Neural Machine Translation

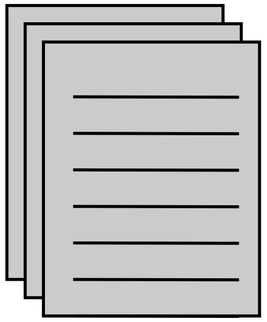
Wei Xu

(many slides from Greg Durrett and Emma Strubell)

Recap: Phrase-Based MT

cat ||| chat ||| 0.9
the cat ||| le chat ||| 0.8
dog ||| chien ||| 0.8
house ||| maison ||| 0.6
my house ||| ma maison ||| 0.9
language ||| langue ||| 0.9
…

Phrase table P(f|e)



Unlabeled English data

Language model P(e)

$$P(e|f) \propto P(f|e)P(e)$$

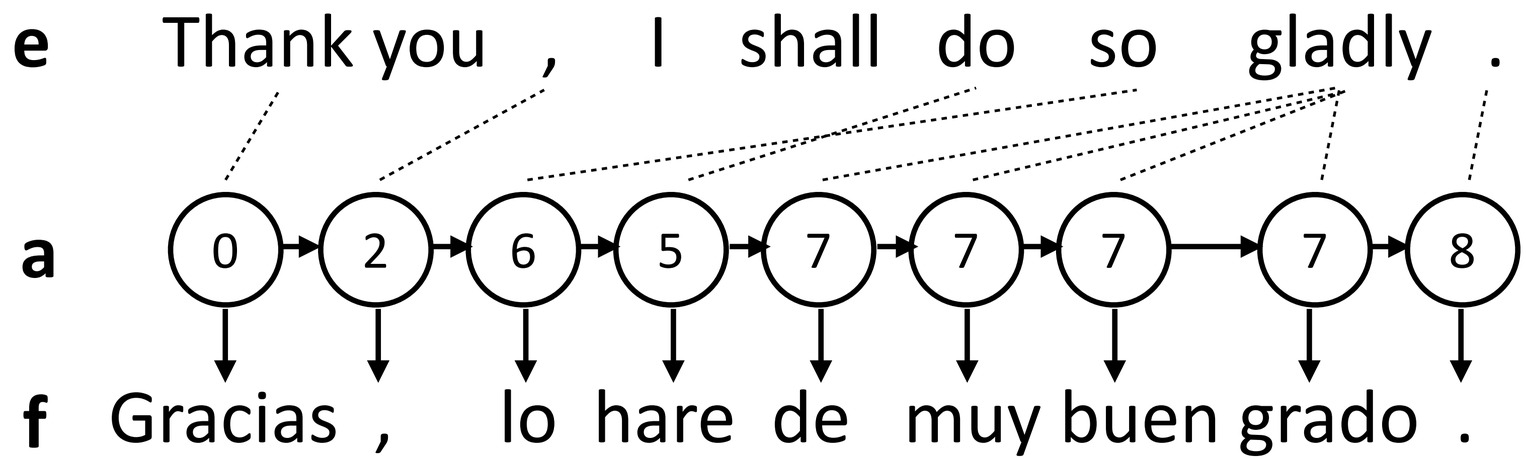
Noisy channel model: combine scores from translation model + language model to translate foreign to English

"Translate faithfully but make fluent English"

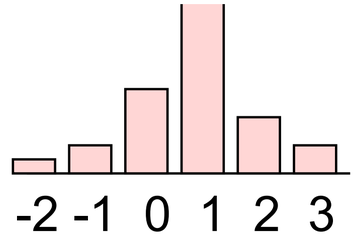
Recap: HMM for Alignment

- Sequential dependence between a's to capture monotonicity

$$P(\mathbf{f}, \mathbf{a}|\mathbf{e}) = \prod_{i=1}^{n} P(f_i|e_{a_i})P(a_i|a_{i-1})$$



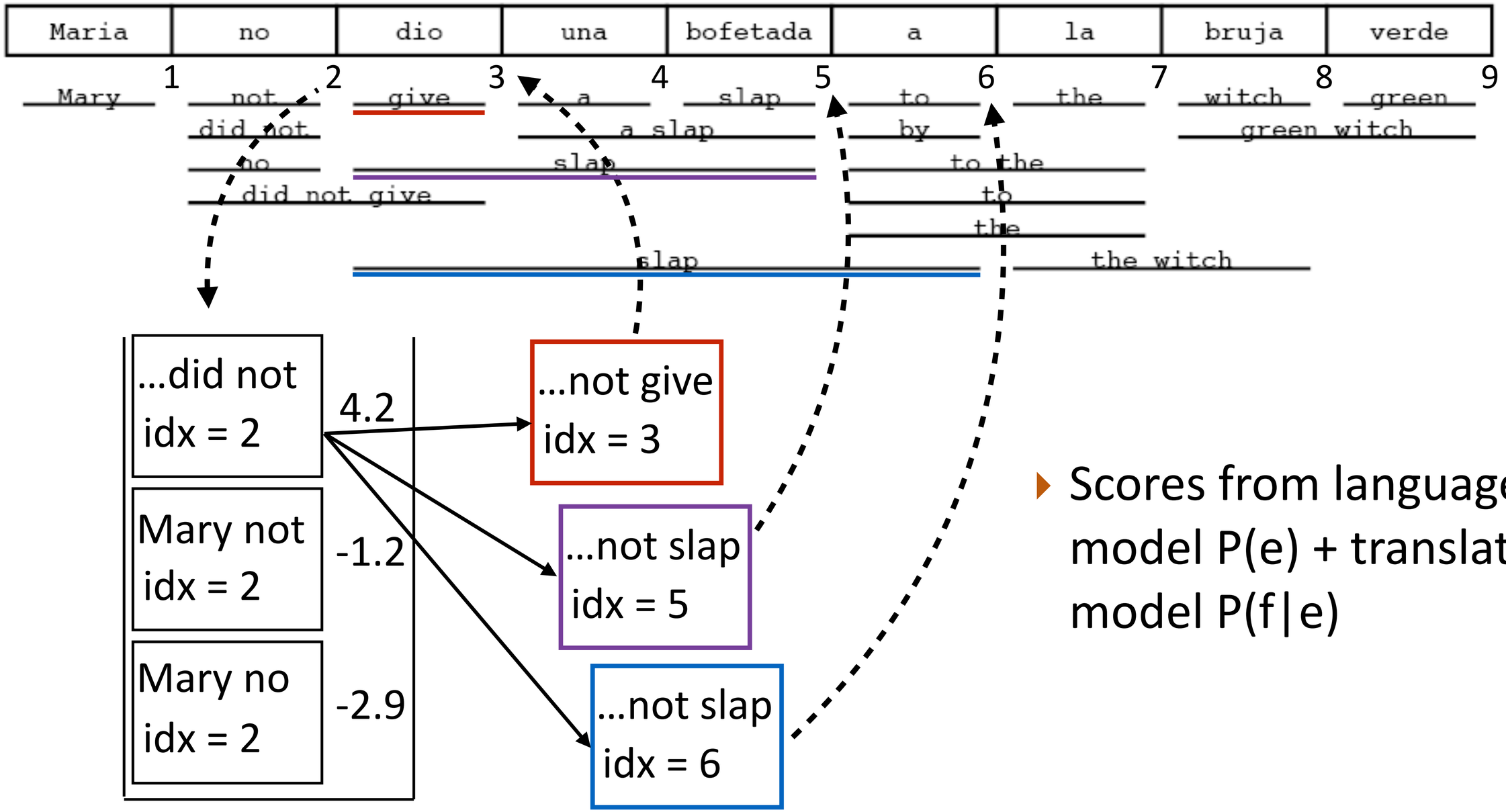
- Alignment dist parameterized by jump size: $P(a_j - a_{j-1})$



- $P(f_i|e_{a_i})$: word translation table

Brown et al. (1993)

Recap: Beam Search for Decoding



- Scores from language model P(e) + translation model P(f|e)

Recap: Evaluating MT

- Fluency: does it sound good in the target language?
- Fidelity/adequacy: does it capture the meaning of the original?
- BLEU score: geometric mean of 1-, 2-, 3-, and 4-gram precision vs. a reference, multiplied by brevity penalty

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right).$$

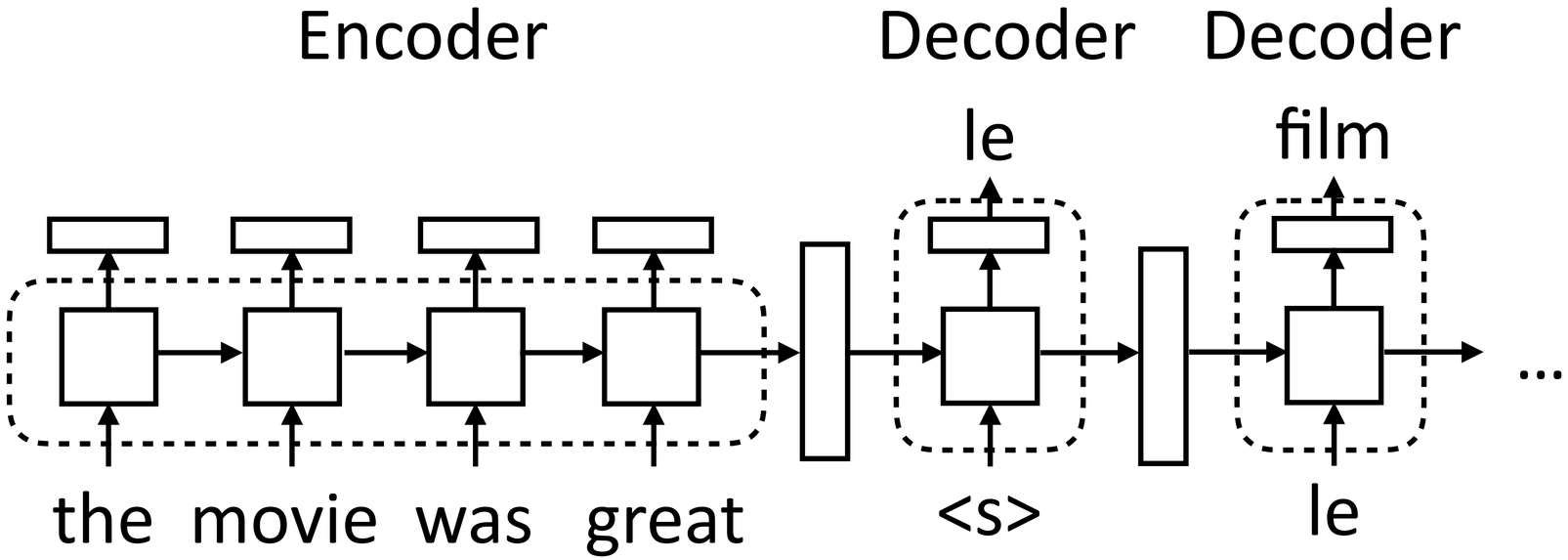
- Typically $n = 4$, $w_i = 1/4$

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}.$$

- r = length of reference
  c = length of prediction

- Does this capture fluency and adequacy?

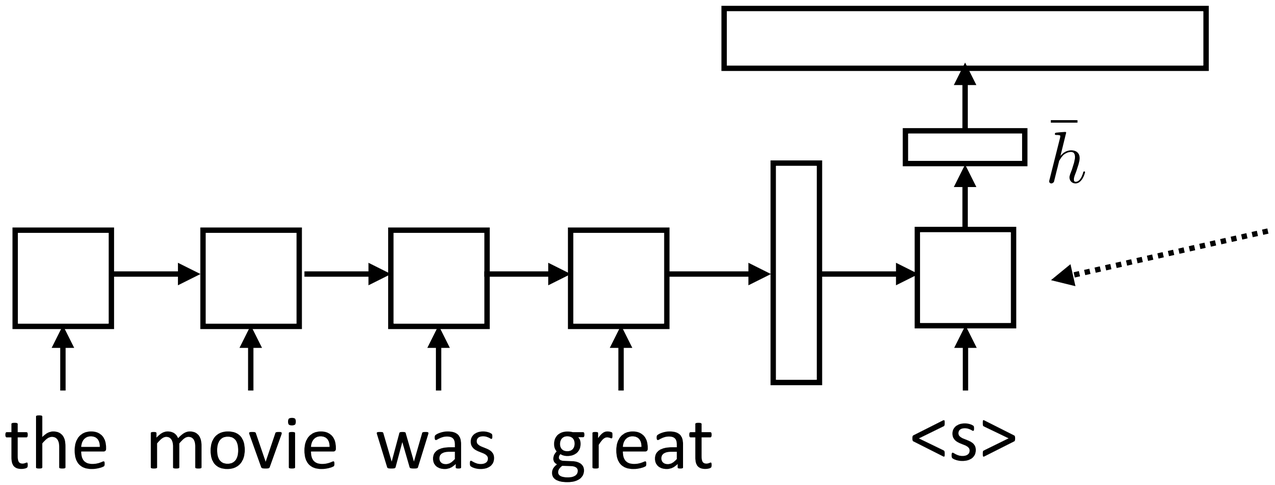
Recap: seq2seq Models



- Encoder: consumes sequence of tokens, produces a vector. Analogous to encoders for classification/tagging tasks
- Decoder: separate module, single cell. Takes two inputs: hidden state (vector *h* or tuple (*h*, *c*)) and previous token. Outputs token + new state

Recap: seq2seq Models

- Generate next word conditioned on previous word as well as hidden state
- W size is |vocab| x |hidden state|, softmax over entire vocabulary



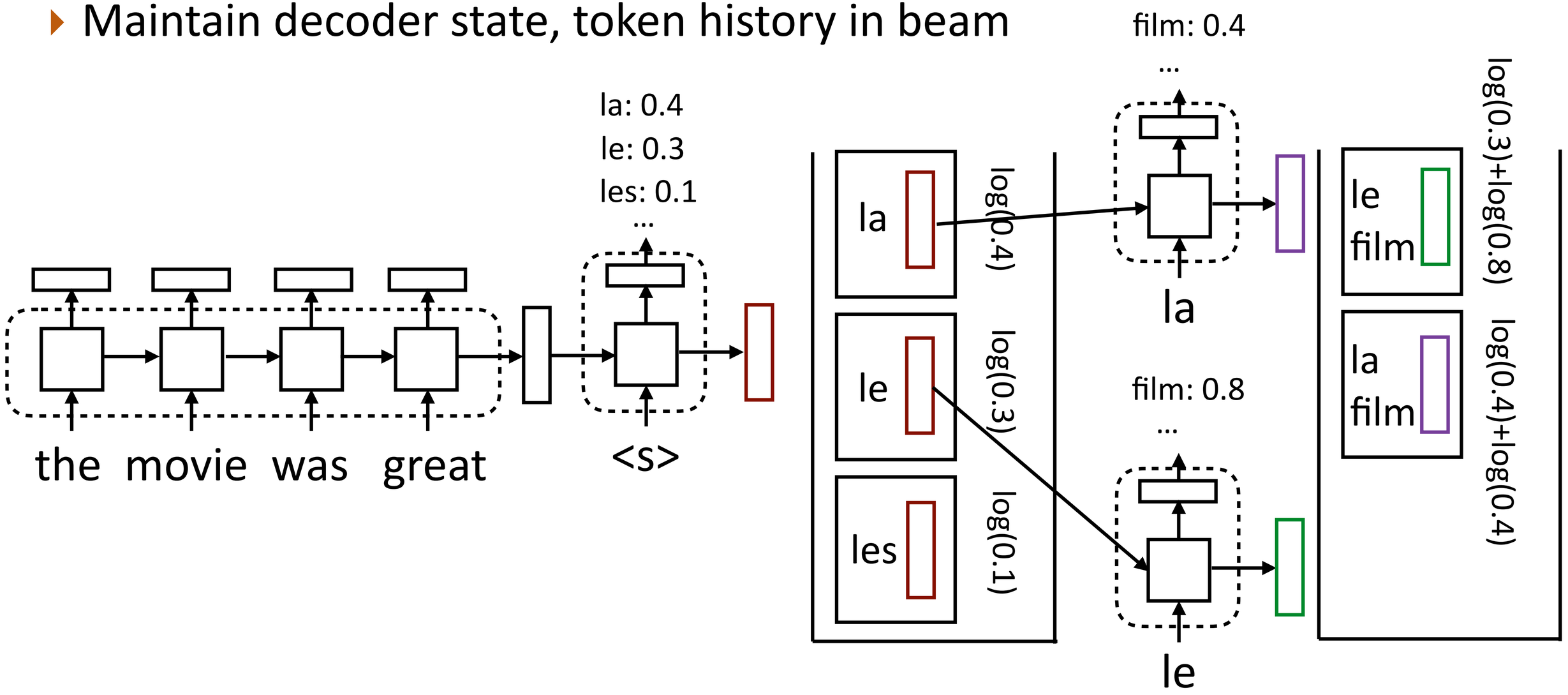
$$P(y_i|\mathbf{x}, y_1, \ldots, y_{i-1}) = \text{softmax}(W\bar{h})$$

$$P(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} P(y_i|\mathbf{x}, y_1, \ldots, y_{i-1})$$

Decoder has separate parameters from encoder, so this can learn to be a language model (produce a plausible next word given current one)

Recap: Beam Search for decoding

- Maintain decoder state, token history in beam



- Keep **both** *film* states! Hidden state vectors are different

Recap: NL-to-SQL Generation

- Convert natural language description into a SQL query against some DB

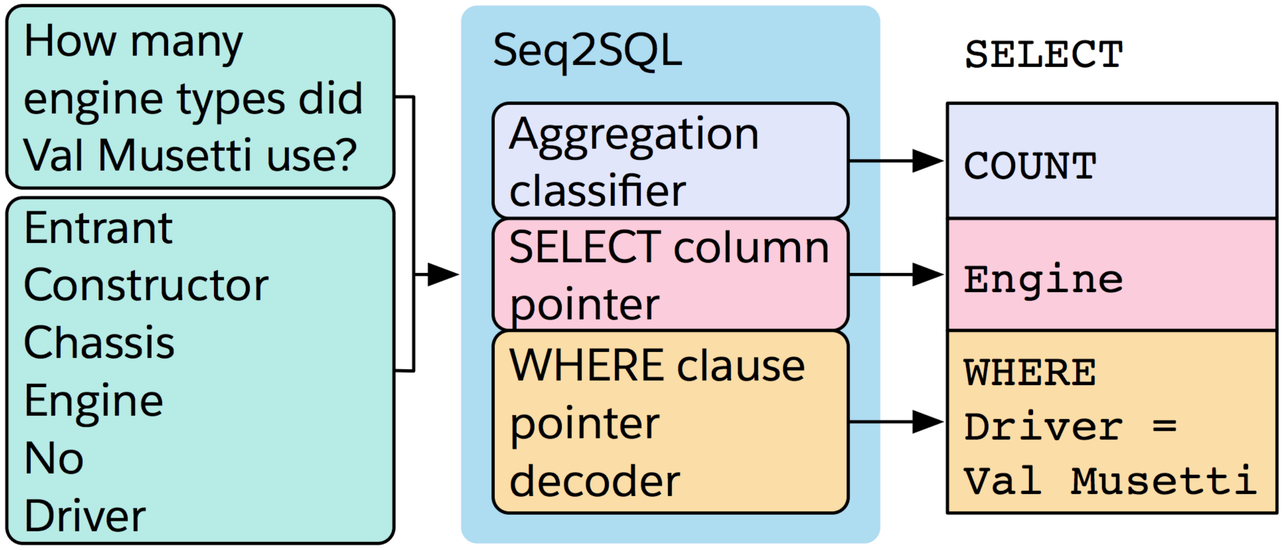
Question:

How many CFL teams are from York College?

SQL:

```
SELECT COUNT CFL Team FROM
CFLDraft WHERE College = “York”
```

- How to ensure that well-formed SQL is generated?
  - Three components
- How to capture column names + constants?
  - Pointer mechanisms



Zhong et al. (2017)

Fairseq

Decoder

Our Decoder will predict the next word, conditioned on the Encoder's final hidden state and an embedded representation of the previous target word – which is sometimes called *teacher forcing*. More specifically, we'll use a `torch.nn.LSTM` to produce a sequence of hidden states that we'll project to the size of the output vocabulary to predict each target word.

```
import torch
from fairseq.models import FairseqDecoder

class SimpleLSTMDecoder(FairseqDecoder):

    def __init__(
        self, dictionary, encoder_hidden_dim=128, embed_dim=128, hidden_dim=128,
        dropout=0.1,
    ):
        super().__init__(dictionary)

        # Our decoder will embed the inputs before feeding them to the LSTM.
        self.embed_tokens = nn.Embedding(
            num_embeddings=len(dictionary),
            embedding_dim=embed_dim,
            padding_idx=dictionary.pad(),
        )
        self.dropout = nn.Dropout(p=dropout)

        # We'll use a single-layer, unidirectional LSTM for simplicity.
        self.lstm = nn.LSTM(
            # For the first layer we'll concatenate the Encoder's final hidden
            # state with the embedded target tokens.
            input_size=encoder_hidden_dim + embed_dim,
            hidden_size=hidden_dim,
            num_layers=1,
            bidirectional=False,
        )

        # Define the output projection.
        self.output_projection = nn.Linear(hidden_dim, len(dictionary))
```

Neural MT Details

Encoder-Decoder MT

- Sutskever seq2seq paper: first major application of LSTMs to NLP
- Basic encoder-decoder with beam search

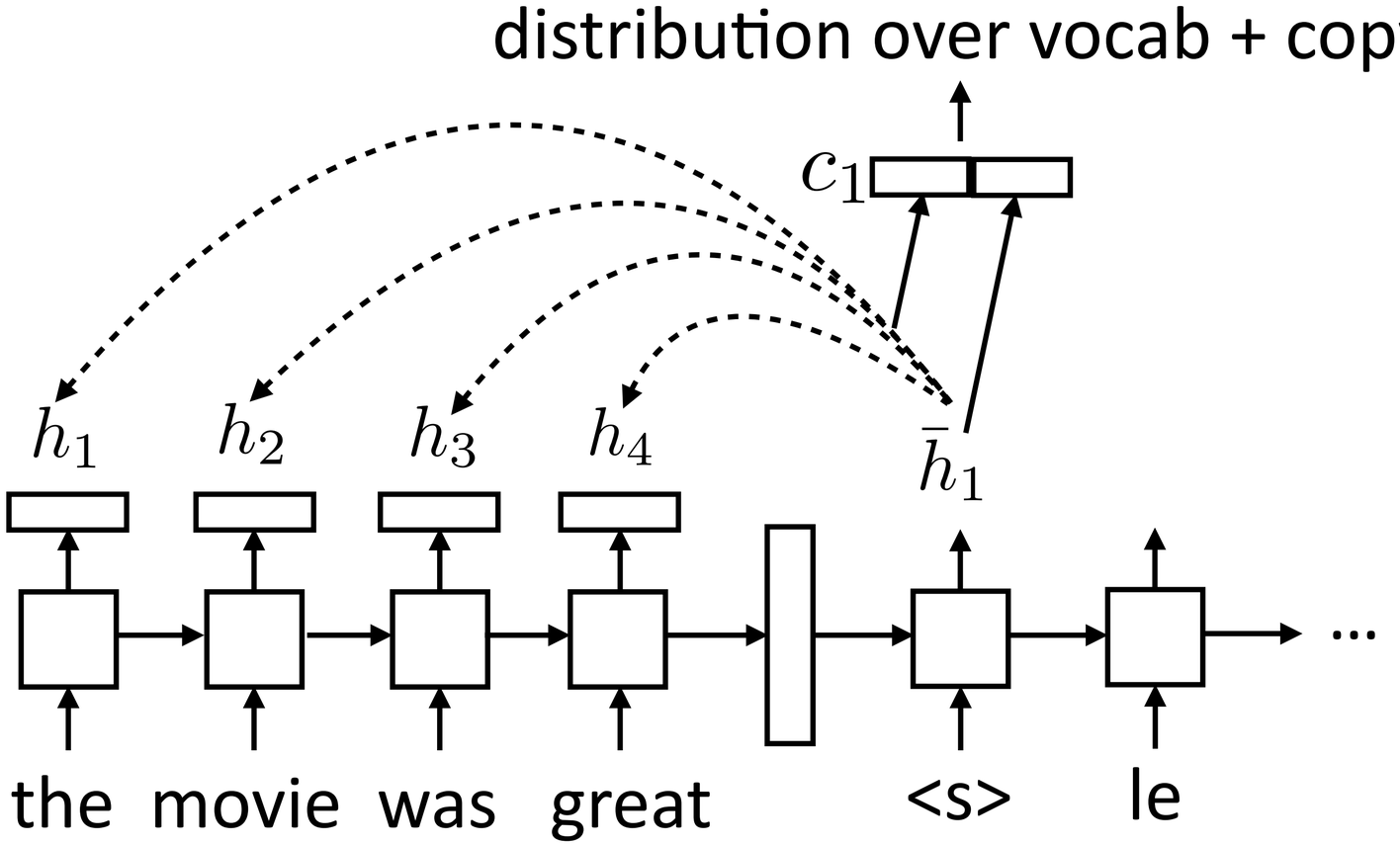
Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	**34.81**

- SOTA = 37.0 — not all that competitive…

Sutskever et al. (2014)

Encoder-Decoder MT

- Better model from seq2seq lectures: encoder-decoder with **attention** and **copying** for rare words



$$e_{ij} = f(\bar{h}_i, h_j)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{j'} \exp(e_{ij'})}$$

$$c_i = \sum_j \alpha_{ij} h_j$$

$$P(y_i | \mathbf{x}, y_1, \ldots, y_{i-1}) = \text{softmax}(W[c_i; \bar{h}_i])$$

Results: WMT English-French

- 12M sentence pairs

Classic phrase-based system: ~**33** BLEU, uses additional target-language data

Rerank with LSTMs: **36.5** BLEU (long line of work here; Devlin+ 2014)

Sutskever+ (2014) seq2seq single: **30.6** BLEU

Sutskever+ (2014) seq2seq ensemble: **34.8** BLEU

Luong+ (2015) seq2seq ensemble with attention and rare word handling: **37.5** BLEU

- But English-French is a really easy language pair and there's *tons* of data for it! Does this approach work for anything harder?

Results: WMT English-German

- 4.5M sentence pairs

Classic phrase-based system: **20.7** BLEU

Luong+ (2014) seq2seq: **14** BLEU

Luong+ (2015) seq2seq ensemble with rare word handling: **23.0** BLEU

- Not nearly as good in absolute BLEU, but not really comparable across languages
- French, Spanish = easiest
  German, Czech = harder
  Japanese, Russian = hard (grammatically different, lots of morphology…)

MT Examples

src	In einem Interview sagte Bloom jedoch , dass er und Kerr sich noch immer lieben .
ref	However , in an interview , Bloom has said that he and *Kerr* still love each other .
best	In an interview , however , Bloom said that he and *Kerr* still love .
base	However , in an interview , Bloom said that he and **Tina** were still <unk> .

- best = with attention, base = no attention
- NMT systems can hallucinate words, especially when not using attention — phrase-based doesn't do this

Luong et al. (2015)

MT Examples

src	Wegen der von Berlin und der Europäischen Zentralbank verhängten strengen Sparpolitik in Verbindung mit der Zwangsjacke , in die die jeweilige nationale Wirtschaft durch das Festhalten an der gemeinsamen Währung genötigt wird , sind viele Menschen der Ansicht , das Projekt Europa sei zu weit gegangen
ref	The *austerity imposed by Berlin and the European Central Bank , coupled with the straitjacket* imposed on national economies through adherence to the common currency , has led many people to think Project Europe has gone too far .
best	Because of the strict *austerity measures imposed by Berlin and the European Central Bank in connection with the straitjacket* in which the respective national economy is forced to adhere to the common currency , many people believe that the European project has gone too far .
base	Because of the pressure **imposed by the European Central Bank and the Federal Central Bank with the strict austerity** imposed on the national economy in the face of the single currency , many people believe that the European project has gone too far .

- best = with attention, base = no attention

Luong et al. (2015)

MT Examples

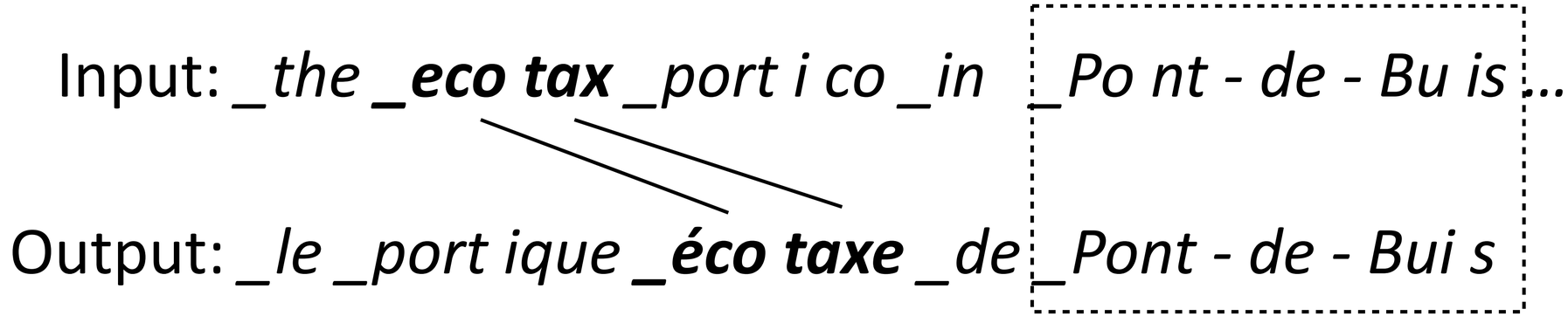
Source	such changes in reaction conditions include , but are not limited to , an increase in temperature or change in ph .
Reference	所(such) 述(said) 反 应(reaction) 条 件(condition) 的(of) 改 变(change) 包 括(include) 但(but) 不(not) 限 于(limit) 温度(temperature) 的(of) 增加(increase) 或(or) pH 值(value) 的(of) 改变(change) 。
PBMT	中(in) 的(of) 这 种(such) 变 化(change) 的(of) 反 应(reaction) 条 件(condition) 包 括(include) , 但(but) 不(not) 限 于(limit) , 增加(increase) 的(of) 温度(temperature) 或(or) pH 变化(change) 。
NMT	这种(such) 反应(reaction) 条件(condition) 的(of) 变化(change) 包括(include) 但(but) 不(not) 限于(limit) pH 或(or) pH 的(of) 变化(change) 。

- NMT can repeat itself if it gets confused (pH or pH)
- Phrase-based MT often gets chunks right, may have more subtle ungrammaticalities

Zhang et al. (2017)

Handling Rare Words

- Words are a difficult unit to work with: copying can be cumbersome, word vocabularies get very large
- Character-level models don't work well
- Solution: "word pieces" (which may be full words but may be subwords)



- Can help with transliteration; capture shared linguistic characteristics between languages (e.g., transliteration, shared word root, etc.)

Wu et al. (2016)

Byte Pair Encoding (BPE)

- Start with every individual byte (basically character) as its own symbol

```
for i in range(num_merges):
  pairs = get_stats(vocab)
  best = max(pairs, key=pairs.get)
  vocab = merge_vocab(best, vocab)
```

- Count bigram character cooccurrences
- Merge the most frequent pair of adjacent characters

- Do this either over your vocabulary (original version) or over a large corpus (more common version)
- Final vocabulary size is often in 10k ~ 30k range for each language

- Most SOTA NMT systems use this on both source + target

Sennrich et al. (2016)

Word Pieces

while voc size < target voc size:

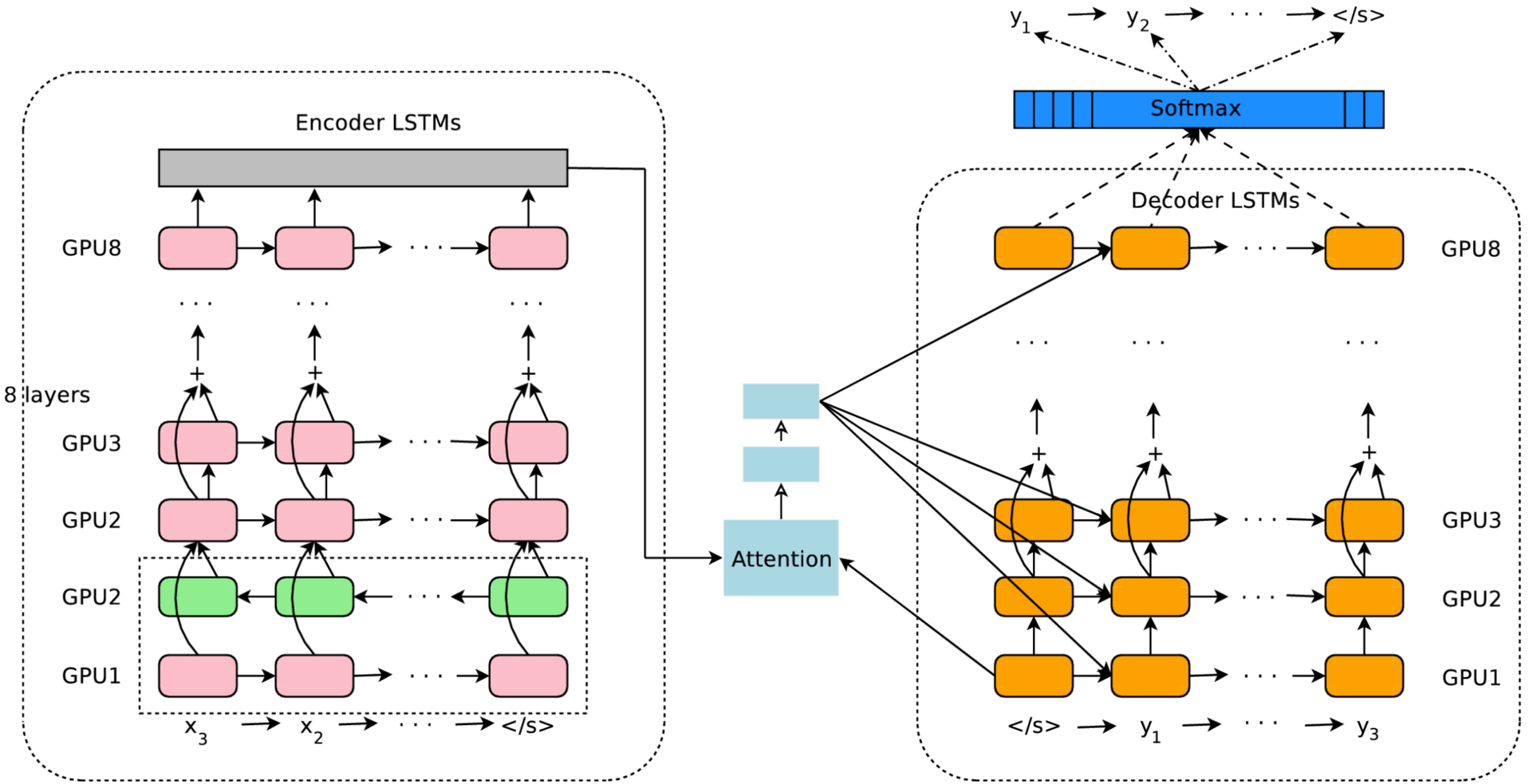
Build a language model over your corpus

Merge pieces that lead to highest improvement in language model perplexity

- SentencePiece library from Google: unigram LM
- Result: way of segmenting input appropriate for translation

Schuster and Nakajima (2012), Wu et al. (2016), Kudo and Richardson (2018)

Google's NMT System



- 8-layer LSTM encoder-decoder with attention, word piece vocabulary of 8k-32k

Wu et al. (2016)

Google's NMT System

English-French:

Google's phrase-based system: 37.0 BLEU

Luong+ (2015) seq2seq ensemble with rare word handling: 37.5 BLEU

Google's 32k word pieces: 38.95 BLEU

English-German:

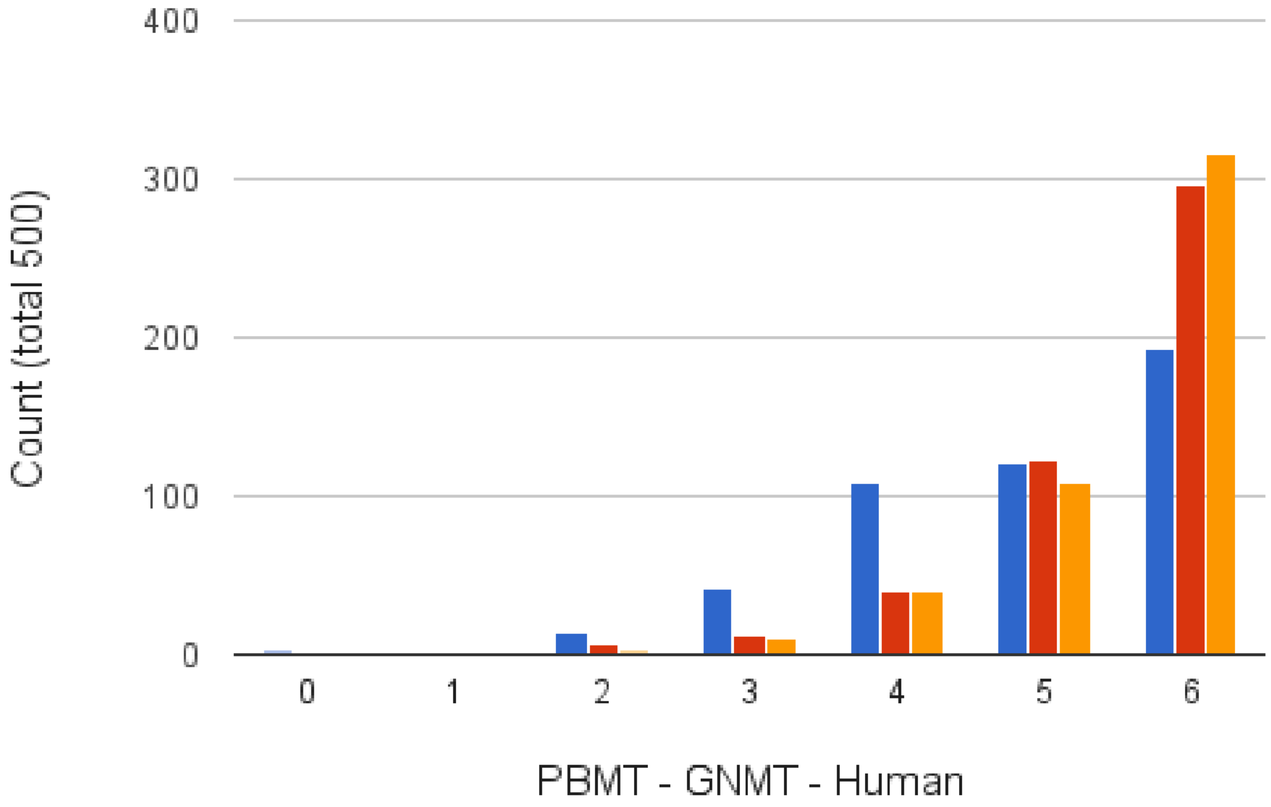
Google's phrase-based system: 20.7 BLEU

Luong+ (2015) seq2seq ensemble with rare word handling: 23.0 BLEU

Google's 32k word pieces: 24.2 BLEU

Wu et al. (2016)

Human Evaluation (En-Es)



- Similar to human-level performance *on English-Spanish*

Figure 6: Histogram of side-by-side scores on 500 sampled sentences from Wikipedia and news websites for a typical language pair, here English → Spanish (PBMT blue, GNMT red, Human orange). It can be seen that there is a wide distribution in scores, even for the human translation when rated by other humans, which shows how ambiguous the task is. It is clear that GNMT is much more accurate than PBMT.

Wu et al. (2016)

Google's NMT System

Source	She was spotted three days later by a dog walker trapped in the quarry	
PBMT	Elle a été repéré trois jours plus tard par un promeneur de chien piégé dans la carrière	6.0
GNMT	Elle a été repérée trois jours plus tard par un traîneau à chiens piégé dans la carrière.	2.0
Human	Elle a été repérée trois jours plus tard par une personne qui promenait son chien coincée dans la carrière	5.0

Gender is correct in GNMT but not in PBMT

"sled"

"walker"

The right-most column shows the human ratings on a scale of 0 (complete nonsense) to 6 (perfect translation)

Wu et al. (2016)

Backtranslation

- Statistical MT methods (e.g., phrase-based MT) used a bilingual corpus of sentences B = (S, T) and a large monolingual corpus T' to train a language model. Can neural MT do the same?
- Approach 1: force the system to generate T' as targets from null inputs

s_1, t_1

s_2, t_2

…

[null], t'_1

[null], t'_2

…

- Approach 2: generate synthetic sources with a T->S machine translation system (backtranslation)

s_1, t_1

s_2, t_2

…

$MT(t'_1)$, t'_1

$MT(t'_2)$, t'_2

…

Sennrich et al. (2015)

Backtranslation

name	training data	instances	BLEU tst2011	tst2012	tst2013	tst2014
baseline (Gülçehre et al., 2015)			18.4	18.8	19.9	18.7
deep fusion (Gülçehre et al., 2015)			20.2	20.2	21.3	**20.6**
baseline	parallel	7.2m	18.6	18.2	18.4	18.3
$\text{parallel}_{\text{synth}}$	parallel/$\text{parallel}_{\text{synth}}$	6m/6m	19.9	20.4	20.1	20.0
$\text{Gigaword}_{\text{mono}}$	parallel/$\text{Gigaword}_{\text{mono}}$	7.6m/7.6m	18.8	19.6	19.4	18.2
$\text{Gigaword}_{\text{synth}}$	parallel/$\text{Gigaword}_{\text{synth}}$	8.4m/8.4m	**21.2**	**21.1**	**21.8**	20.4

Table 6: Turkish→English translation performance (*tokenized* BLEU) on IWSLT test sets (TED talks). Single models. Number of training instances varies due to early stopping.

- Gigaword: large monolingual English corpus
- $\text{parallel}_{\text{synth}}$: backtranslate training data; makes additional noisy source sentences which could be useful

Sennrich et al. (2015)

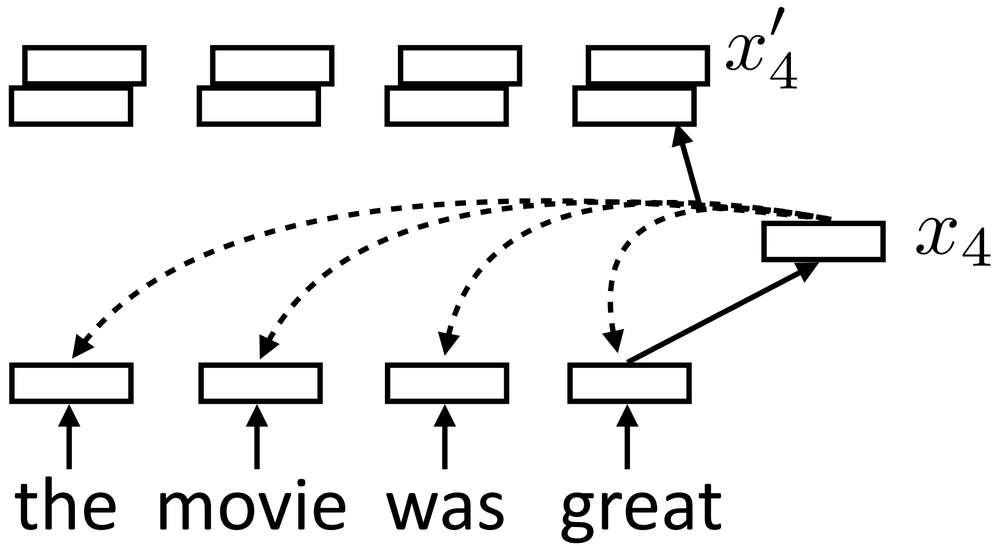
Transformers for MT

Recall: Self-Attention

- Each word forms a "query" which then computes attention over each word

$$\alpha_{i,j} = \text{softmax}(x_i^\top x_j) \quad \text{scalar}$$

$$x'_i = \sum_{j=1}^{n} \alpha_{i,j} x_j \quad \text{vector = sum of scalar * vector}$$

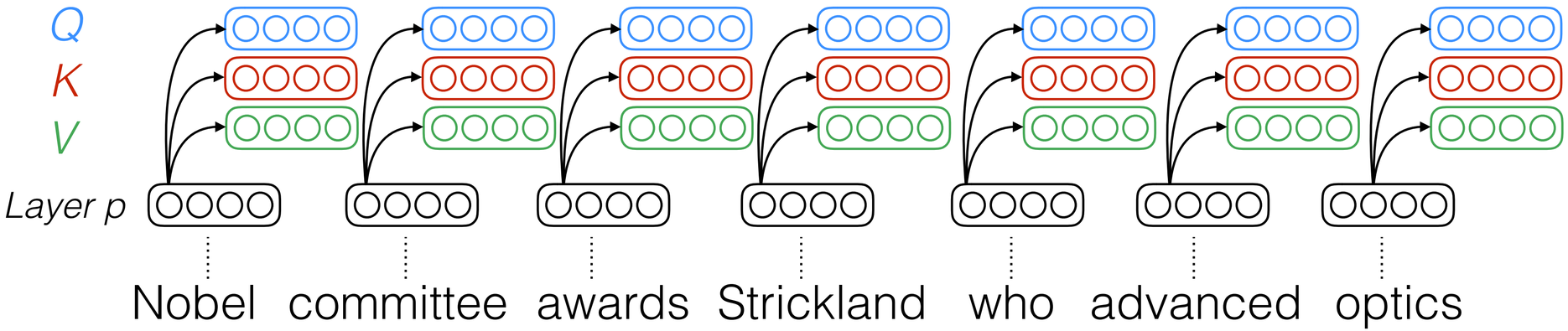


- Multiple "heads" analogous to different convolutional filters. Use parameters W_k and V_k to get different attention values + transform vectors

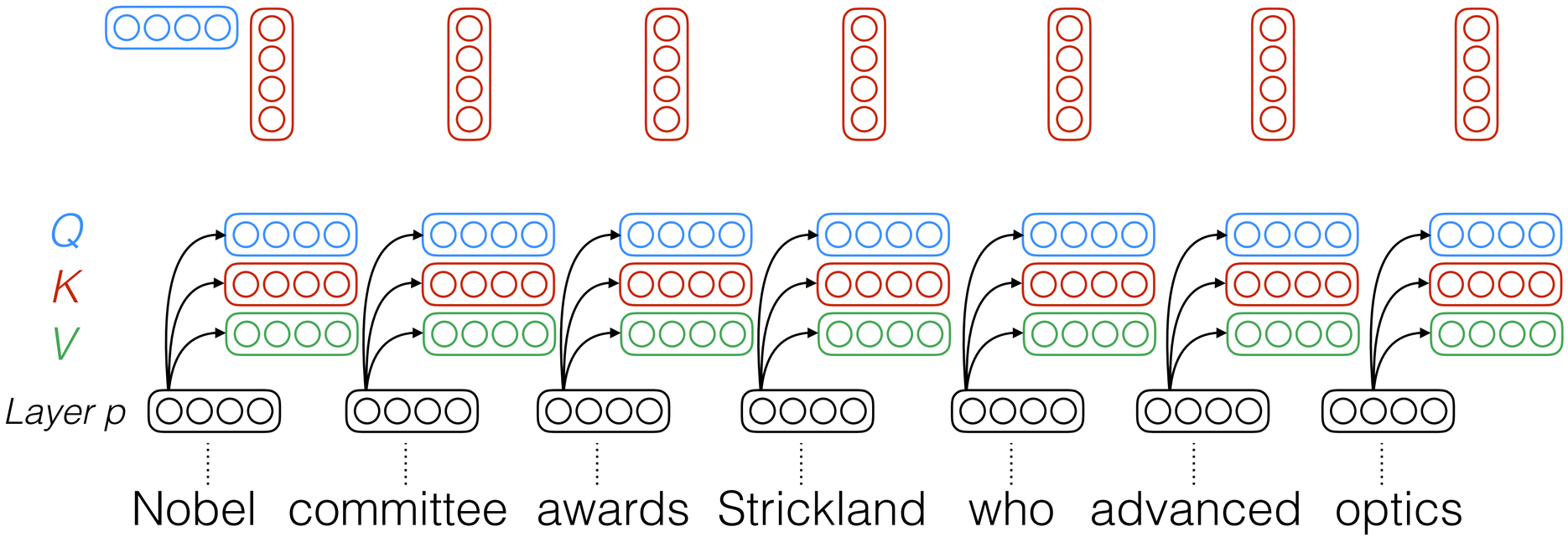
$$\alpha_{k,i,j} = \text{softmax}(x_i^\top W_k x_j) \qquad x'_{k,i} = \sum_{j=1}^{n} \alpha_{k,i,j} V_k x_j$$

Vaswani et al. (2017)

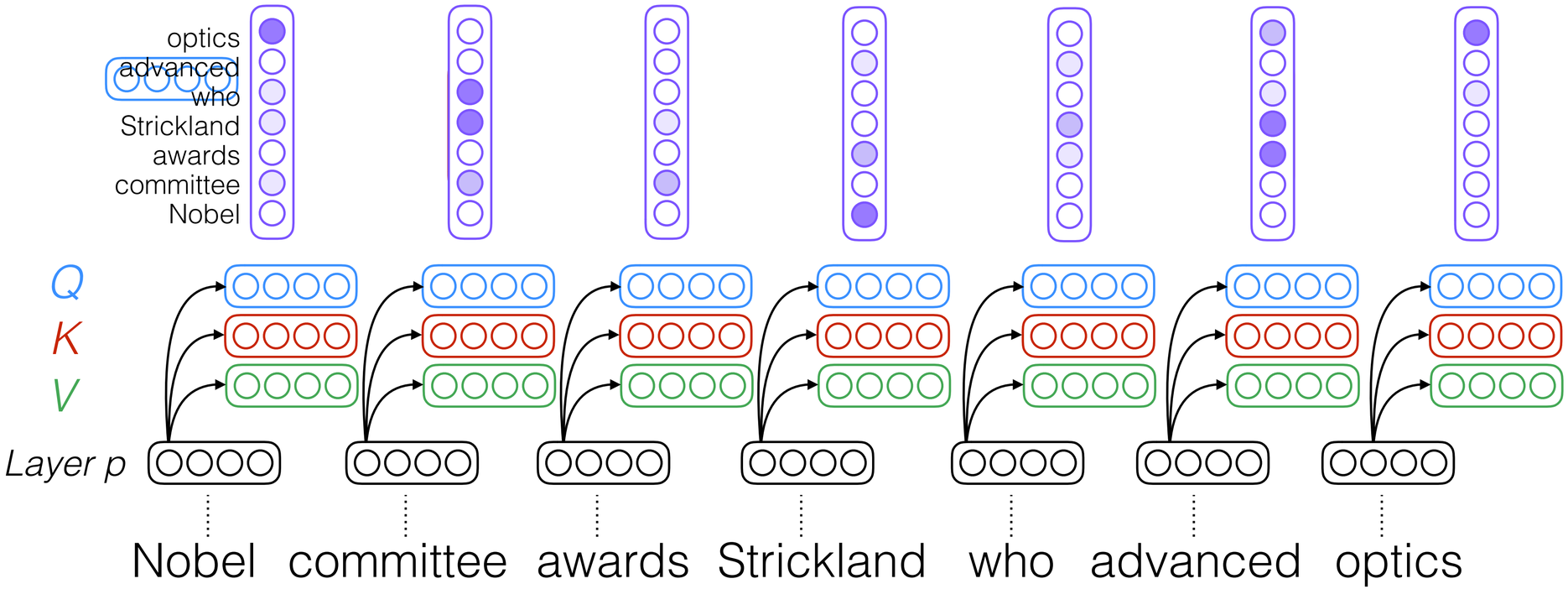
Self-attention



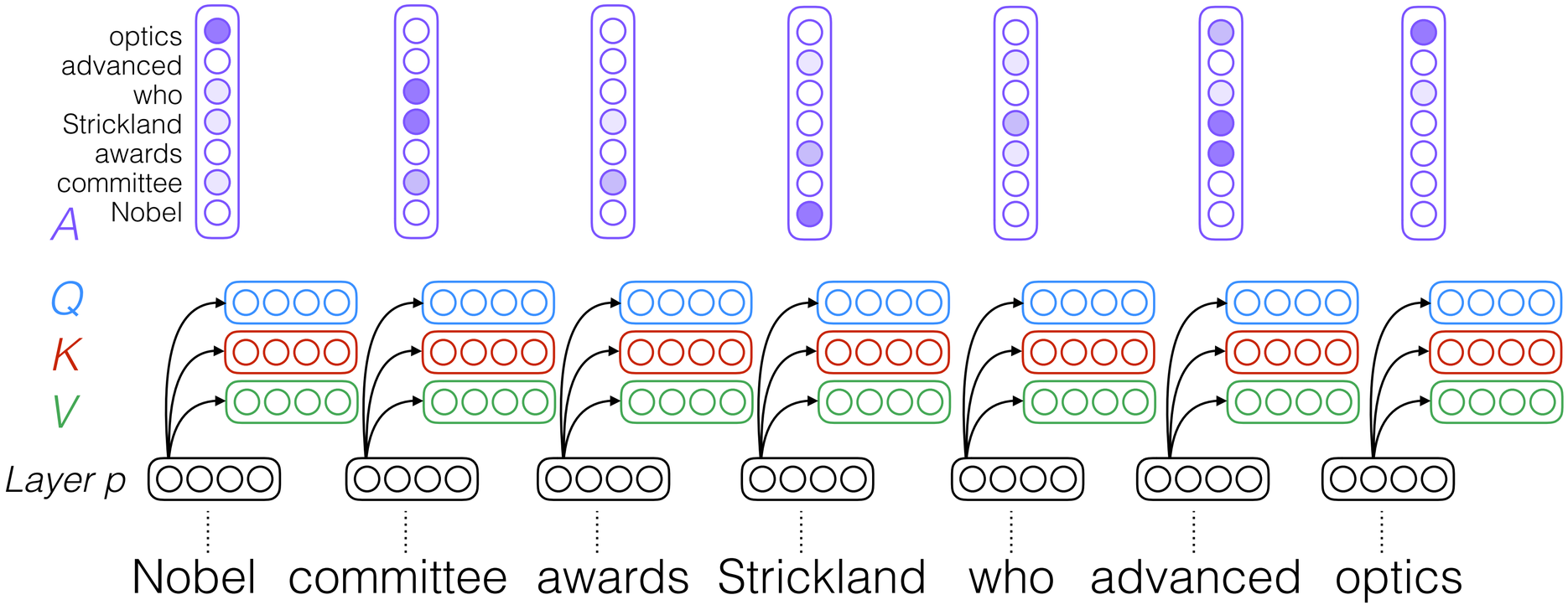
Self-attention



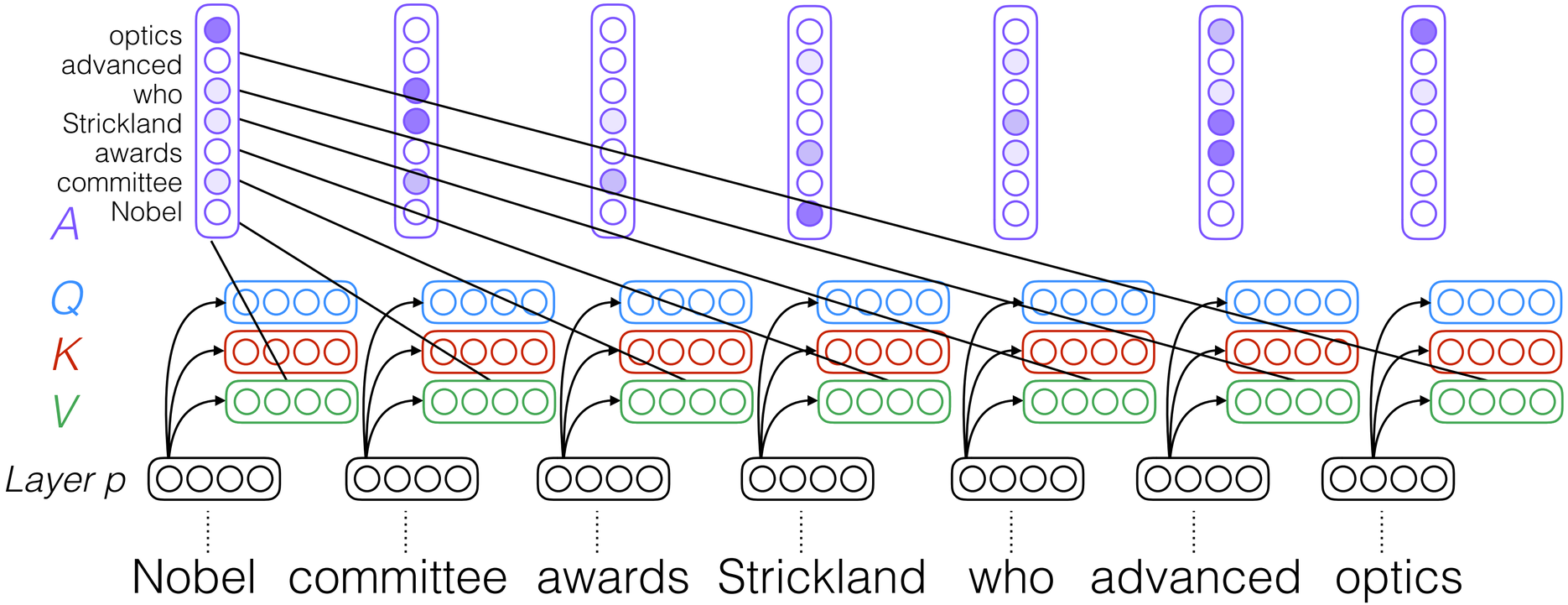
Self-attention



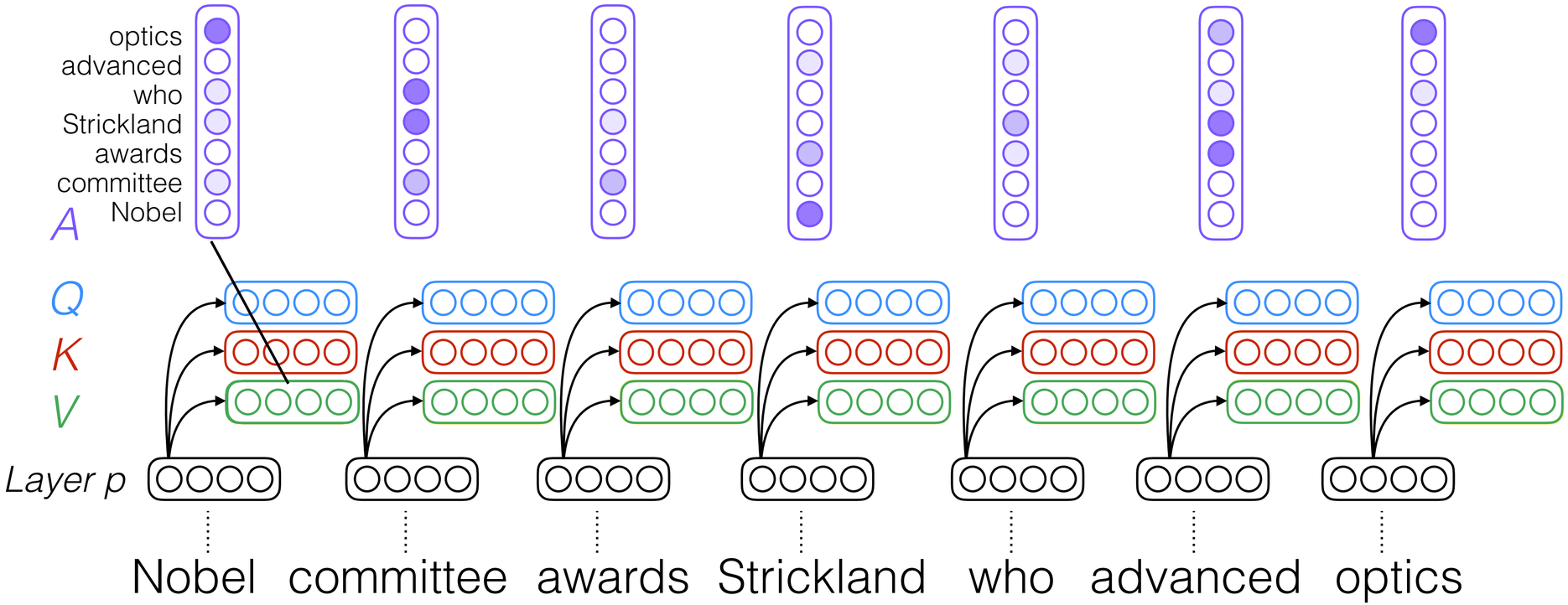
Self-attention



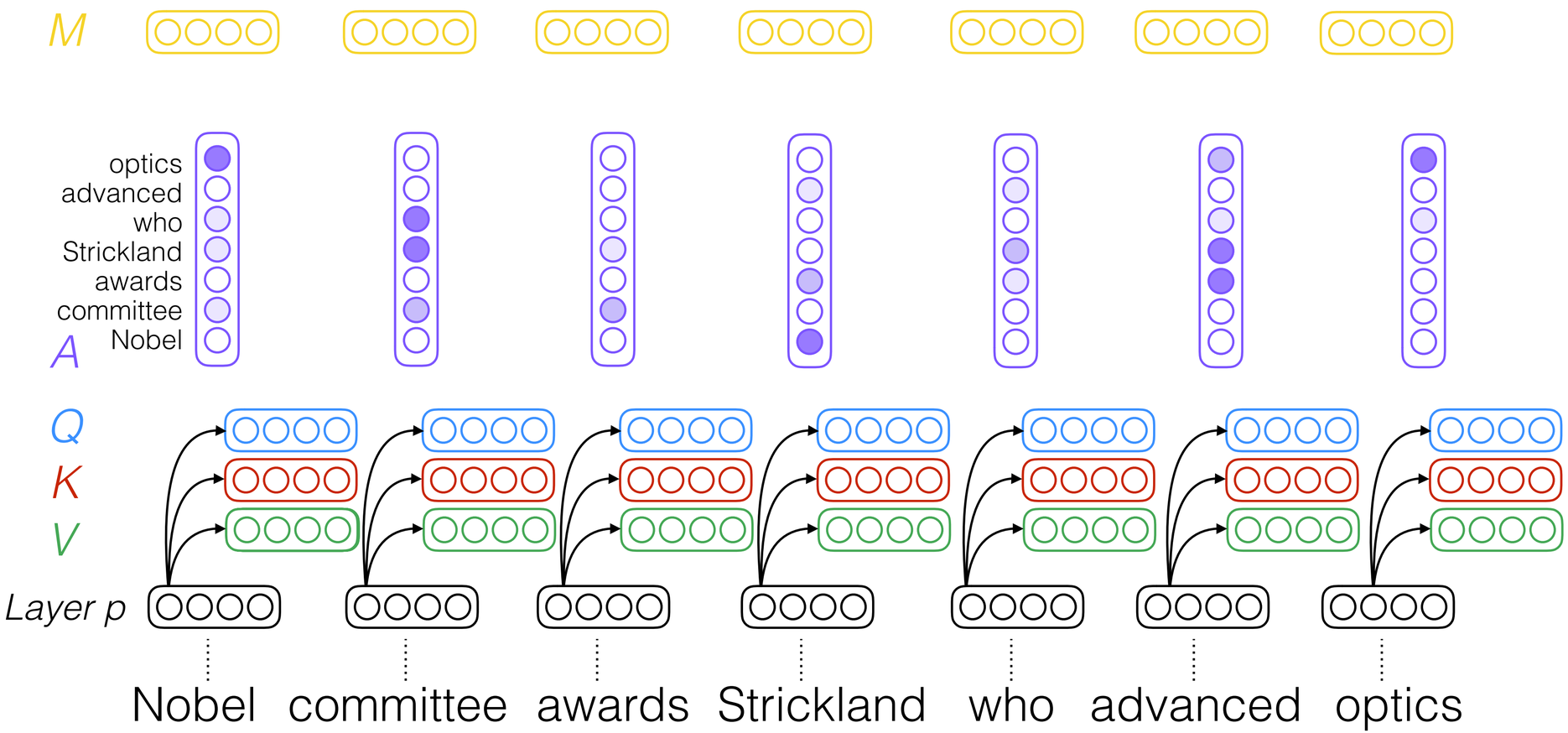
Self-attention



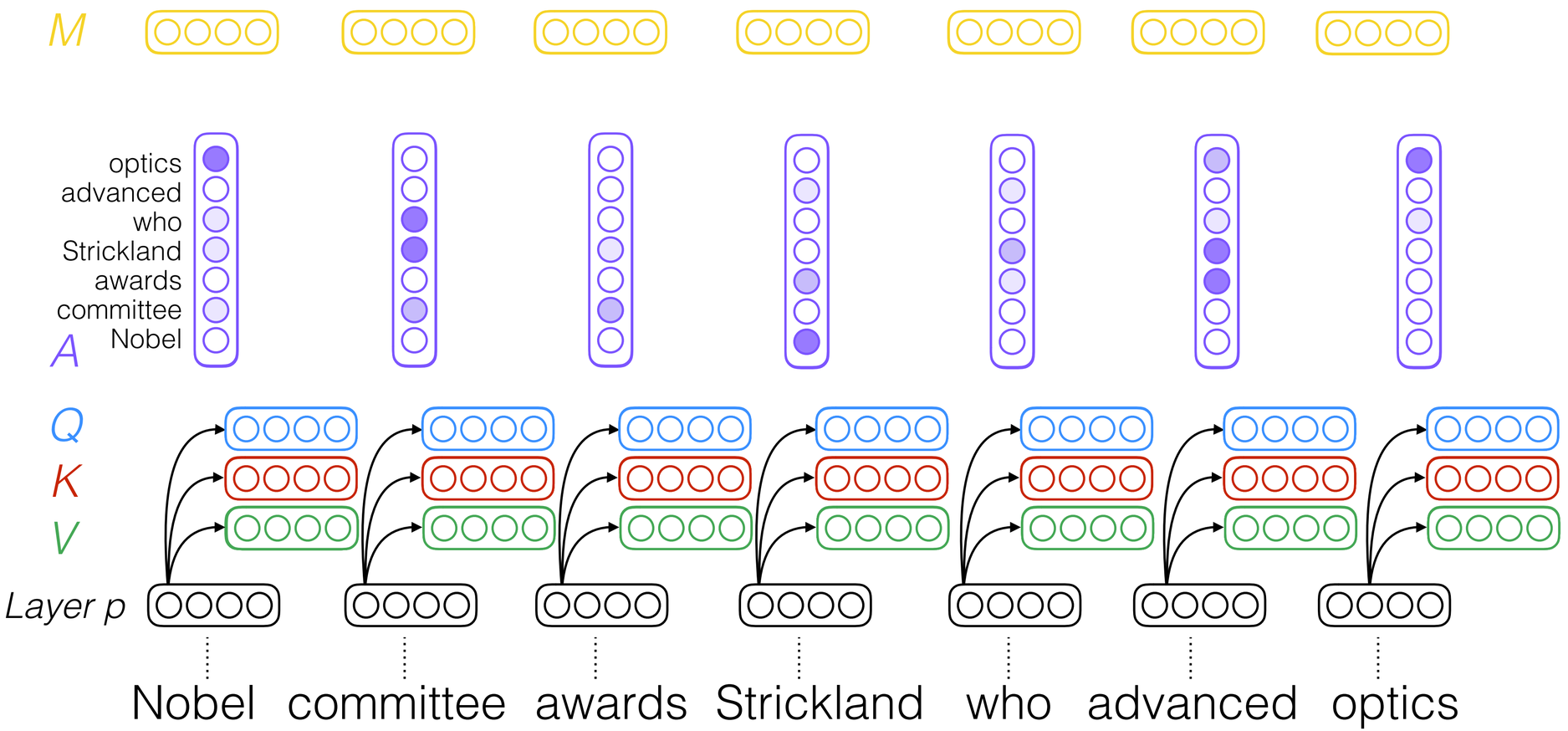
Self-attention



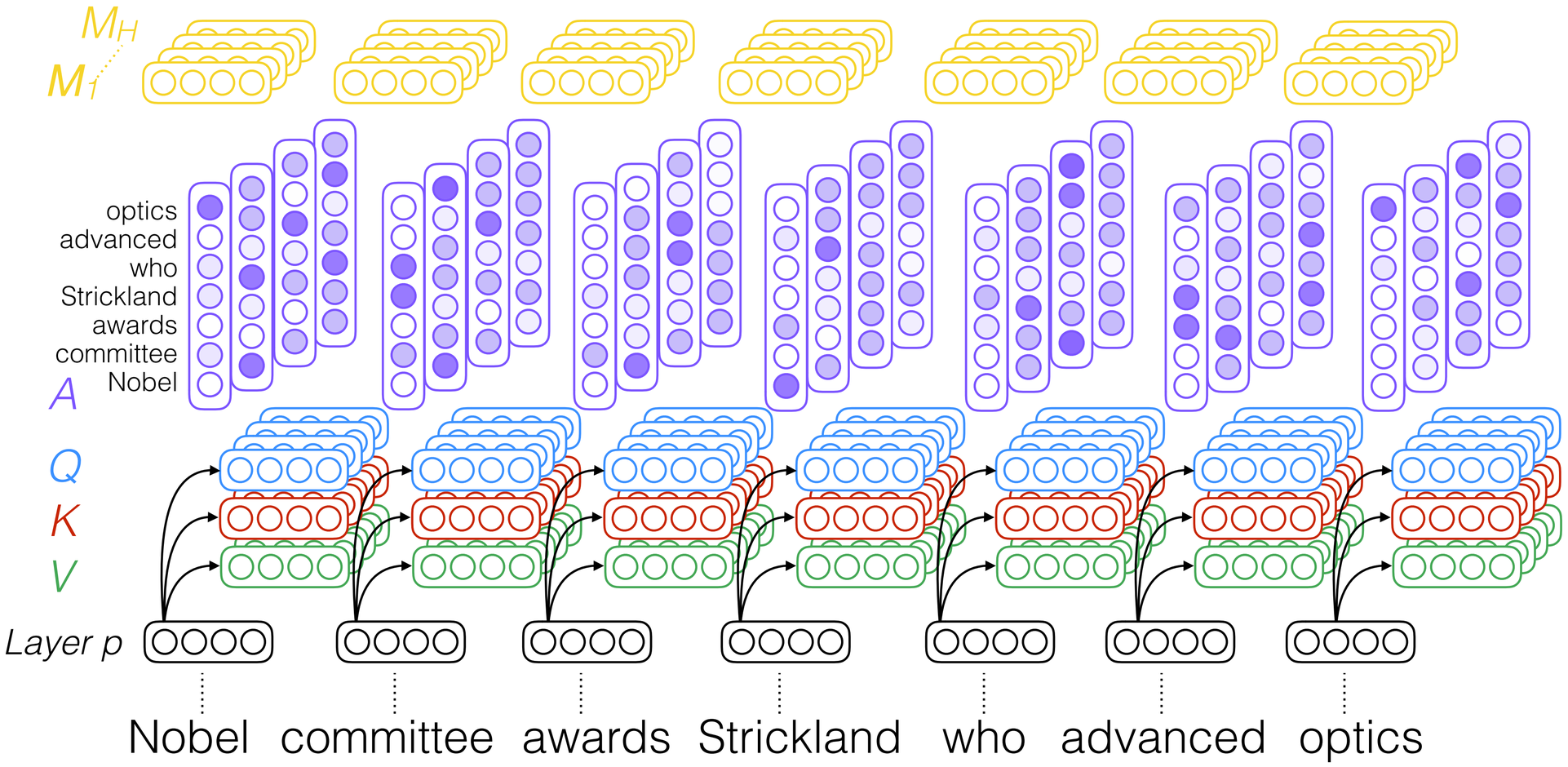
Self-attention



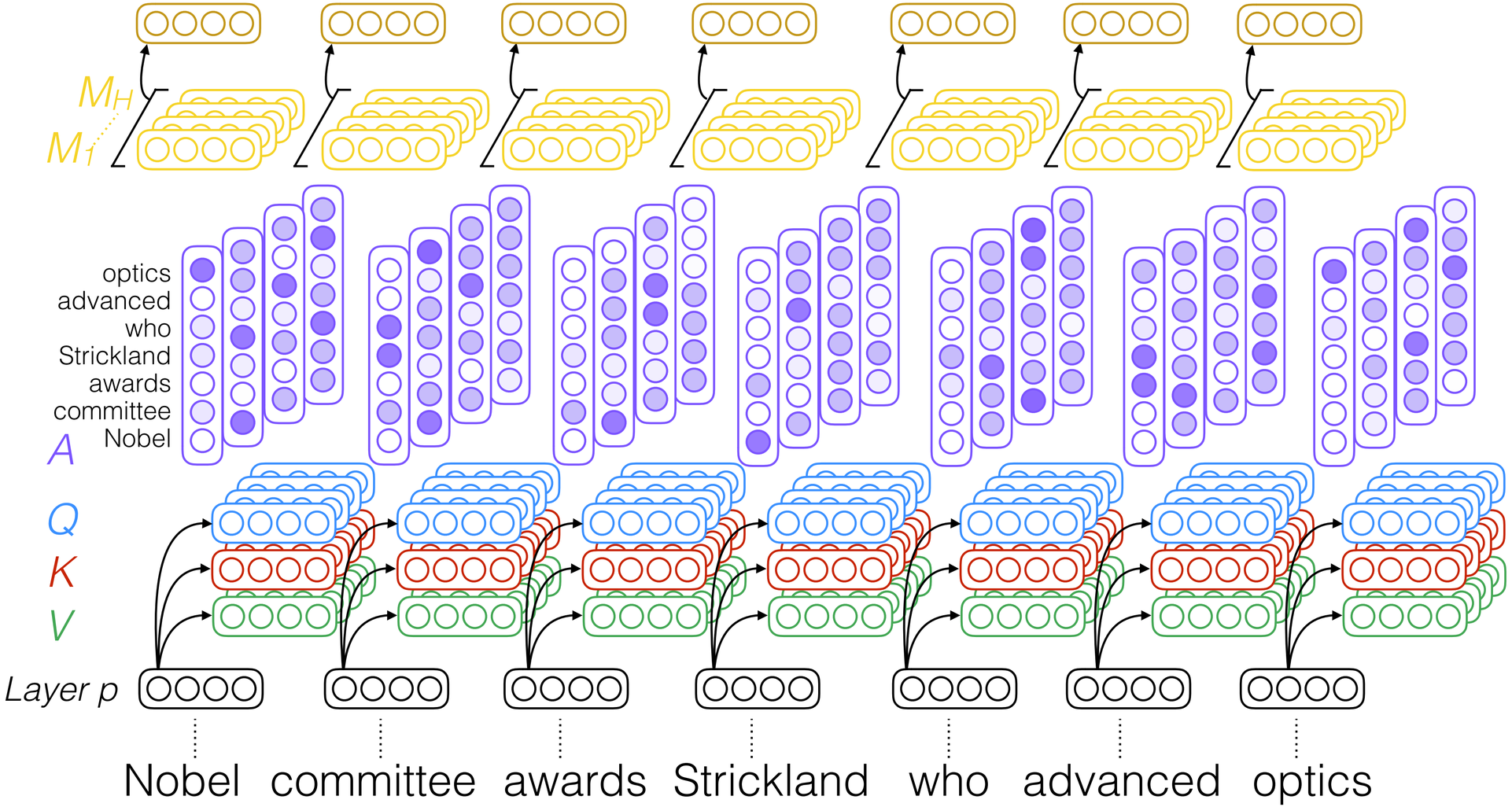
Self-attention



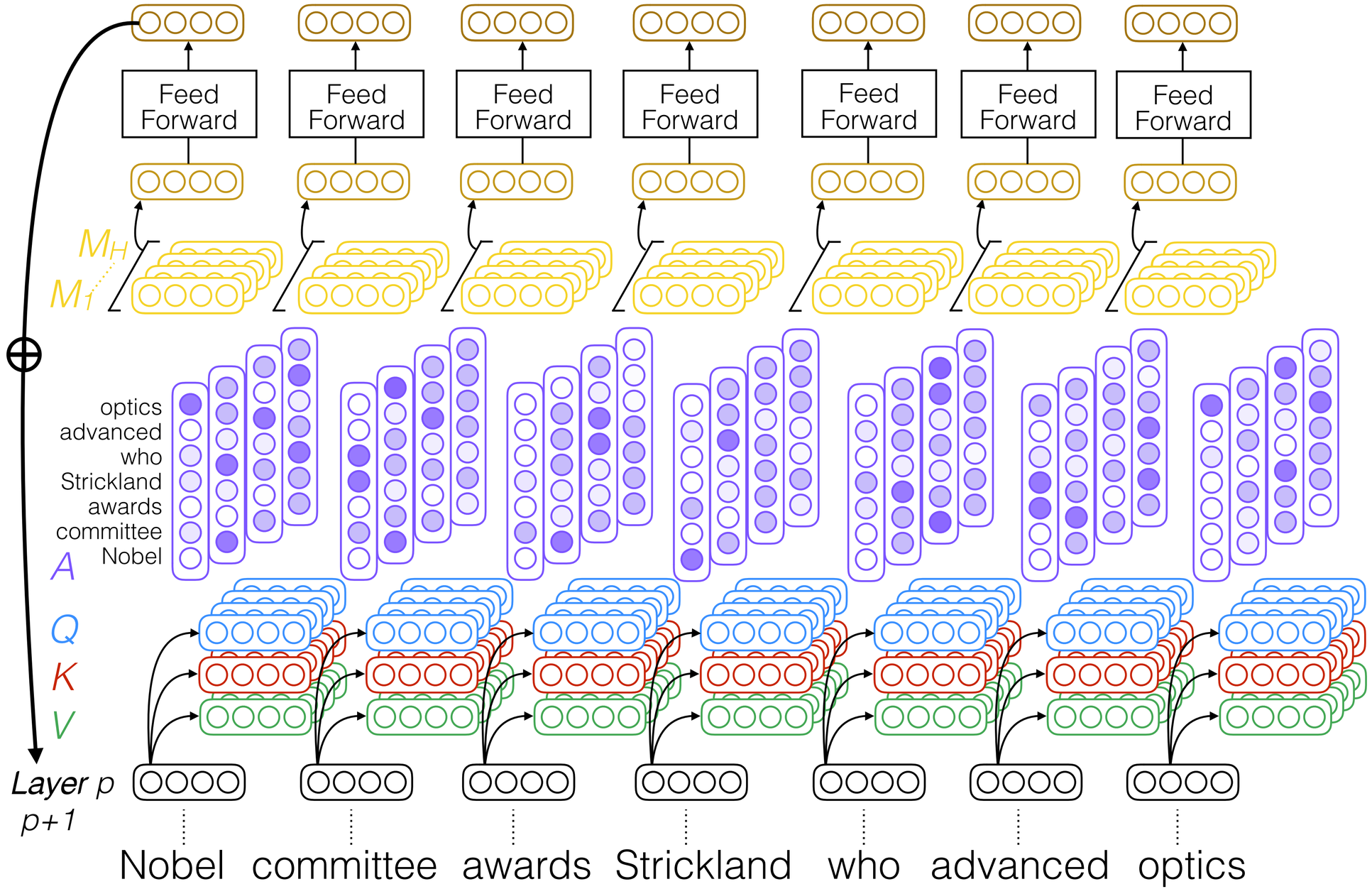
Multi-head self-attention



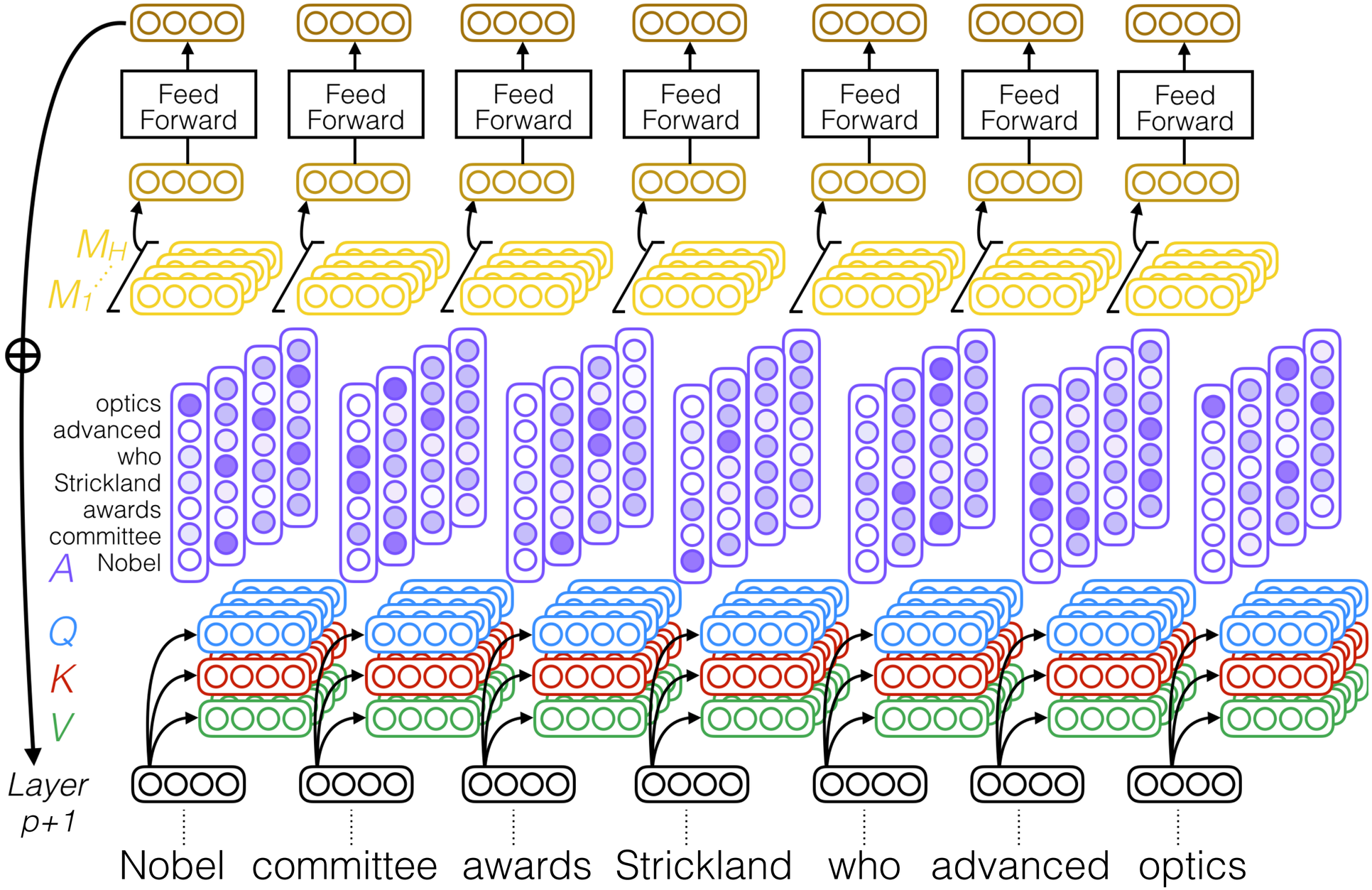
Multi-head self-attention



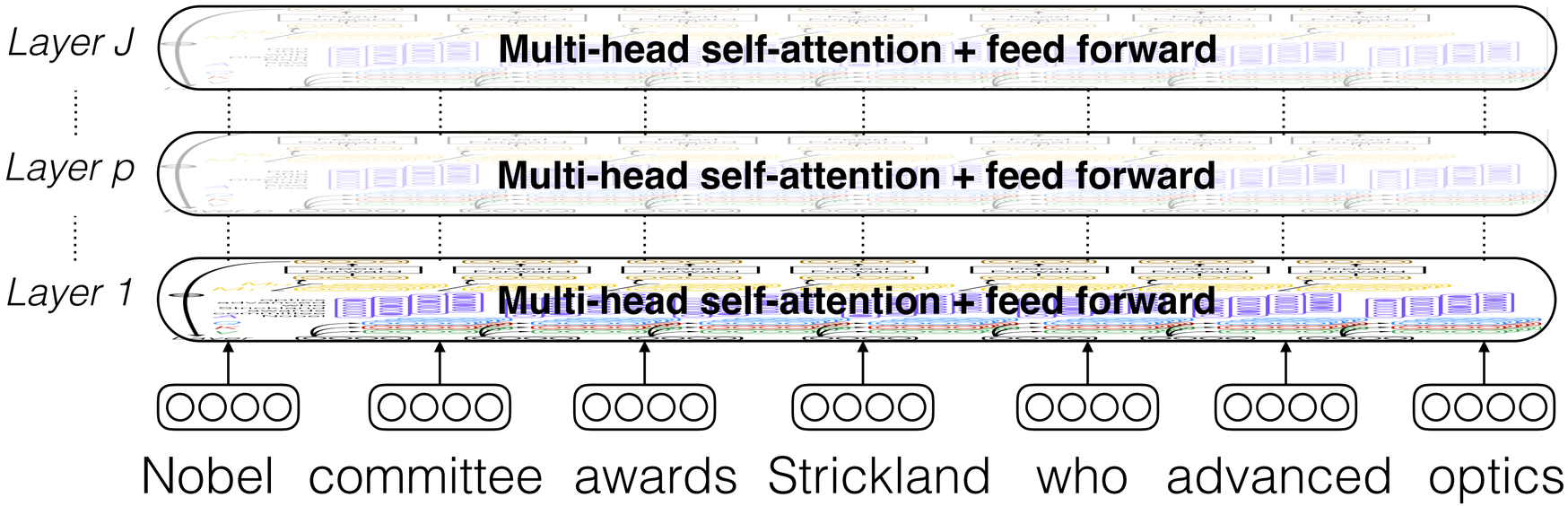
Multi-head self-attention



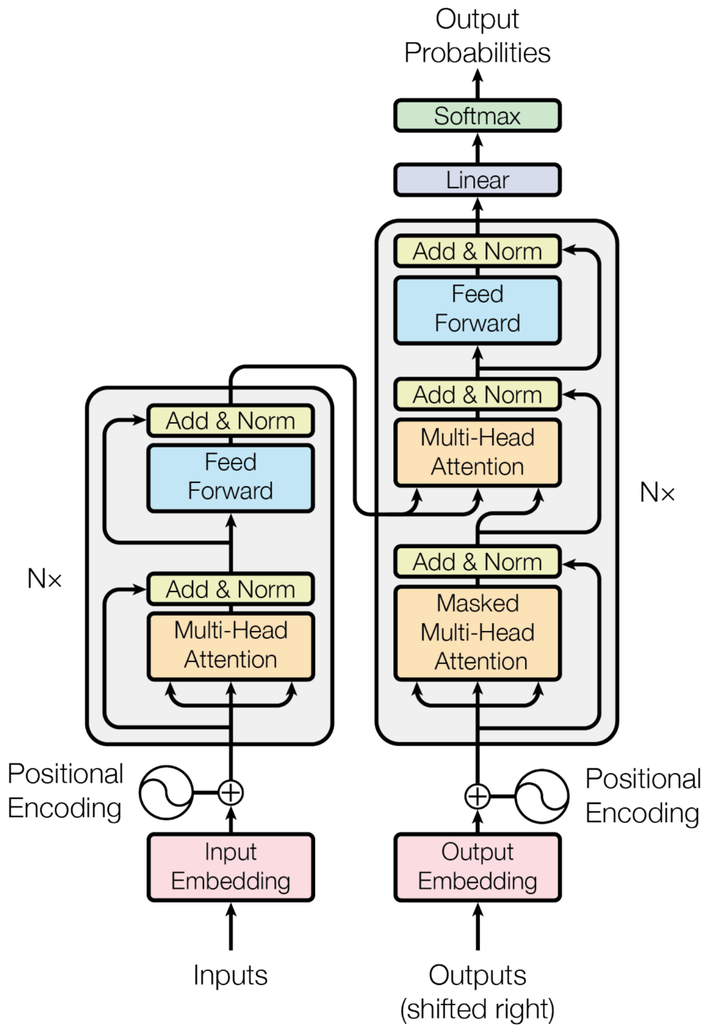
Multi-head self-attention



Multi-head self-attention



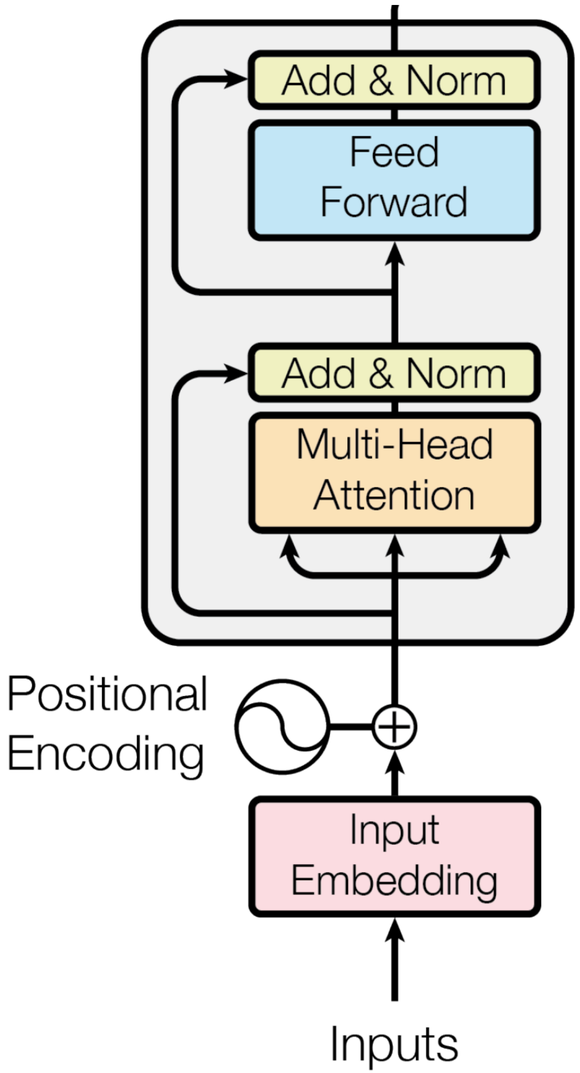
Transformers

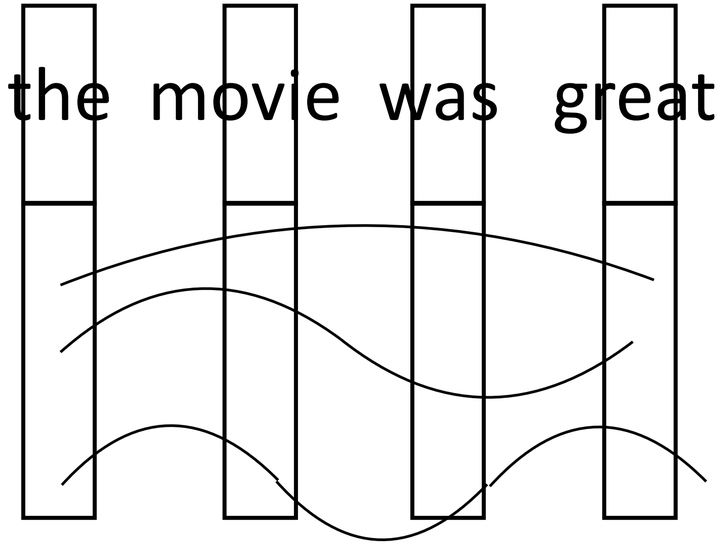


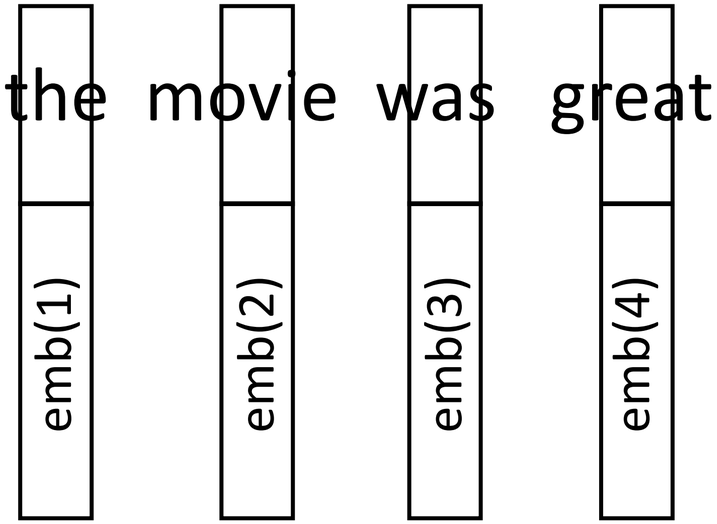
- Encoder and decoder are both transformers
- Decoder consumes the previous generated token (and attends to input), but has *no recurrent state*
- Many other details to get it to work: residual connections, layer normalization, positional encoding, optimizer with learning rate schedule, label smoothing ….

Vaswani et al. (2017)

Transformers



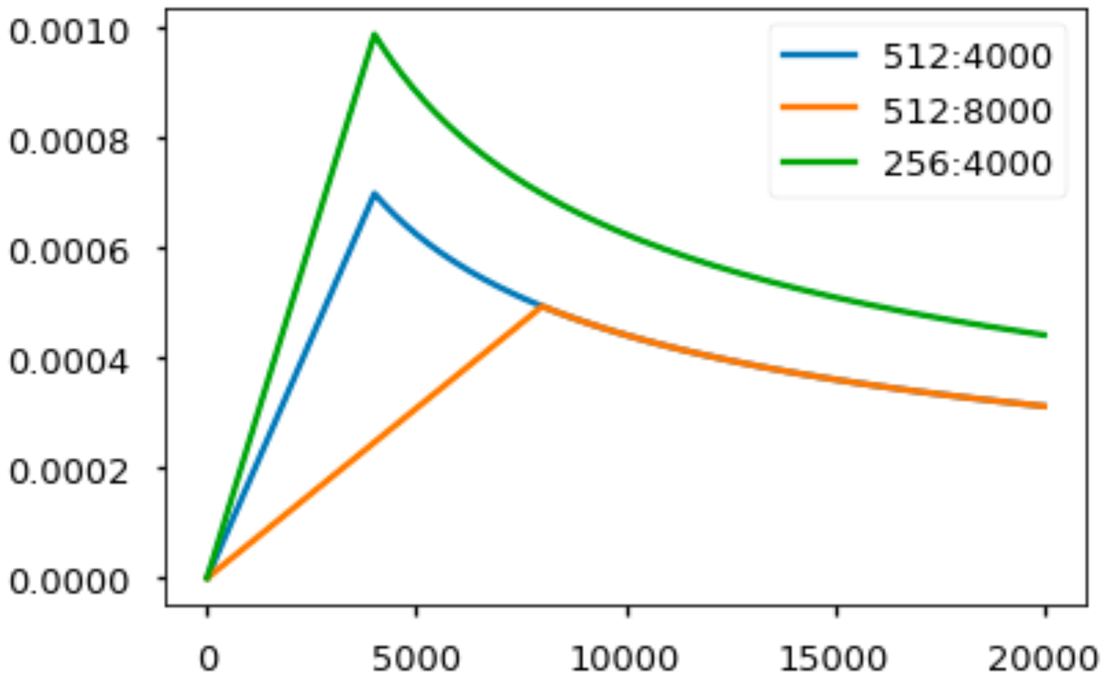




- Augment word embedding with position embeddings, each dim is a sine/cosine wave of a different frequency. Closer points = higher dot products
- Works essentially as well as just encoding position as a one-hot vector

Vaswani et al. (2017)

Transformers



- Adam optimizer with varied learning rate over the course of training
- Linearly increase for warmup, then decay proportionally to the inverse square root of the step number
- This part is very important!

Vaswani et al. (2017)

Label Smoothing

- Instead of using a one-not target distribution, create a distribution that has “confidence” of the correct word and the rest of the “smoothing” mass distributed throughout the vocabulary.
- Implemented by minimizing KL-divergence between smoothed ground truth probabilities and the probabilities computed by model.

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

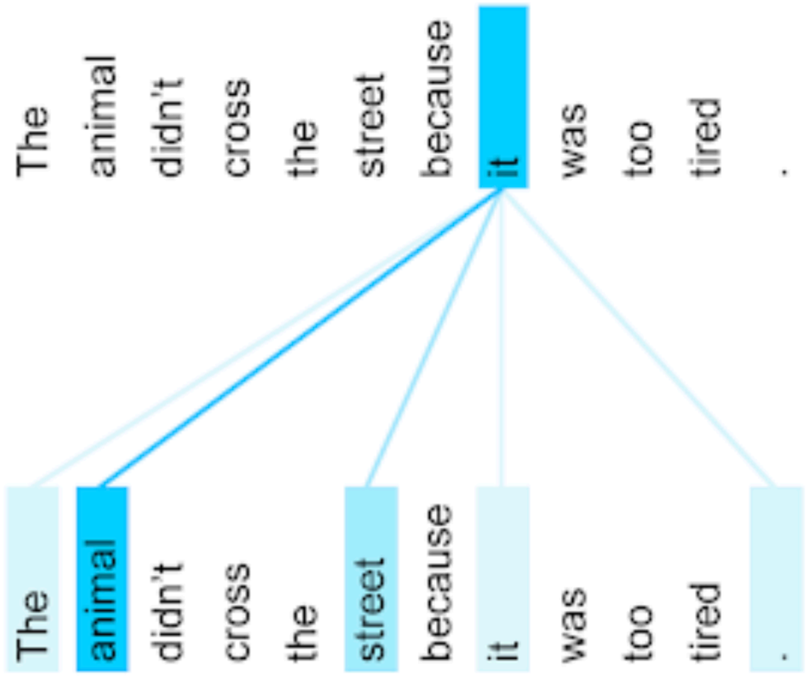
Transformers

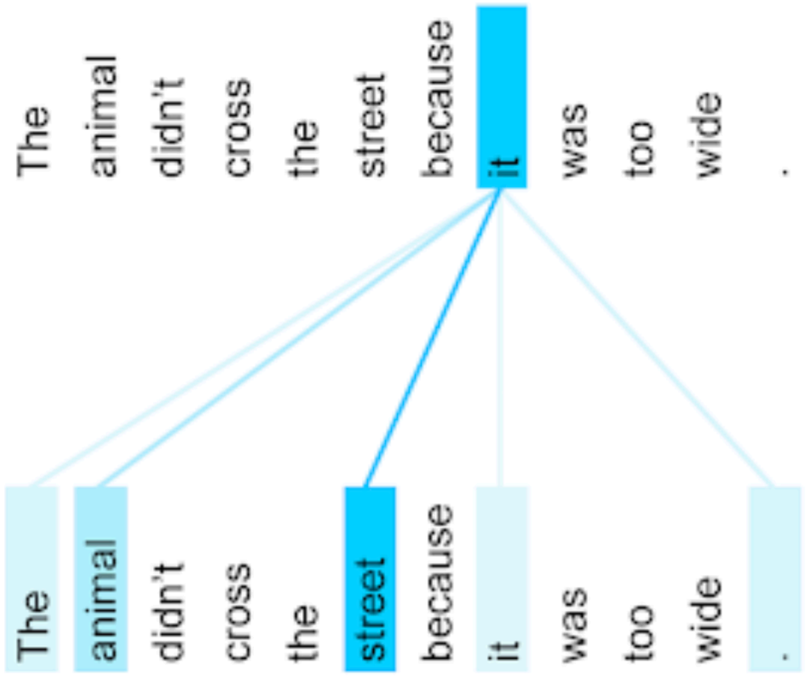
Model	BLEU	
	EN-DE	EN-FR
ByteNet [18]	23.75	
Deep-Att + PosUnk [39]		39.2
GNMT + RL [38]	24.6	39.92
ConvS2S [9]	25.16	40.46
MoE [32]	26.03	40.56
Deep-Att + PosUnk Ensemble [39]		40.4
GNMT + RL Ensemble [38]	26.30	41.16
ConvS2S Ensemble [9]	26.36	**41.29**
Transformer (base model)	27.3	38.1
Transformer (big)	**28.4**	**41.8**

- Big = 6 layers, 1000 dim for each token, 16 heads, base = 6 layers + other params halved

Vaswani et al. (2017)

Visualization





https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Takeaways

- Can build MT systems with LSTM encoder-decoders, CNNs, or transformers
- Word piece / byte pair models are really effective and easy to use
- State of the art systems are getting pretty good, but lots of challenges remain, especially for low-resource settings
- Transformer is a very strong model (when data is large enough); training can be tricky

Text Simplification

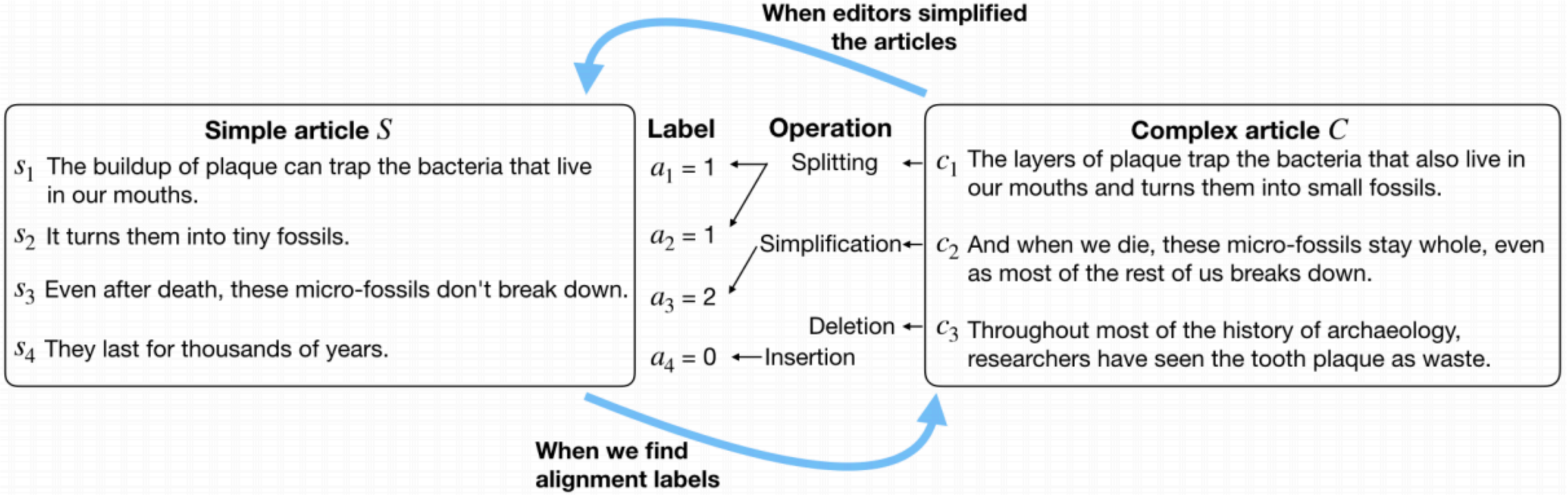


Figure 1: An example of sentence alignment between an original news article (right) and its simplified version (left) in Newsela. The label a_i for each simple sentence s_i is the index of complex sentence c_{a_i} it aligned to.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, Wei Xu. "Neural CRF Model for Sentence Alignment in Text Simplification" in ACL (2020)

Text Simplification

	Evaluation on our new test set						Evaluation on old test set					
	SARI	**add**	**keep**	**del**	**FK**	**Len**	**SARI**	**add**	**keep**	**del**	**FK**	**Len**
Complex (input)	11.9	0.0	35.5	0.0	12	24.3	12.5	0.0	37.7	0.0	11	22.9
Models trained on old dataset (original NEWSELA corpus released in (Xu et al., 2015))												
Transformer$_{rand}$	33.1	1.8	22.1	75.4	**6.8**	14.2	34.1	2.0	25.5	**74.8**	6.7	14.1
LSTM	35.6	2.8	**32.1**	72.1	8.0	16.3	36.2	2.5	**34.9**	71.3	7.6	16.1
EditNTS	35.4	1.8	30.0	75.4	7.1	14.1	36.2	1.7	32.8	73.8	7.0	14.1
Transformer$_{bert}$	34.4	2.4	25.1	**75.8**	7.0	14.5	35.1	2.7	27.8	**74.8**	6.8	14.3
Models trained on our new dataset (NEWSELA-AUTO)												
Transformer$_{rand}$	35.6	3.2	28.4	74.9	7.1	14.3	35.2	2.5	29.8	73.5	7.0	14.1
LSTM	35.8	3.9	30.5	73.1	6.9	14.2	36.4	3.3	33.0	72.9	6.6	13.9
EditNTS	35.8	2.4	29.3	75.6	6.3	11.6	35.7	1.8	31.1	74.2	**6.0**	11.5
Transformer$_{bert}$	**36.6**	**4.5**	31.0	74.3	**6.8**	**13.3**	**36.8**	**3.8**	33.1	73.4	6.8	**13.5**

94k sent. pairs

394k sent. pairs

Table 5: Automatic evaluation results on NEWSELA test sets comparing models trained on our new dataset NEWSELA-AUTO against the existing dataset (Xu et al., 2015). We report **SARI, the main automatic metric** for simplification, precision for deletion and F1 scores for adding and keeping operations. We also show Flesch-Kincaid (FK) grade level readability, and average sentence length (Len). Add scores are low partially because we are using one reference. **Bold** typeface and underline denote the best and the second best performances respectively. For FK and Len, we consider the values closest to reference as the best.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, Wei Xu. "Neural CRF Model for Sentence Alignment in Text Simplification" in ACL (2020)