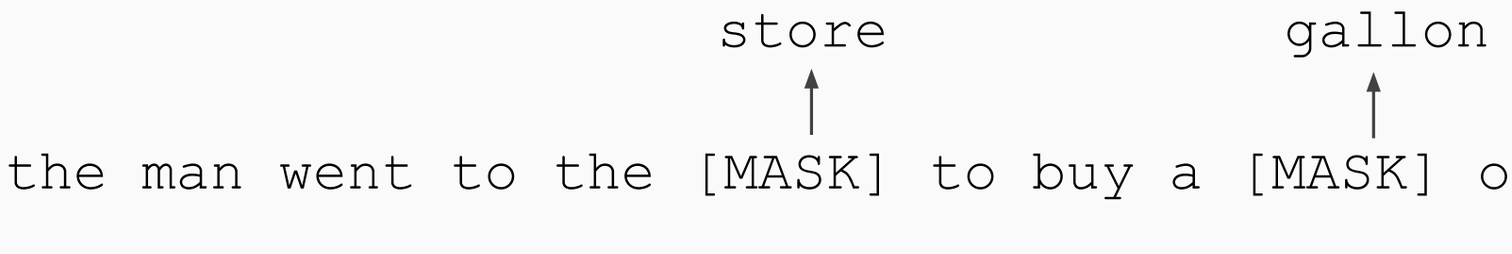


Bidirectional context
Words can “see themselves”



Masked LM

- **Solution**: Mask out *k*% of the input words, and then predict the masked words
  - We always use *k* = 15%



What are the pros and cons of increasing *k*?

Masked LM

- Problem: Mask token never seen at fine-tuning
- Solution: 15% of the words to predict, but don't replace with `[MASK]` 100% of the time. Instead:
- 80% of the time, replace with `[MASK]`

  `went to the store → went to the [MASK]`
- 10% of the time, replace random word

  `went to the store → went to the running`
- 10% of the time, keep same

  `went to the store → went to the store`

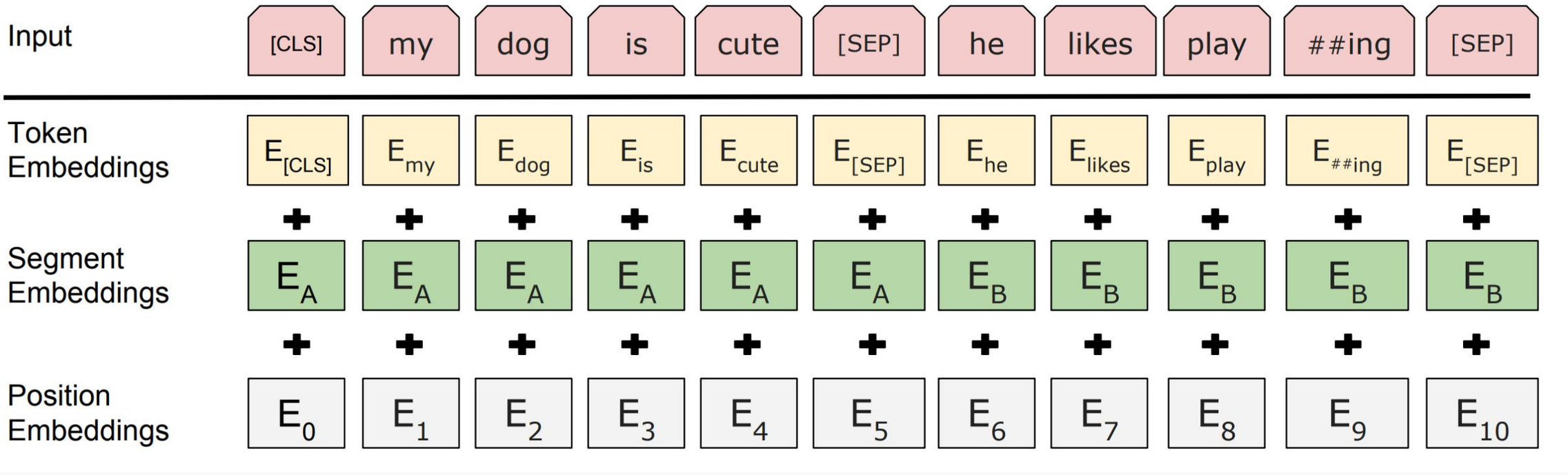
Next Sentence Prediction

- To learn *relationships* between sentences, predict whether Sentence B is actual sentence that proceeds Sentence A, or a random sentence

Sentence A = `The man went to the store.`
Sentence B = `He bought a gallon of milk.`
Label = `IsNextSentence`

Sentence A = `The man went to the store.`
Sentence B = `Penguins are flightless.`
Label = `NotNextSentence`

Input Representation

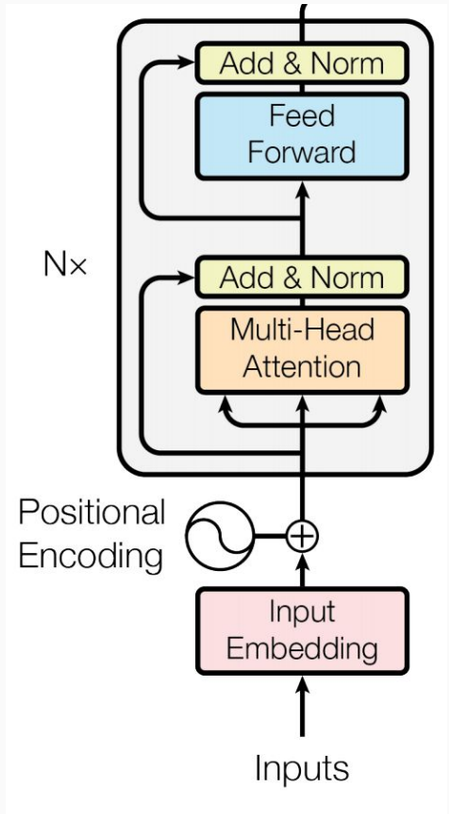


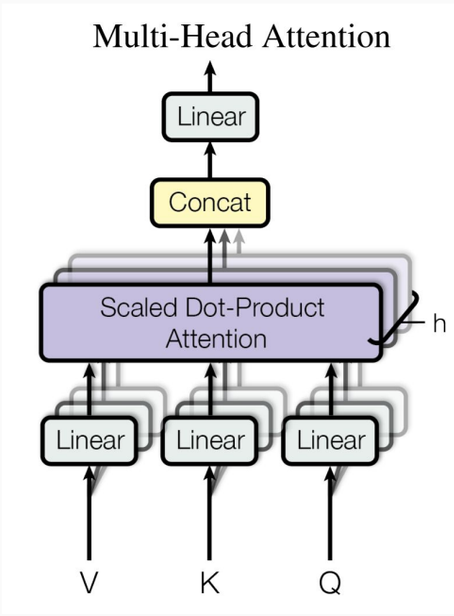
- Use 30,000 WordPiece vocabulary on input.
- Each token is sum of three embeddings
- Single sequence is much more efficient.

Model Architecture

Transformer encoder

- Multi-headed self attention
  - Models context
- Feed-forward layers
  - Computes non-linear hierarchical features
- Layer norm and residuals
  - Makes training deep networks healthy
- Positional embeddings
  - Allows model to learn relative positioning





Model Architecture

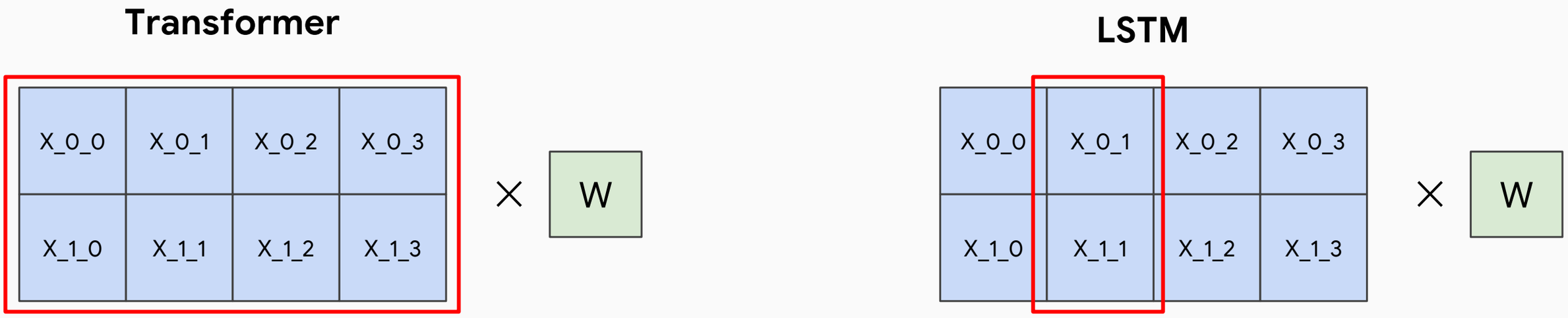
- Empirical advantages of Transformer vs. LSTM:

What are they?

Model Architecture

- Empirical advantages of Transformer vs. LSTM:

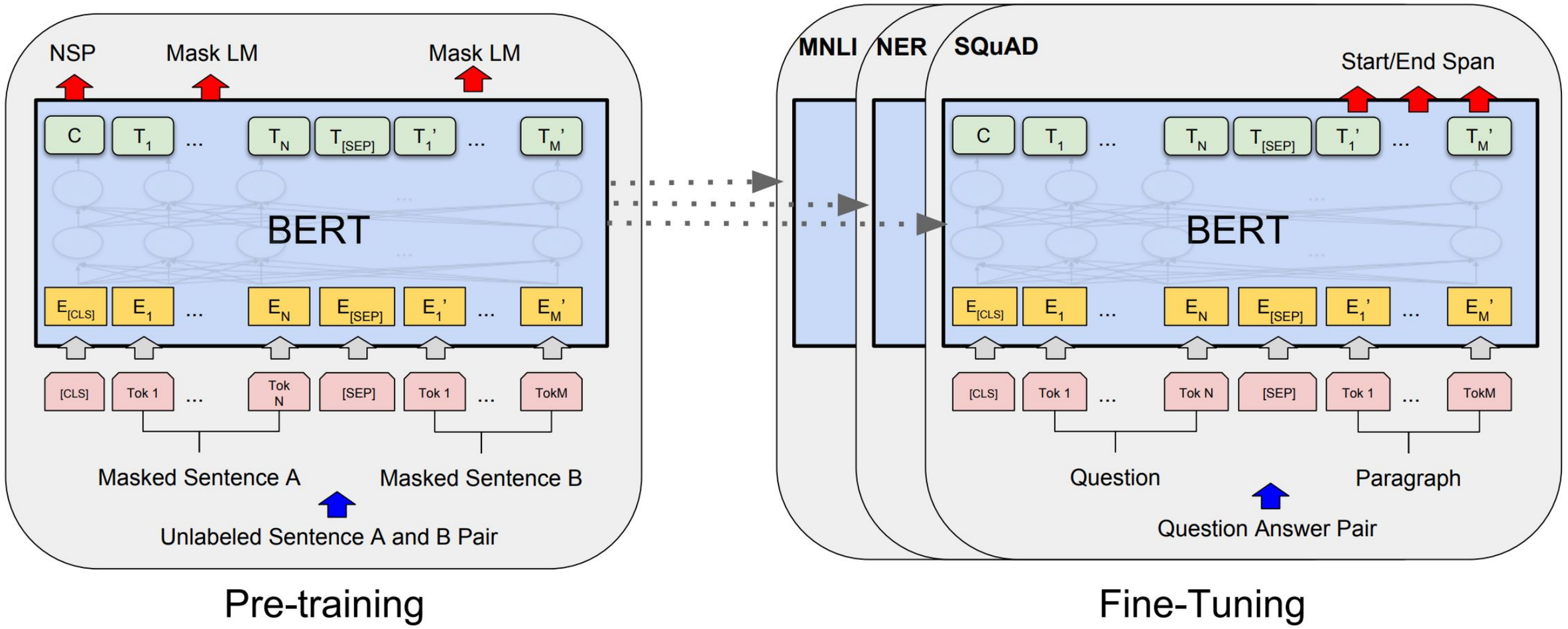
1. Self-attention == no locality bias
   - Long-distance context has “equal opportunity”
2. Single multiplication per layer == efficiency on TPU
   - Effective batch size is number of *words*, not *sequences*



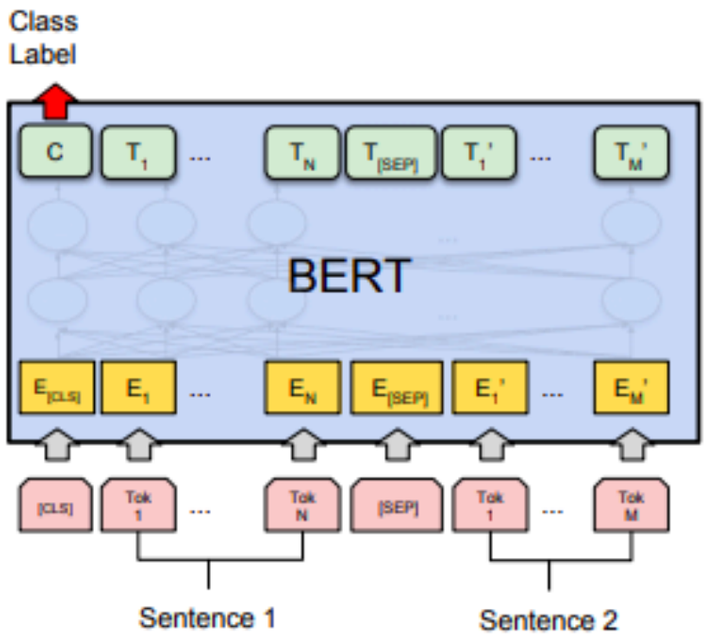
Model Details

- Data: Wikipedia (2.5B words) + BookCorpus (800M words)
- Batch Size: 131,072 words (1024 sequences * 128 length or 256 sequences * 512 length)
- Training Time: 1M steps (~40 epochs)
- Optimizer: AdamW, 1e-4 learning rate, linear decay
- `BERT-Base`: 12-layer, 768-hidden, 12-head
- `BERT-Large`: 24-layer, 1024-hidden, 16-head
- Trained on 4x4 or 8x8 TPU slice for 4 days

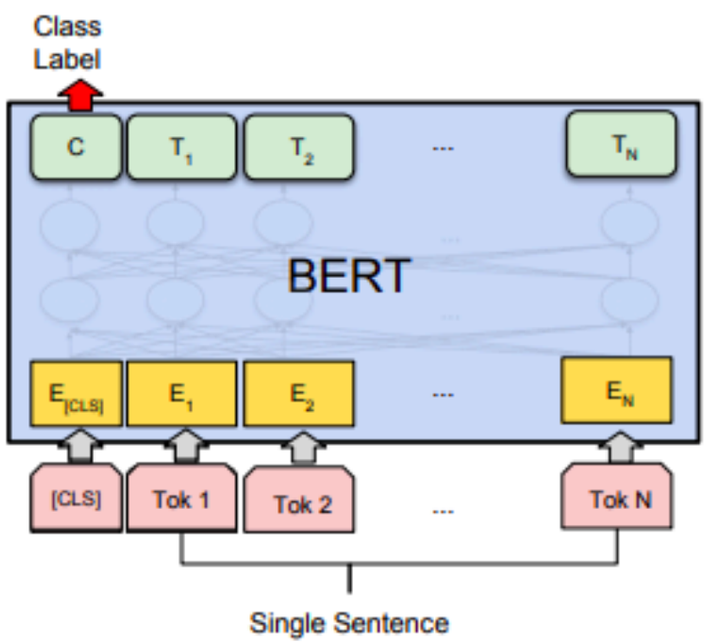
Fine-Tuning Procedure



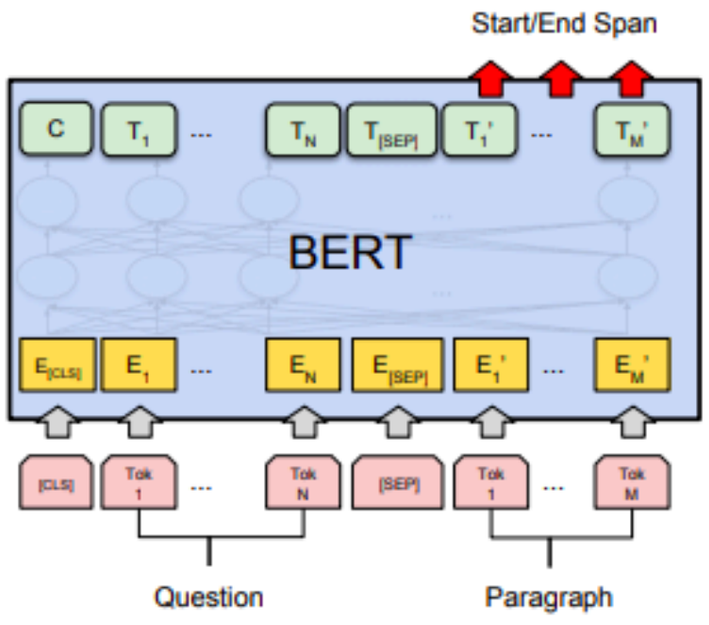
Fine-Tuning Procedure



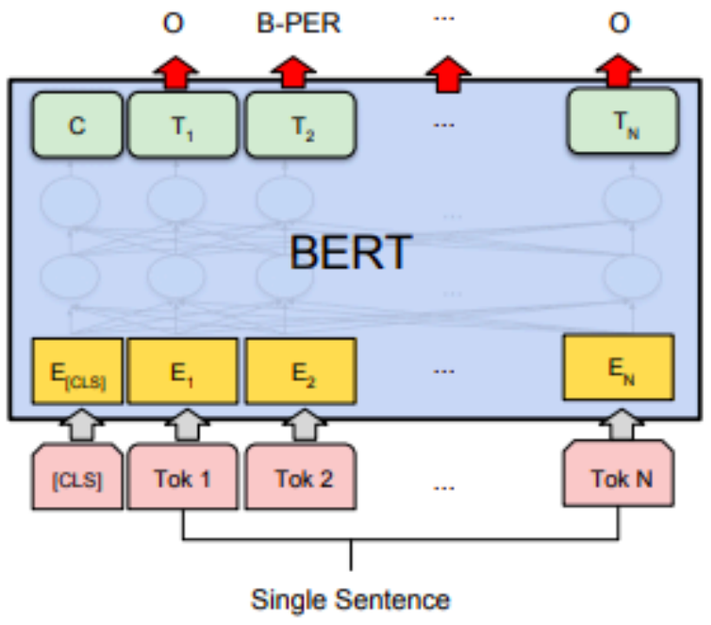
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



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



(c) Question Answering Tasks:
SQuAD v1.1



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

GLUE Results

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**

MultiNLI

Premise: Hills and mountains are especially sanctified in Jainism.
Hypothesis: Jainism hates nature.
Label: Contradiction

CoLa

Sentence: The wagon rumbled down the road.
Label: Acceptable

Sentence: The car honked down the road.
Label: Unacceptable

SWAG

A girl is going across a set of monkey bars. She

(i) jumps up across the monkey bars.

(ii) struggles onto the bars to grab her head.

(iii) gets to the end and stands on a wooden plank.

(iv) jumps up and does a back flip.

- Run each Premise + Ending through BERT.
- Produce logit for each pair on token 0 (`[CLS]`)

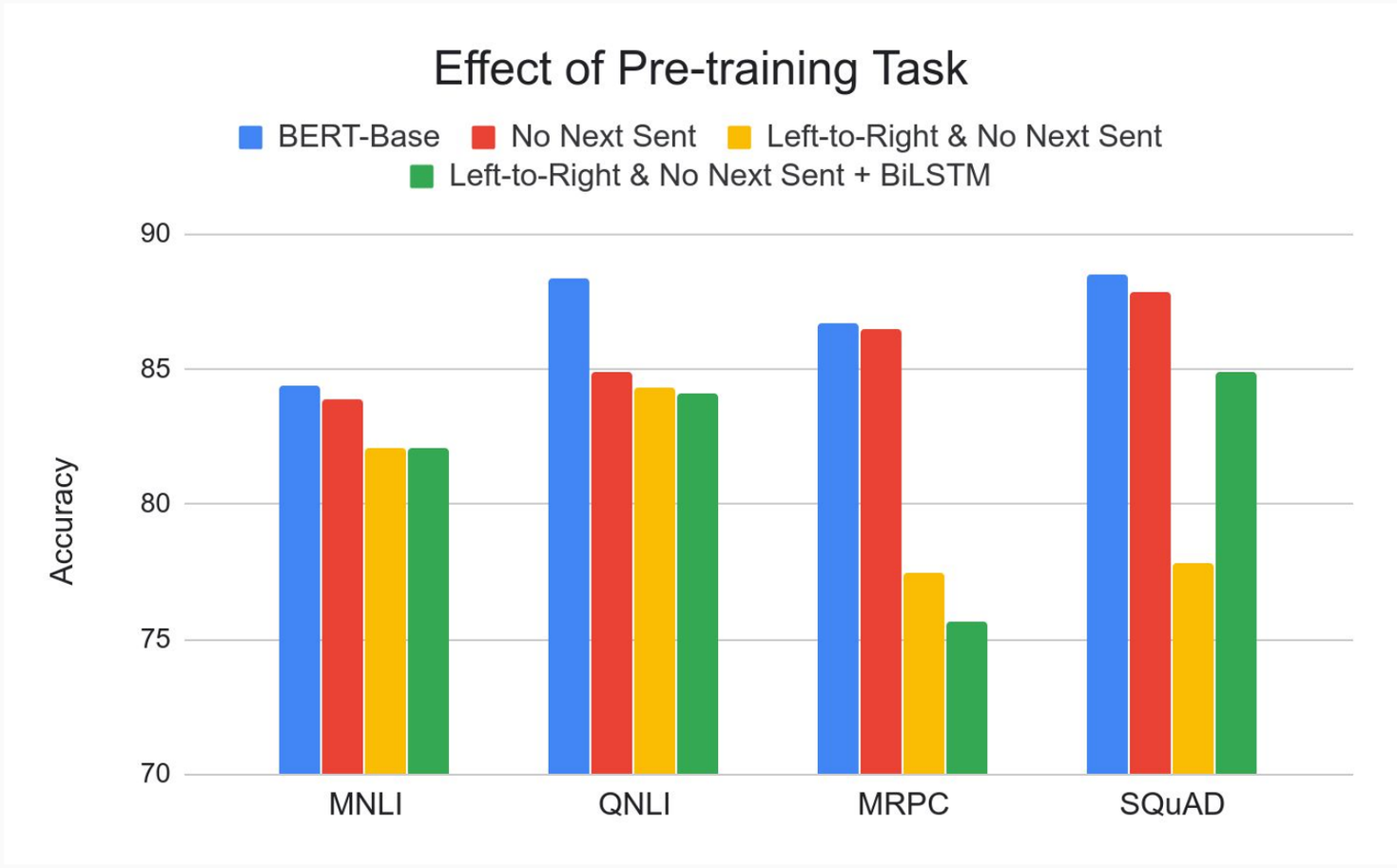
$$P_i = \frac{e^{V \cdot C_i}}{\sum_{j=1}^{4} e^{V \cdot C_j}}$$

Leaderboard

Human Performance (88.00%)
Running Best
Submissions

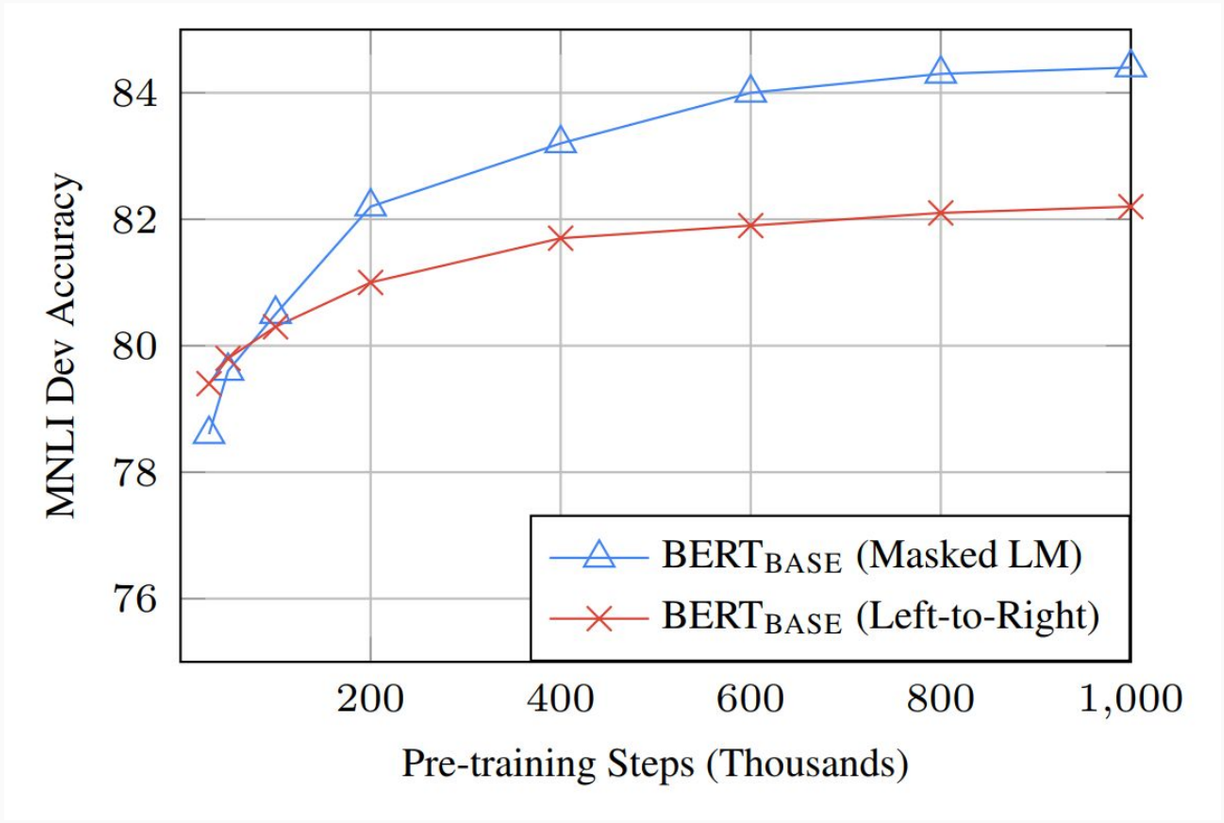
Rank	Model	Test Score
1	**BERT (Bidirectional Encoder Representations from Transfo...** *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova* 10/11/2018	**86.28%**
2	**OpenAI Transformer Language Model** *Original work by Alec Radford, Karthik Narasimhan, Tim Salimans, ...* 10/11/2018	**77.97%**
3	**ESIM with ELMo** *Zellers, Rowan and Bisk, Yonatan and Schwartz, Roy and Choi, Yejin* 08/30/2018	**59.06%**
4	**ESIM with Glove** *Zellers, Rowan and Bisk, Yonatan and Schwartz, Roy and Choi, Yejin* 08/29/2018	**52.45%**

Effect of Pre-training Task



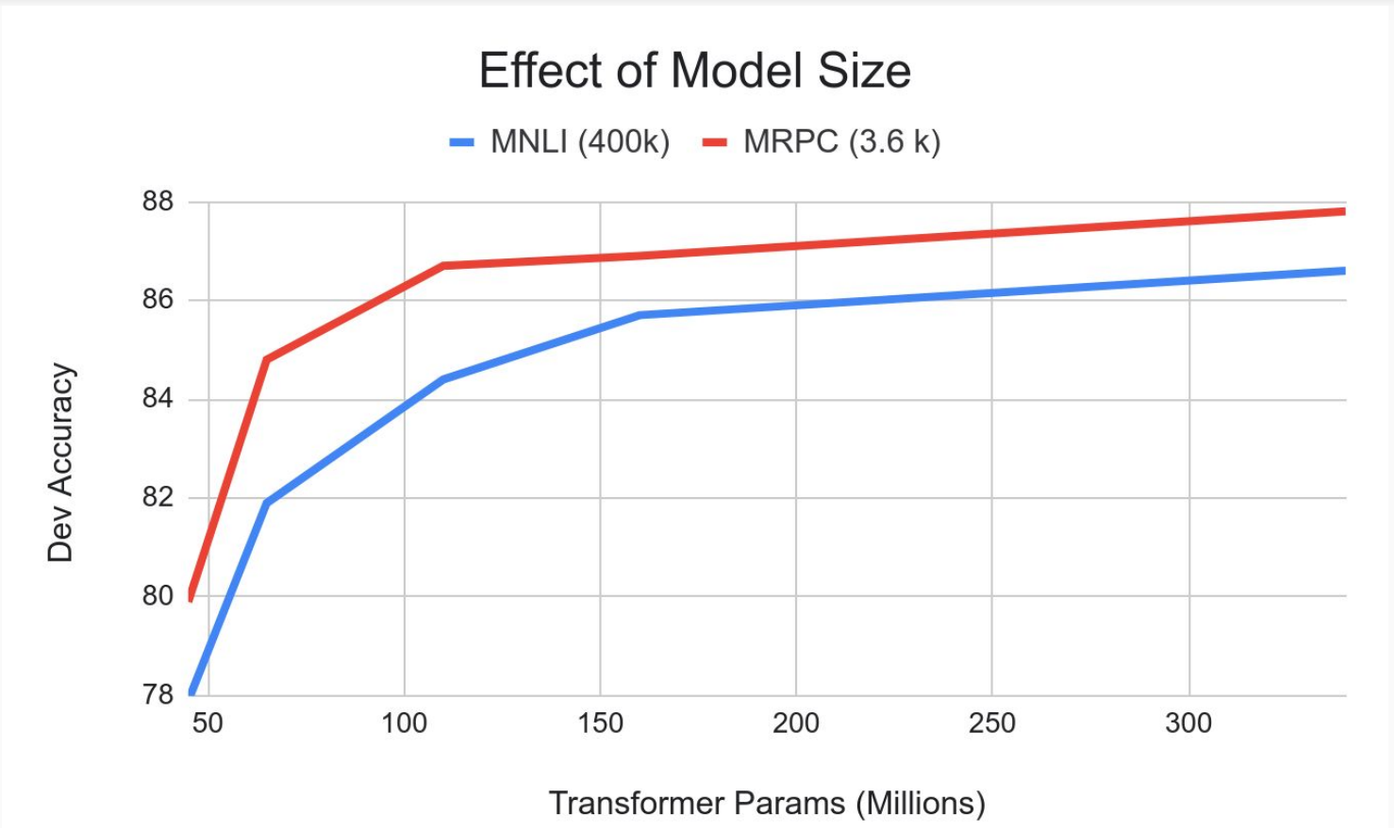
- Masked LM (compared to left-to-right LM) is very important on some tasks, Next Sentence Prediction is important on other tasks.
- Left-to-right model does very poorly on word-level task (SQuAD), although this is mitigated by BiLSTM

Effect of Directionality and Training Time



- Masked LM takes slightly longer to converge because we only predict 15% instead of 100%
- But absolute results are much better almost immediately

Effect of Model Size



- Big models help *a lot*
- Going from 110M -> 340M params helps even on datasets with 3,600 labeled examples
- Improvements have *not* asymptoted

Effect of Masking Strategy

Masking Rates			Dev Set Results		
			MNLI	NER	
MASK	SAME	RND	Fine-tune	Fine-tune	Feature-based
80%	10%	10%	84.2	95.4	94.9
100%	0%	0%	84.3	94.9	94.0
80%	0%	20%	84.1	95.2	94.6
80%	20%	0%	84.4	95.2	94.7
0%	20%	80%	83.7	94.8	94.6
0%	0%	100%	83.6	94.9	94.6

- Masking 100% of the time hurts on feature-based approach

- Using random word 100% of time hurts slightly

Multilingual BERT

- Trained single model on 104 languages from Wikipedia. Shared 110k WordPiece vocabulary.

System	English	Chinese	Spanish
XNLI Baseline - Translate Train	73.7	67.0	68.8
XNLI Baseline - Translate Test	73.7	68.4	70.7
BERT - Translate Train	81.9	76.6	77.8
BERT - Translate Test	81.9	70.1	74.9
BERT - Zero Shot	81.9	63.8	74.3

- XNLI is MultiNLI translated into multiple languages.
- Always evaluate on human-translated Test.
- Translate Train: MT English Train into Foreign, then fine-tune.
- Translate Test: MT Foreign Test into English, use English model.
- Zero Shot: Use Foreign test on English model.

Common Questions

- Is *deep* bidirectionality really necessary? What about ELMo-style shallow bidirectionality on bigger model?
- Advantage: Slightly faster training time
- Disadvantages:
  - Will need to add non-pre-trained bidirectional model on top
  - Right-to-left SQuAD model doesn't see question
  - Need to train two models
  - Off-by-one: LTR predicts next word, RTL predicts previous word
  - Not trivial to add arbitrary pre-training tasks.

Common Questions

- Why did no one think of this before?
- Better question: Why wasn't contextual pre-training popular before 2018 with ELMo?
- Good results on pre-training is >1,000x to 100,000 more expensive than supervised training.
  - E.g., 10x-100x bigger model trained for 100x-1,000x as many steps.
  - Imagine it's 2013: Well-tuned 2-layer, 512-dim LSTM sentiment analysis gets 80% accuracy, training for 8 hours.
  - Pre-train LM on same architecture for a week, get 80.5%.
  - Conference reviewers: "Who would do something so expensive for such a small gain?"

Common Questions

- The model must be learning more than “contextual embeddings”
- Alternate interpretation: Predicting missing words (or next words) requires learning many types of language understanding features.
  - syntax, semantics, pragmatics, coreference, etc.
- Implication: Pre-trained model is much bigger than it needs to be to solve specific task
- Task-specific model distillation words very well

Common Questions

- Is modeling “solved” in NLP? I.e., is there a reason to come up with novel model architectures?
  - But that’s the most fun part of NLP research :(
- Maybe yes, for now, on some tasks, like SQuAD-style QA.
  - At least using the same deep learning “lego blocks”
- Examples of NLP models that are not “solved”:
  - Models that minimize total training cost vs. accuracy on modern hardware
  - Models that are very parameter efficient (e.g., for mobile deployment)
  - Models that represent knowledge/context in latent space
  - Models that represent structured data (e.g., knowledge graph)
  - Models that jointly represent vision and language

Common Questions

- Personal belief: Near-term improvements in NLP will be mostly about making clever use of “free” data.
  - Unsupervised vs. semi-supervised vs. synthetic supervised is somewhat arbitrary.
  - “Data I can get a lot of without paying anyone” vs. “Data I have to pay people to create” is more pragmatic distinction.
- No less “prestigious” than modeling papers:
  - *Phrase-Based & Neural Unsupervised Machine Translation*, Facebook AI Research, EMNLP 2018 Best Paper

Conclusions

- Empirical results from BERT are great, but biggest impact on the field is:
- With pre-training, bigger == better, without clear limits (so far).
- Unclear if adding things on top of BERT really helps by very much.
  - Good for people and companies building NLP systems.
  - Not necessary a “good thing” for researchers, but important.