Efficient / long-range Transformers

CS 685, Fall 2021
Advanced Natural Language Processing

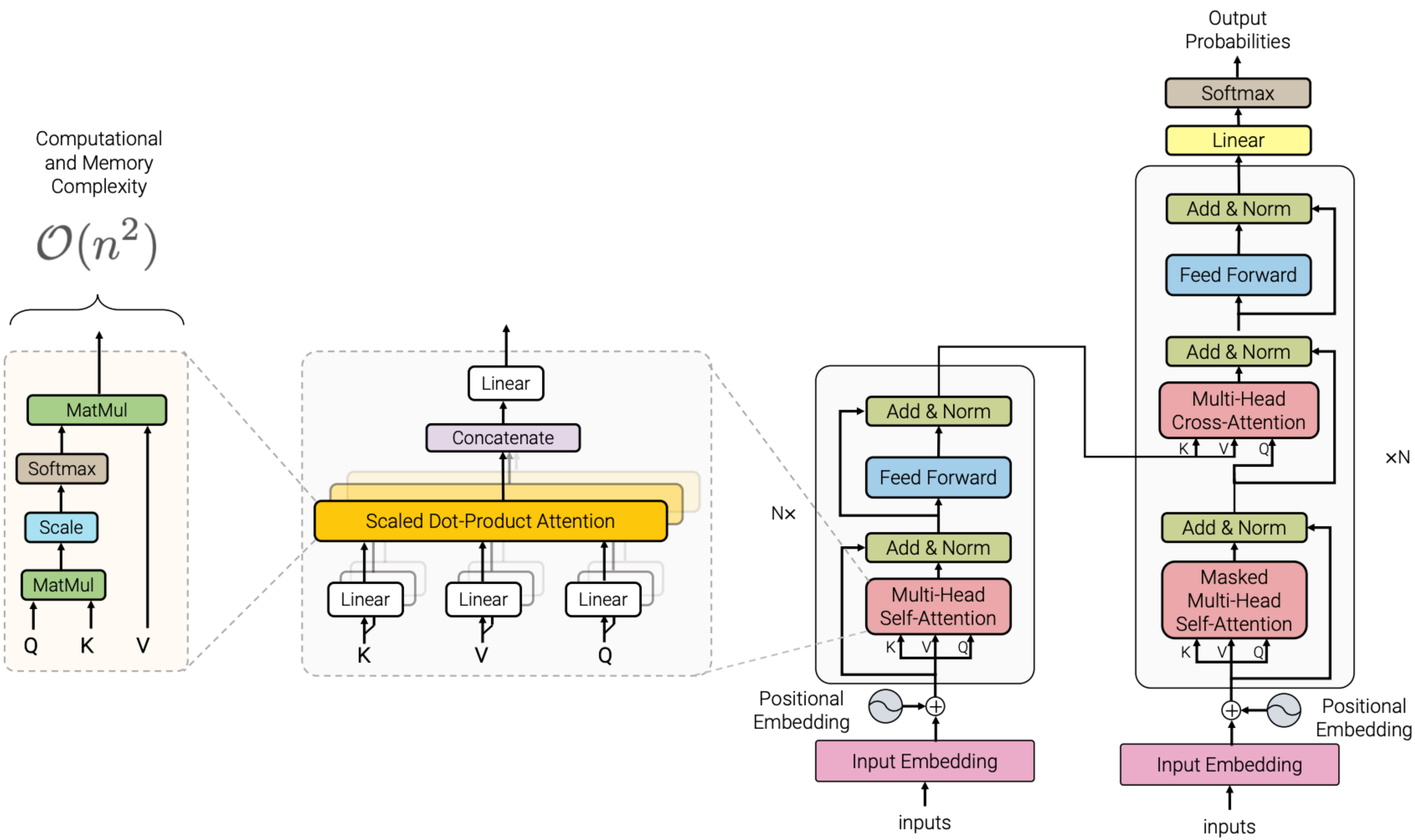
Mohit Iyyer
College of Information and Computer Sciences
University of Massachusetts Amherst

Some slides from Simeng Sun

Stuff from last time

- Quiz 6 due Friday
- Extra credit released, due 12/16
- HW 1 due 11/5
- Midterm coming up on 11/9!
    - Solutions to practice midterm will be posted next week

Quadratic complexity of self-attention



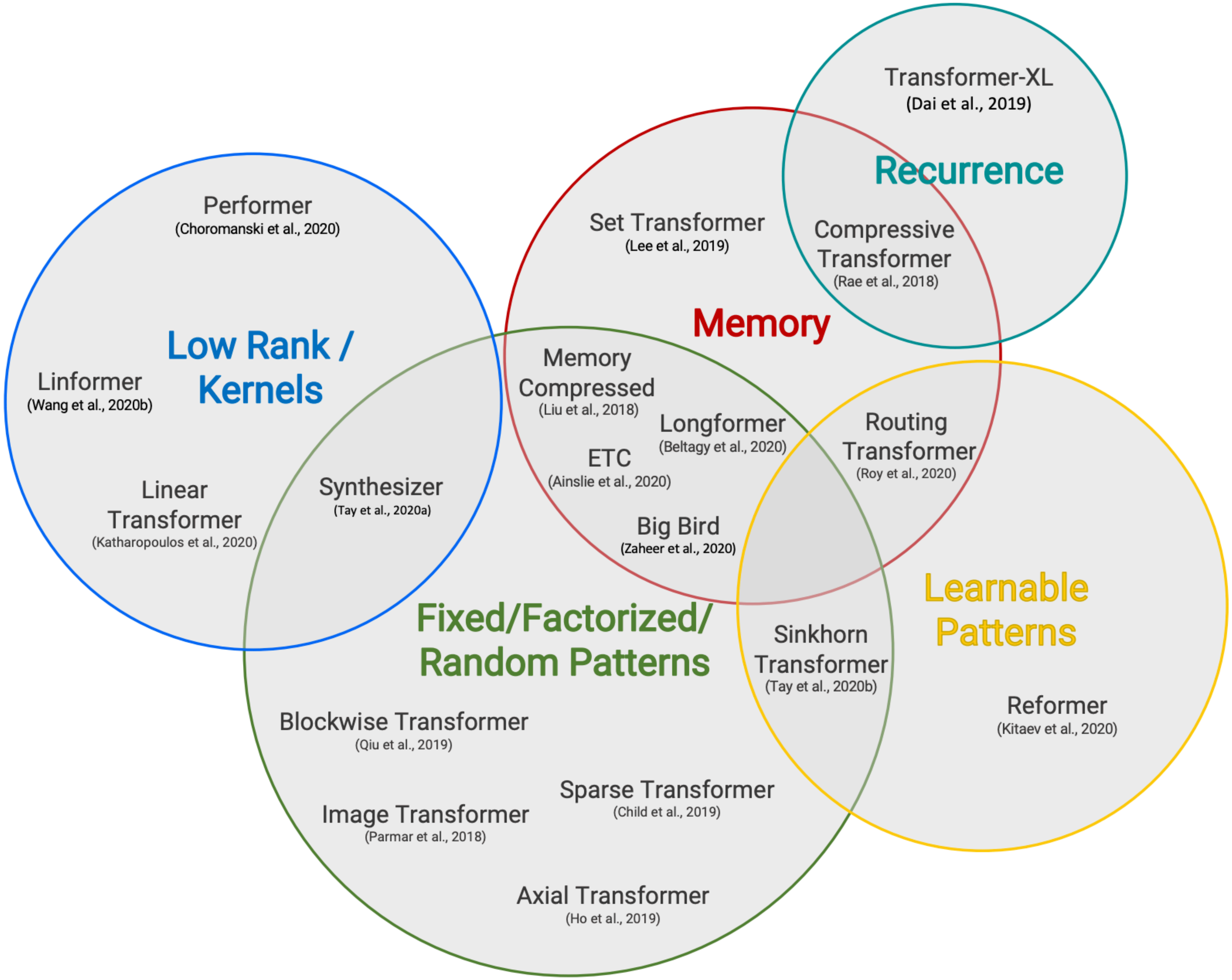
Tay et al., 2020 survey on efficient Transformers

Transformer LMs are slow…

	100 context tokens	200 context tokens	300 context tokens
Generate next 100 tokens	2.3s	2.7s	3.1s
Generate next 200 tokens	4.4s	4.6s	5.7s
Generate next 300 tokens	7.3s	8.4	9.7s

GPT-2 medium timed on a single 2080Ti

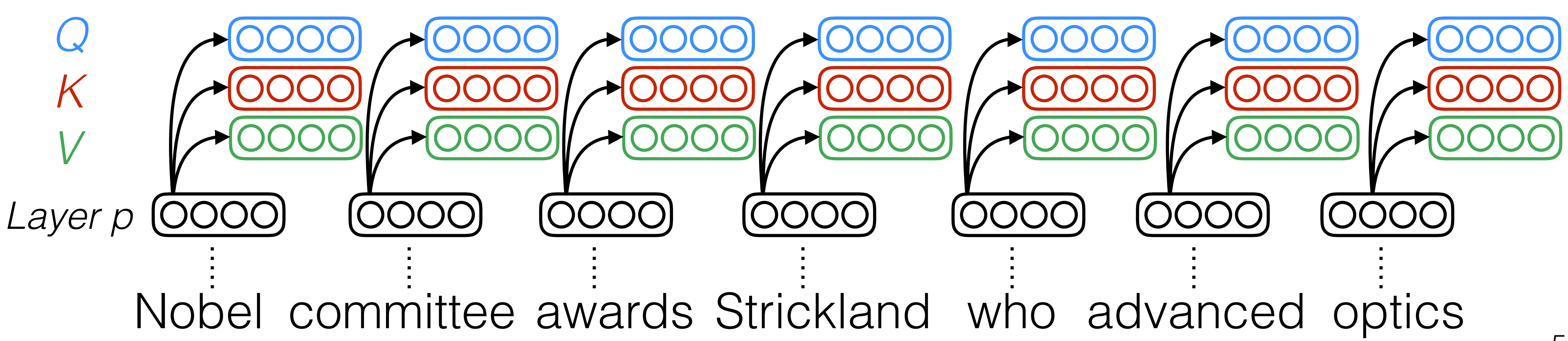
Many attempts to design more efficient self-attention



Tay et al., 2020 survey on efficient Transformers

these awesome slides are from Emma Strubell

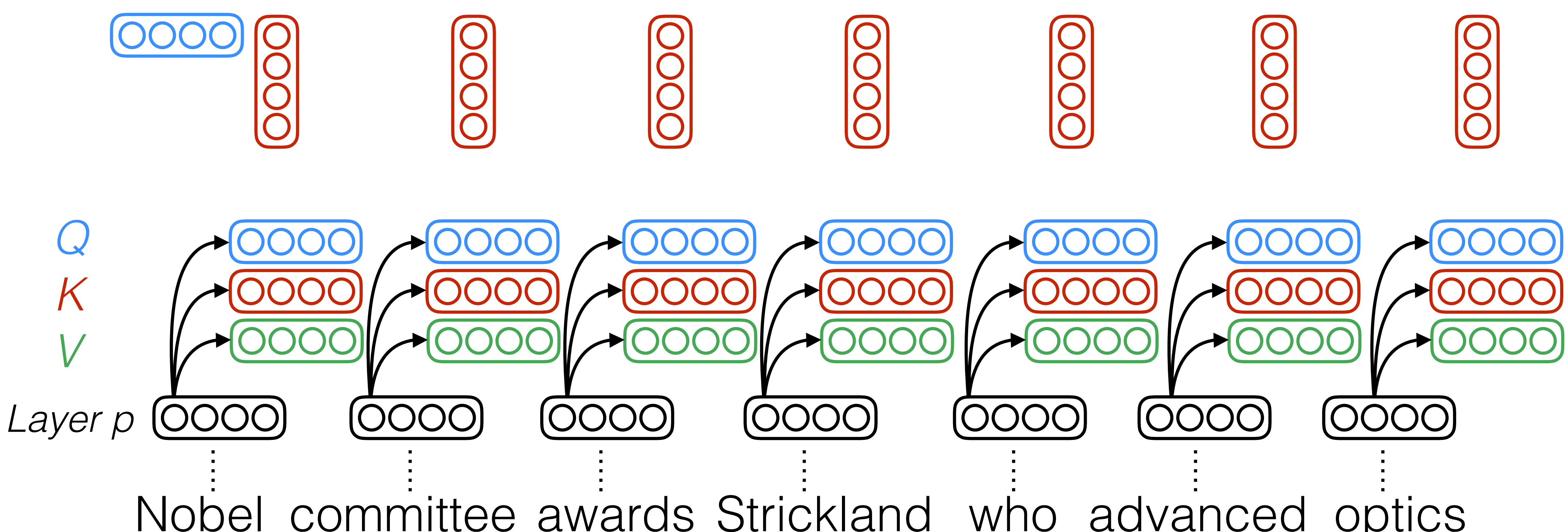
Self-attention



these awesome slides are from Emma Strubell

Self-attention

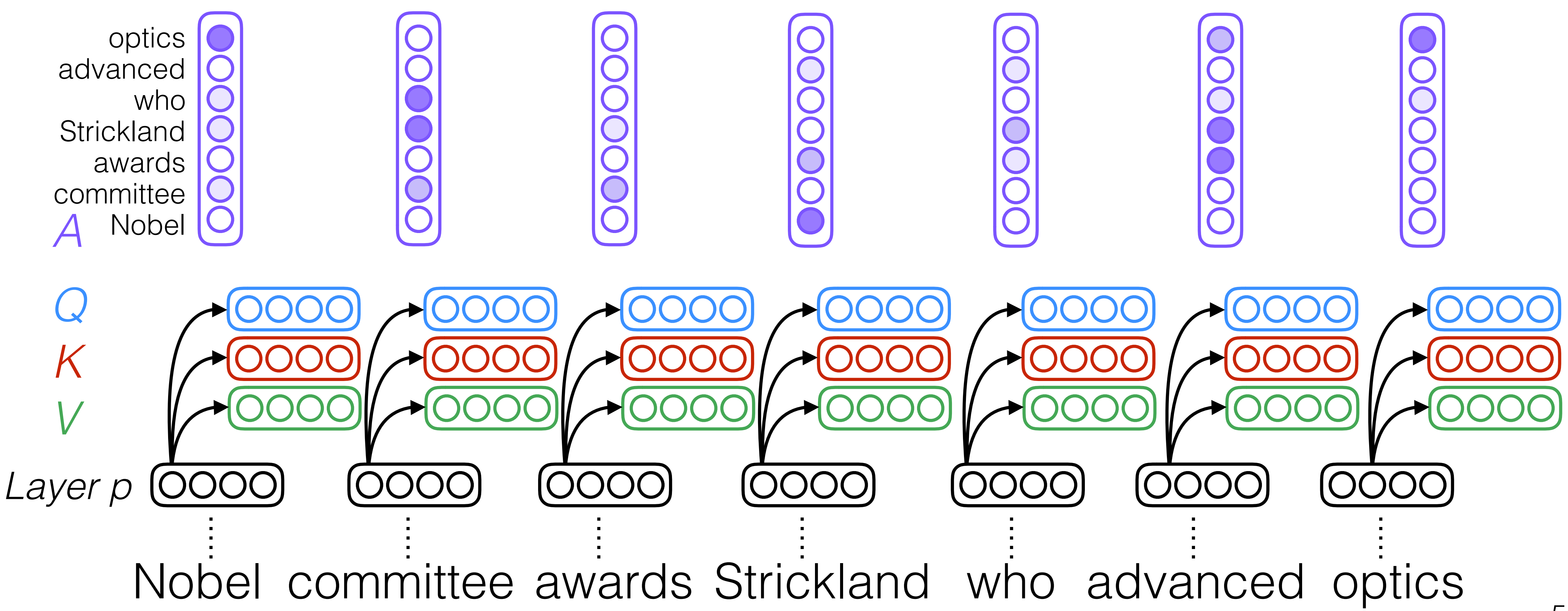
[Vaswani et al. 2017]



these awesome slides are from Emma Strubell

Self-attention

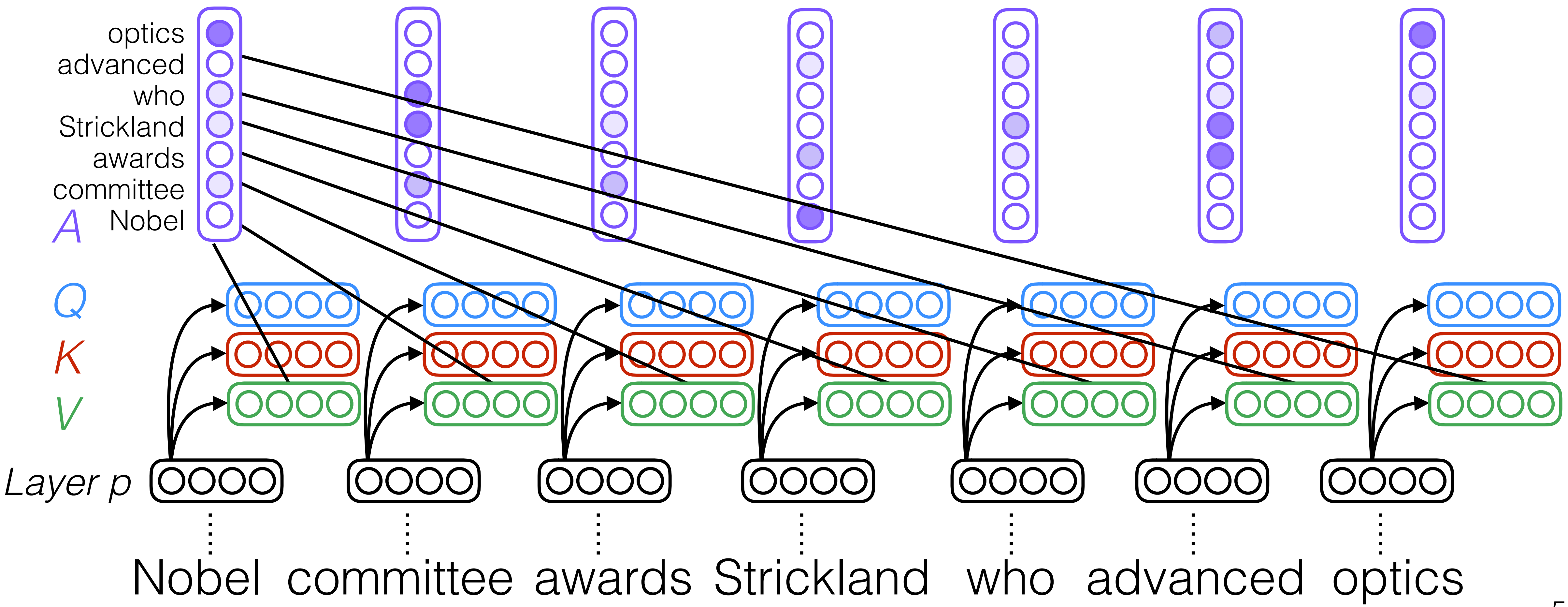
[Vaswani et al. 2017]



these awesome slides are from Emma Strubell

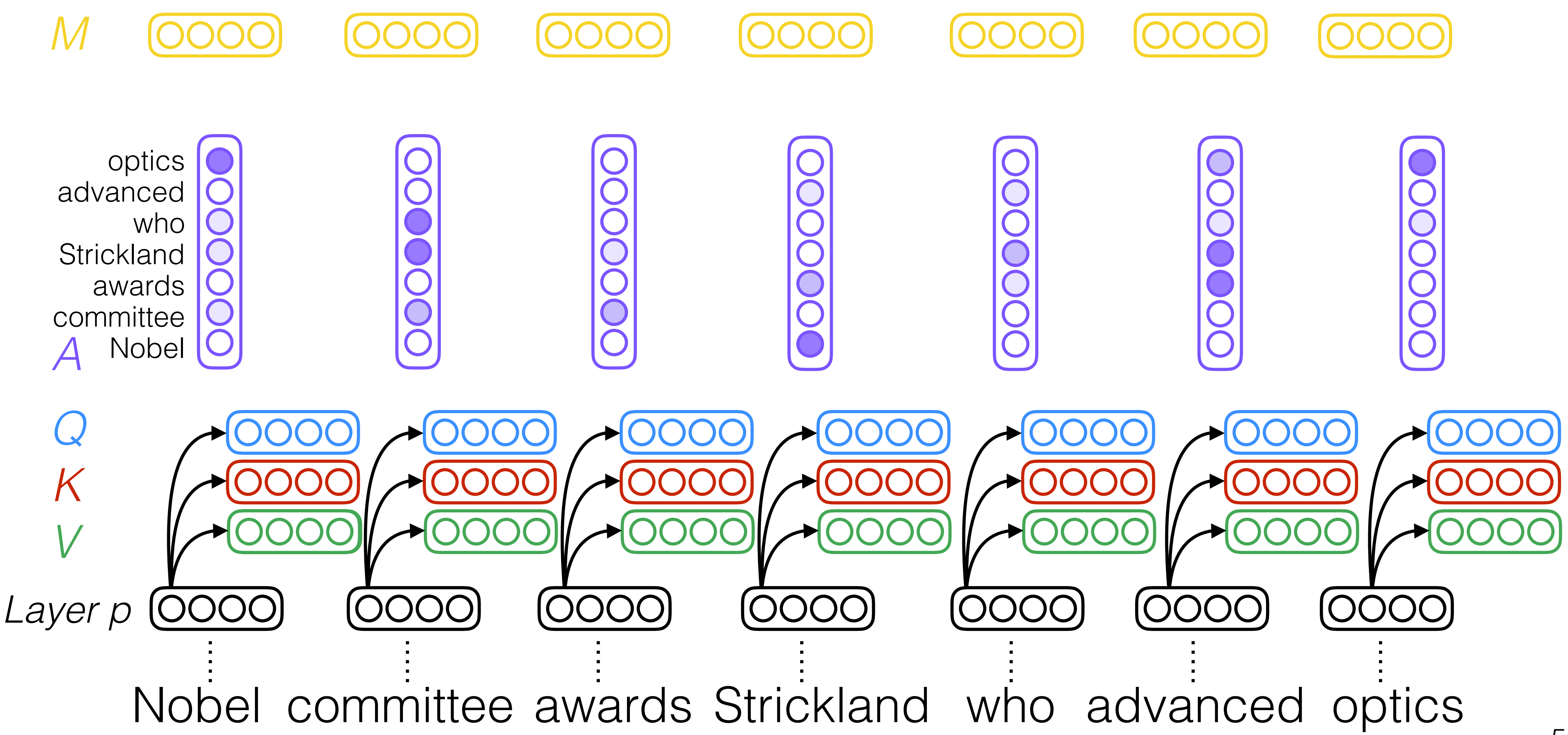
Self-attention

[Vaswani et al. 2017]



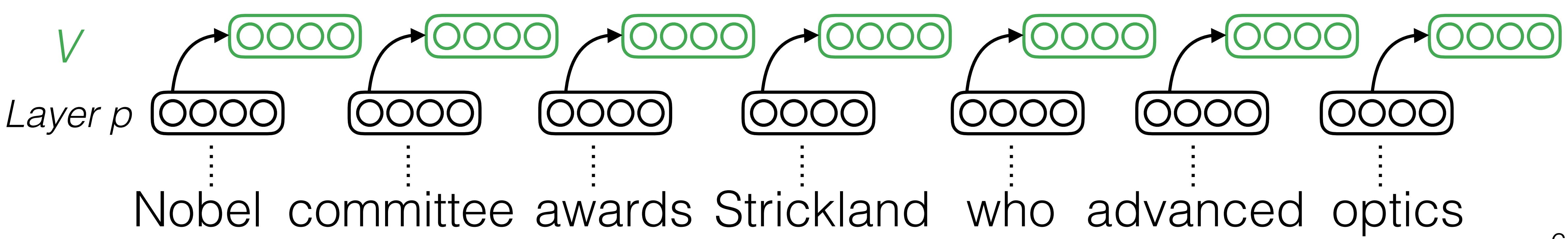
Self-attention

[Vaswani et al. 2017]

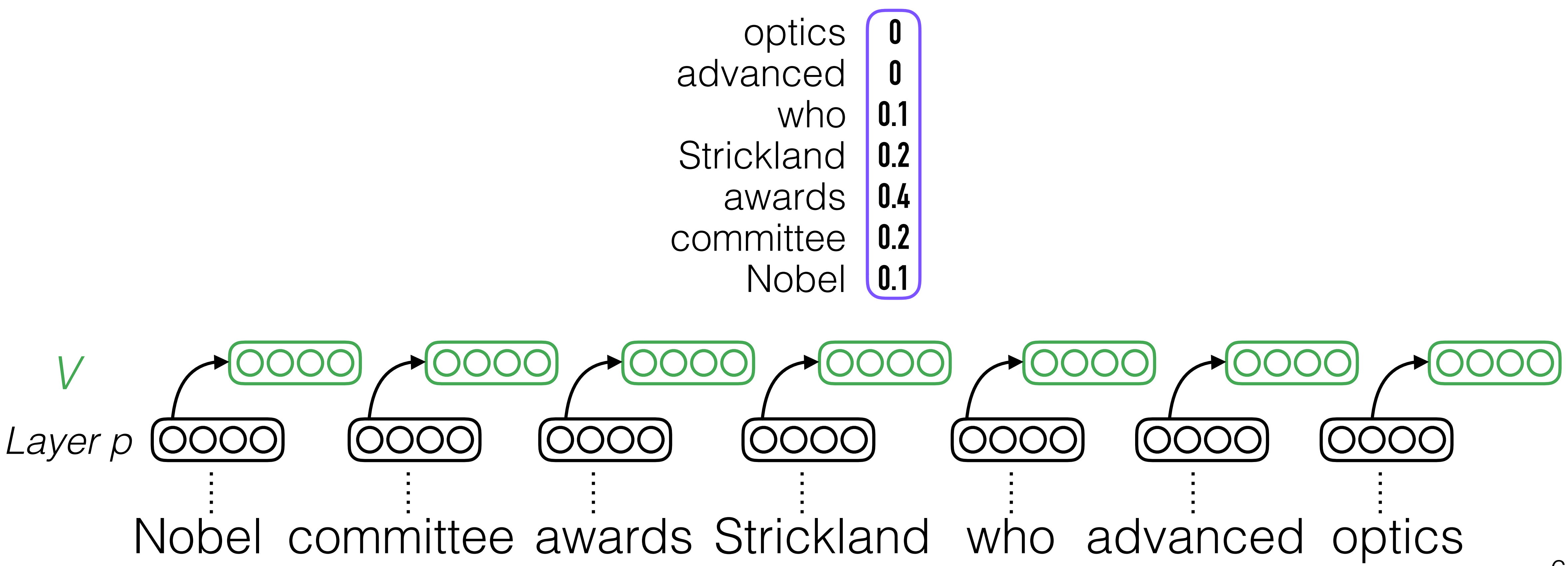


what if instead of learning the attention scores, we simply hard-code them?

Hardcoded self-attention

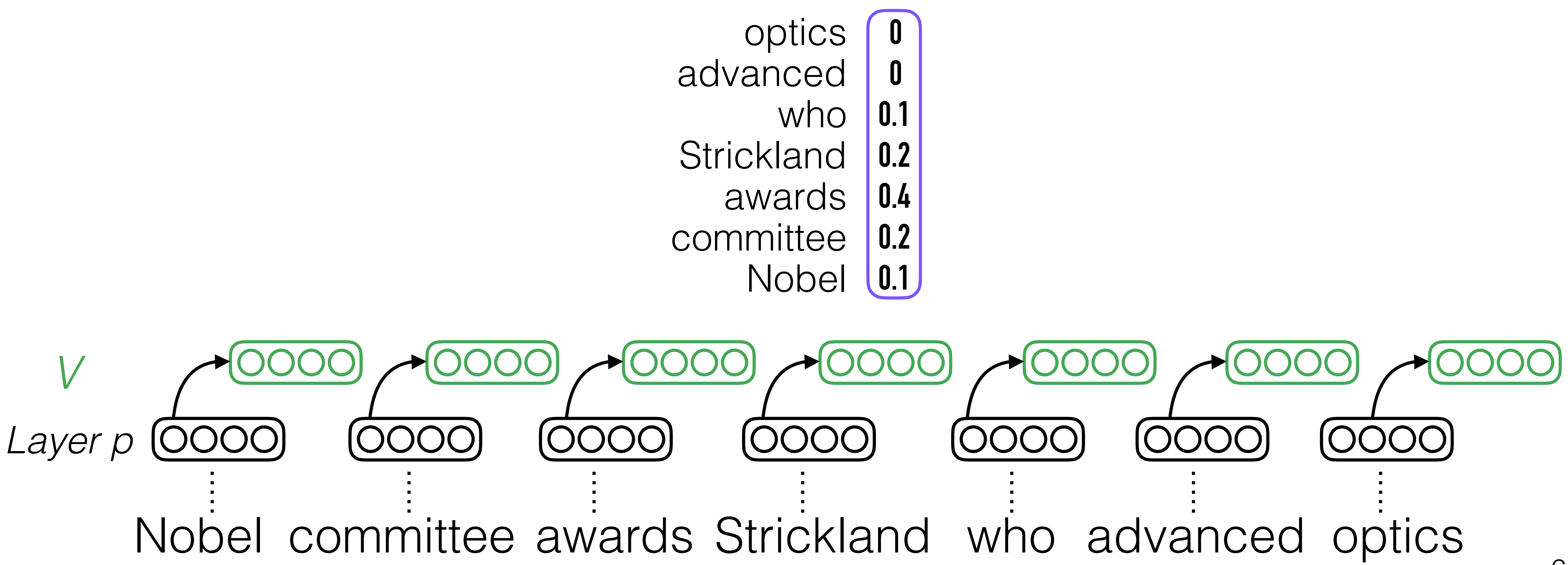


Hardcoded self-attention



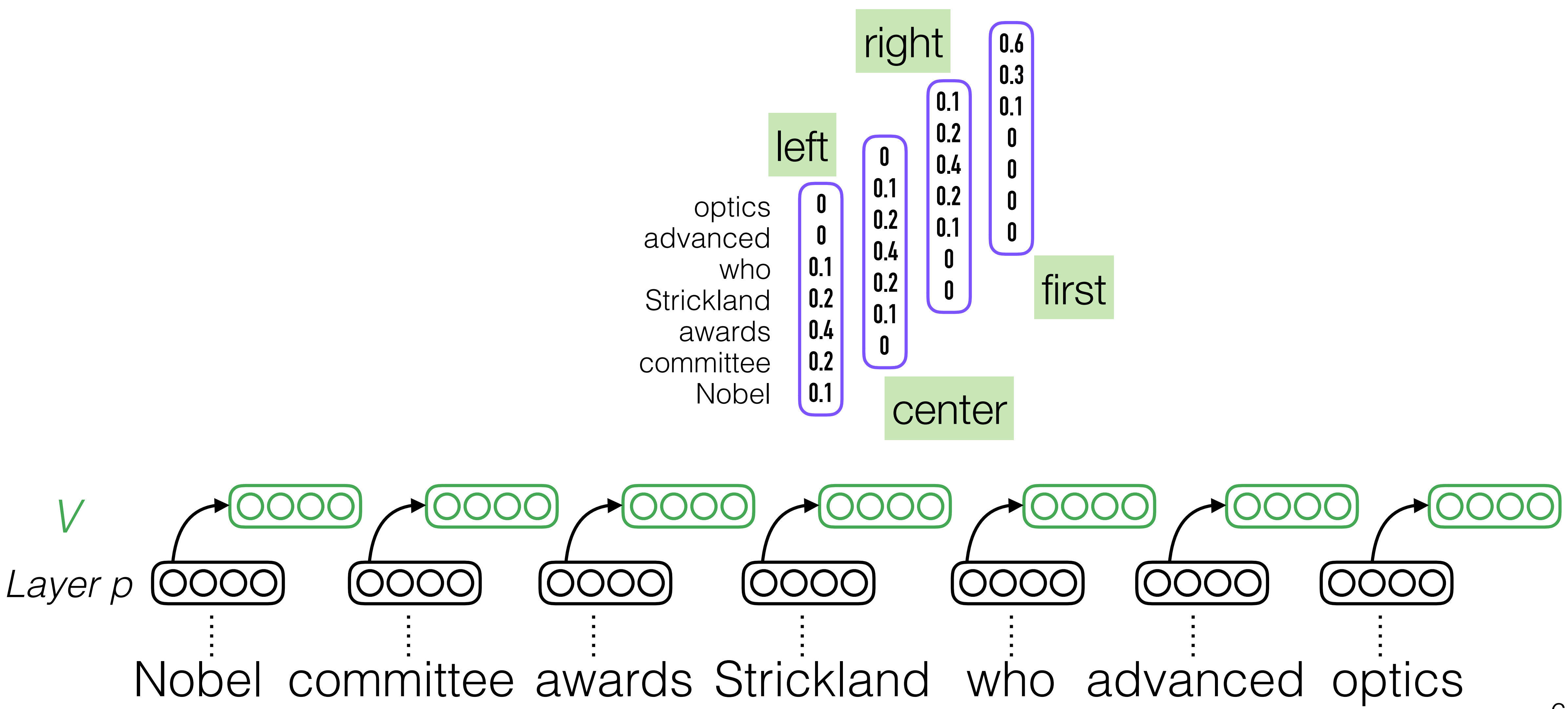
Hardcoded self-attention

These values are from a standard normal distribution centered on the word to the *left* of the current word, “Strickland”



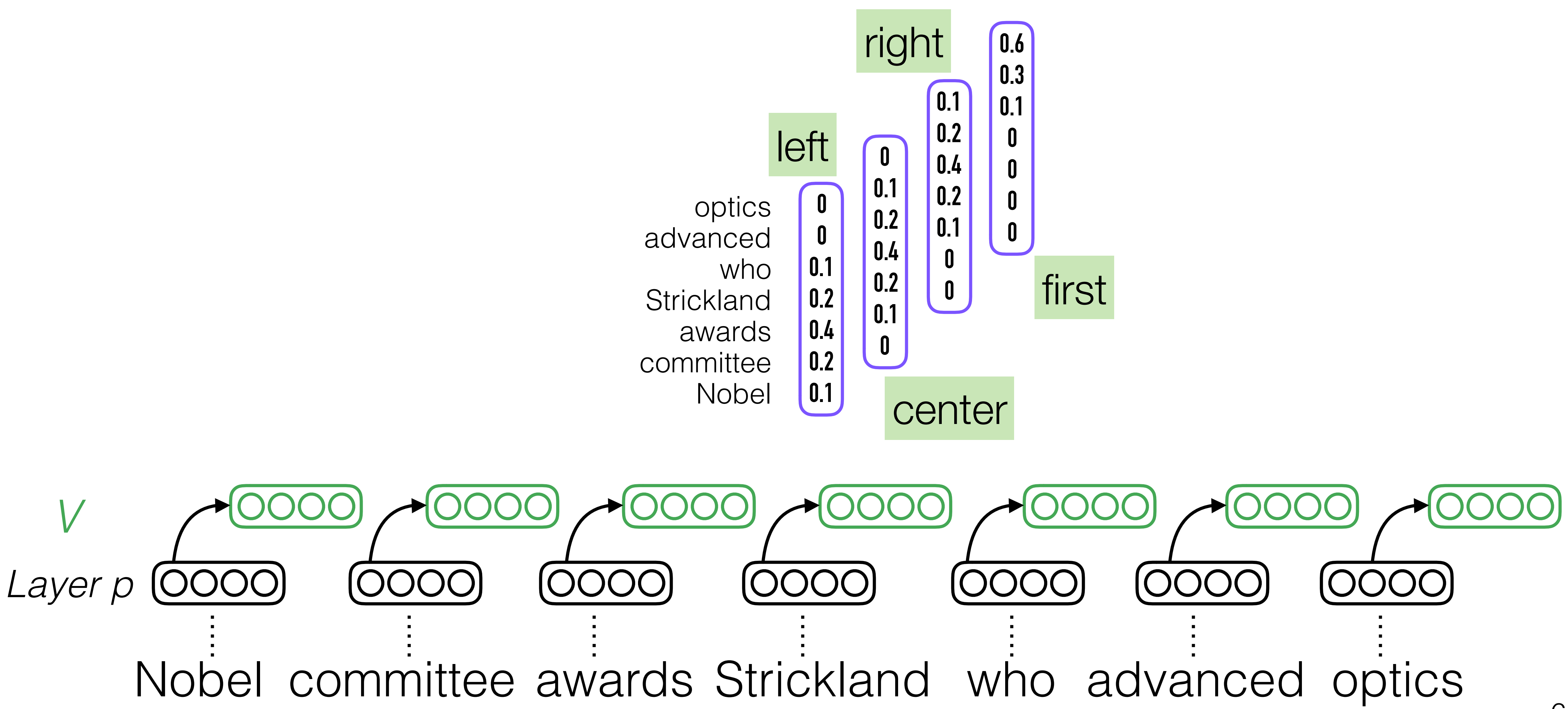
Hardcoded self-attention

We can use distributions centered at different locations to hard-code multiple heads

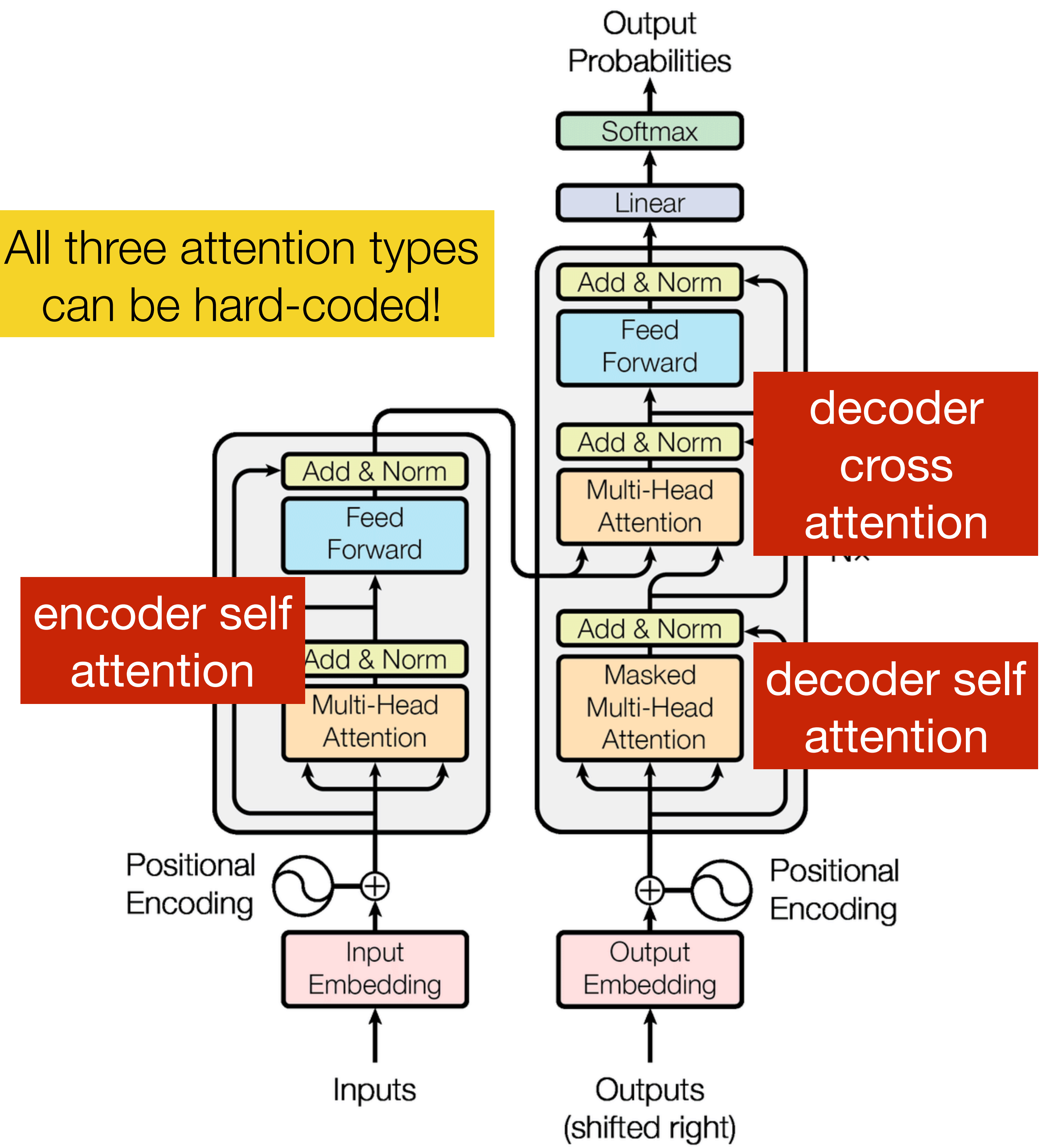


Hardcoded self-attention

This is equivalent to a 1-d convolution with a fixed kernel



All three attention types can be hard-coded!



Removing self-attention in Transformers for machine translation

(ACL 2020)

Weiqiu You

Simeng Sun

how much do you need self-attention?

dataset	all heads learned
IWSLT16 en-de	30.0
IWSLT16 de-en	34.4
WMT16 en-ro	33.0
WMT16 ro-en	33.1
WMT14 en-de	26.8
WMT14 en-fr	40.3

how much do you need self-attention?

dataset	all heads learned	hard-code self attn
IWSLT16 en-de	30.0	30.3
IWSLT16 de-en	34.4	34.8
WMT16 en-ro	33.0	32.4
WMT16 ro-en	33.1	32.8
WMT14 en-de	26.8	26.3
WMT14 en-fr	40.3	39.1

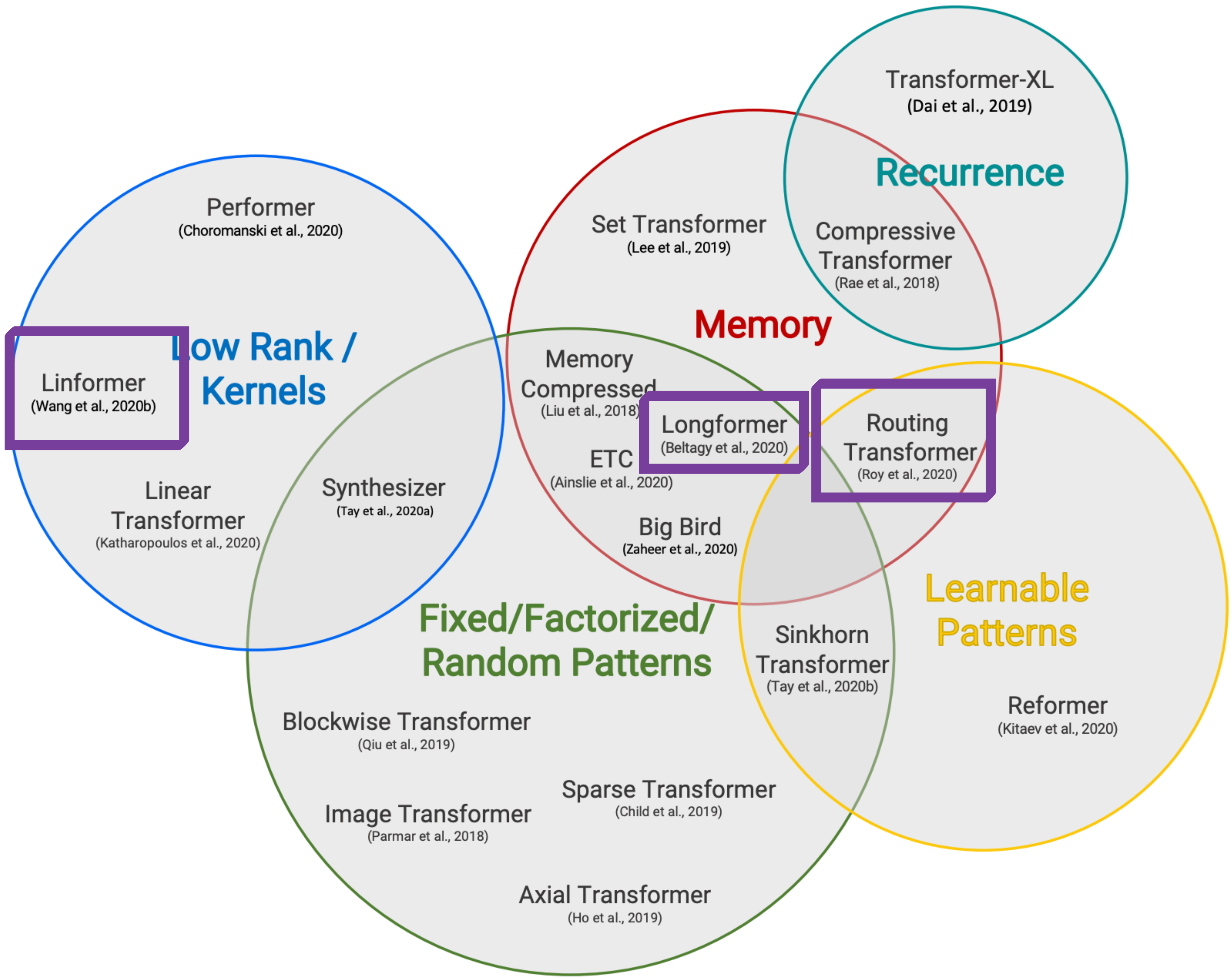
hard-coding self-attention in the encoder and decoder does not significantly impact BLEU!

efficiency improvements

- Batch size increase: hard-coding self-attention and reducing cross attention heads allows us to use **27%** more tokens per batch
- Decoding speed (sentences/sec) increases by **30.2%** over the baseline Transformer

Moral of the story: if latency is important for your task, consider drastic simplifications to your neural models; oftentimes, you can make them faster without significant performance loss

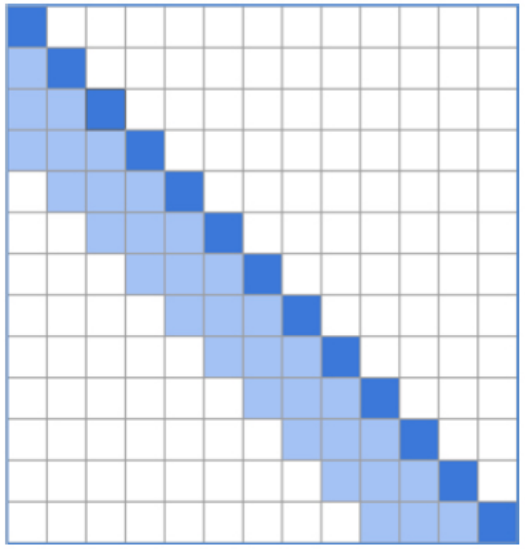
Many attempts to design more efficient self-attention



Tay et al., 2020 survey on efficient Transformers

Local Attention

- Position-based sparse attention

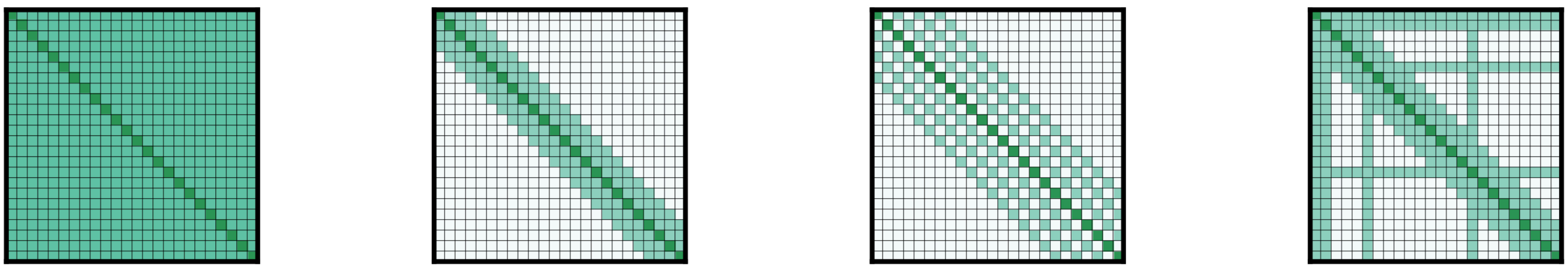


Each token only attends to previous K tokens

Pic screenshot from: https://arxiv.org/pdf/2003.05997.pdf

Longformer

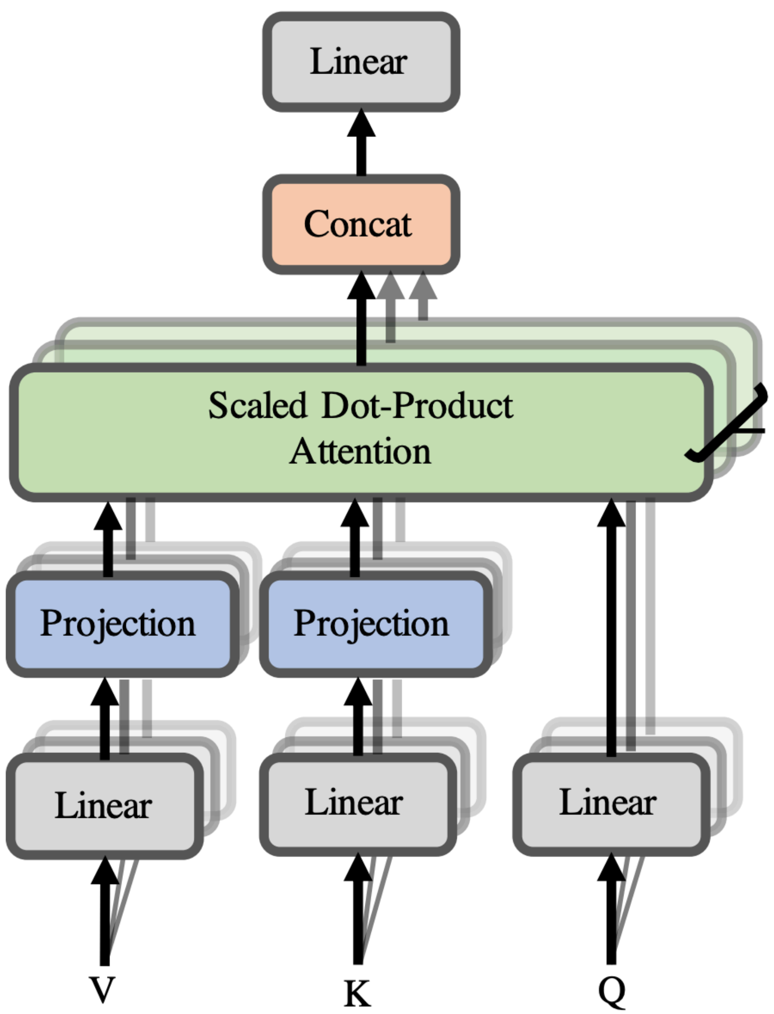
- Try other types of fixed attention patterns

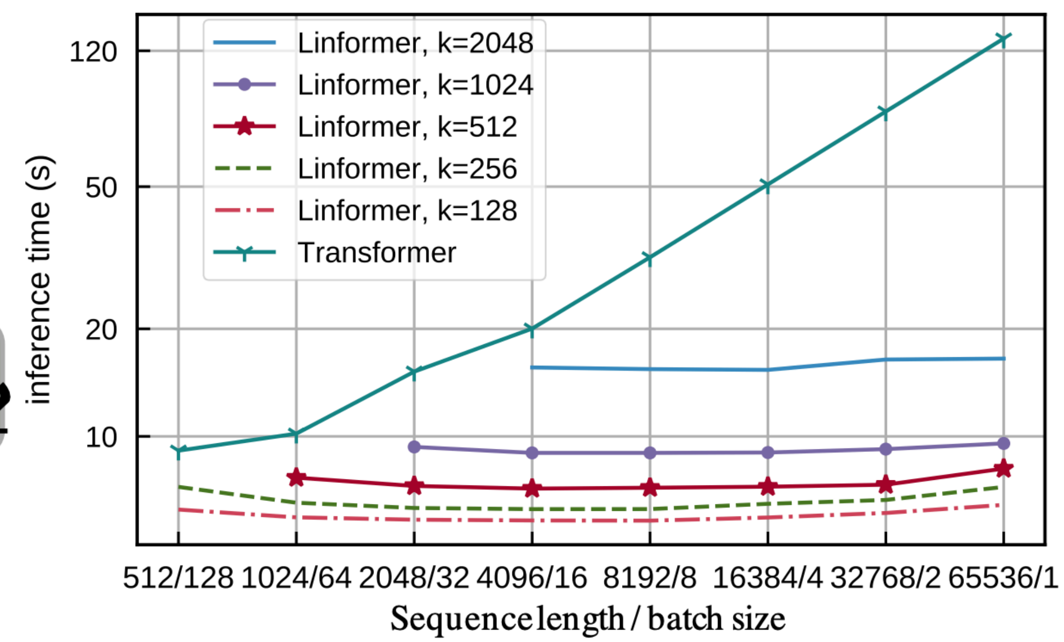


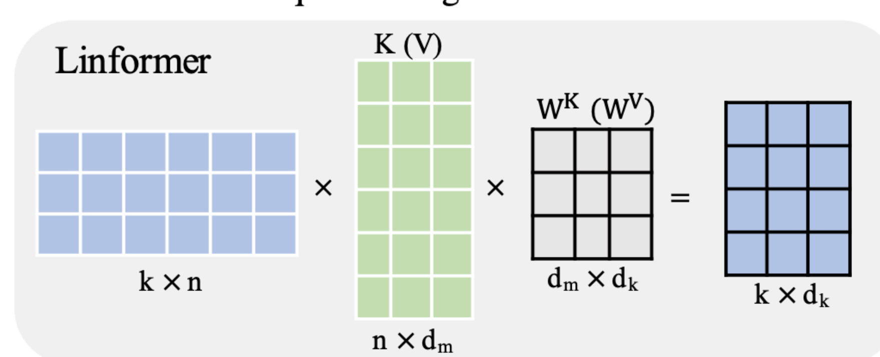
(a) Full n^2 attention (b) Sliding window attention (c) Dilated sliding window (d) Global+sliding window

Linformer

- Main idea: produce a low-rank approximation of the **N** x **N** attention matrix by simply projecting the **N** x **d** keys and values to **N** x **k** where **k** <<<<< **N**

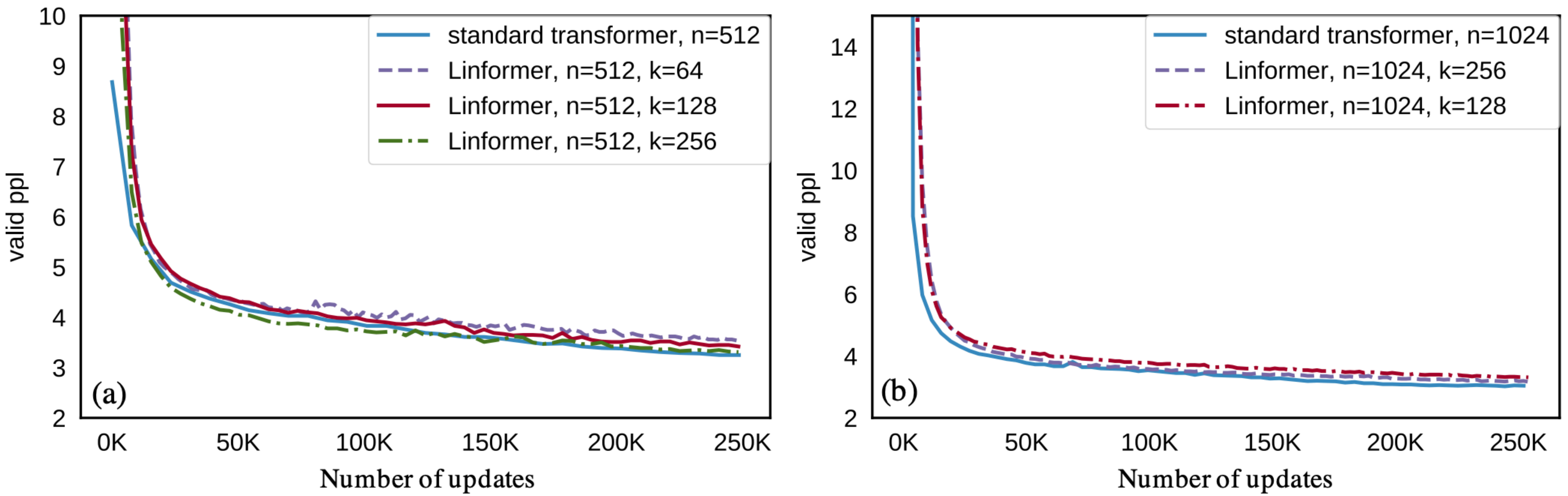






Linformer

- Despite the approximation, not much effect on LM perplexity



Linformer

- Also effective in masked LM settings (e.g., BERT, RoBERTa)

n	Model	SST-2	IMDB	QNLI	QQP	Average
512	Liu et al. (2019), RoBERTa-base	93.1	94.1	90.9	**90.9**	92.25
	Linformer, 128	92.4	94.0	90.4	90.2	91.75
	Linformer, 128, shared kv	**93.4**	93.4	90.3	90.3	91.85
	Linformer, 128, shared kv, layer	93.2	93.8	90.1	90.2	91.83
	Linformer, 256	93.2	94.0	90.6	90.5	92.08
	Linformer, 256, shared kv	93.3	93.6	90.6	90.6	92.03
	Linformer, 256, shared kv, layer	93.1	94.1	**91.2**	90.8	**92.30**
512	Devlin et al. (2019), BERT-base	92.7	93.5	91.8	89.6	91.90
	Sanh et al. (2019), Distilled BERT	91.3	92.8	89.2	88.5	90.45
1024	Linformer, 256	93.0	93.8	90.4	90.4	91.90
	Linformer, 256, shared kv	93.0	93.6	90.3	90.4	91.83
	Linformer, 256, shared kv, layer	93.2	**94.2**	90.8	90.5	92.18

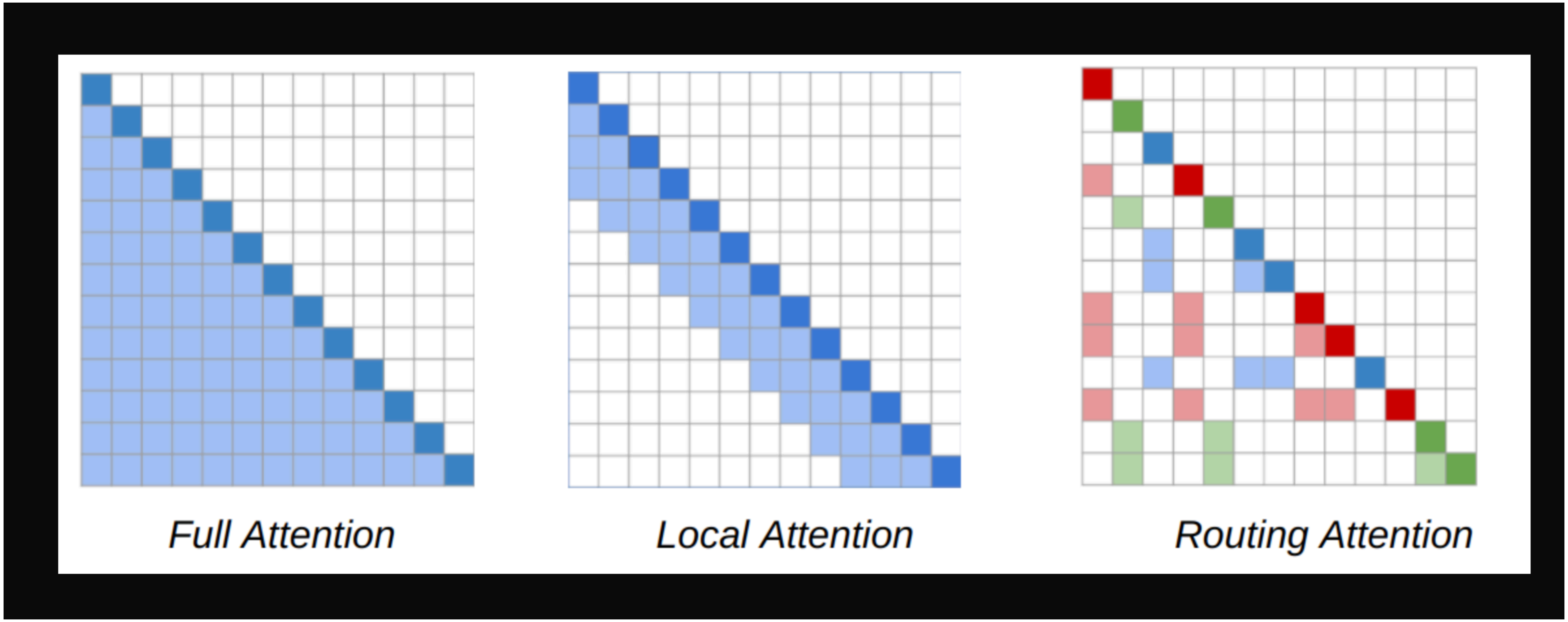
Let’s talk about length

System	Max Input Len.
GPT-2 (Radford et al., 2019)	1024
Adaptive Input (Baevski and Auli, 2019)	3072
BigBird (Zaheer et al., 2020)	4096
LongFormer (Beltagy and Peters and Cohan, 2020)	4098
Routing Transformer (Roy et al., 2021)	8192

LM benchmark

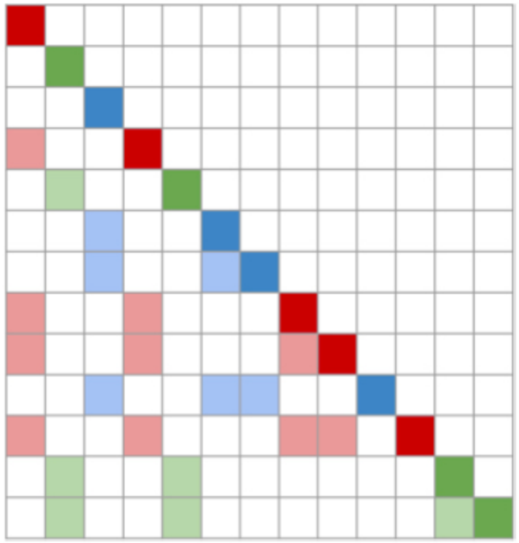
Dataset	Type	Avg. Article Len.
Penn Treebank	News articles	355
WikiText-103	Wikipedia articles	3.6K
PG-19	Books	69K

What about *content-based* sparsity?



Routing Transformer (RT)

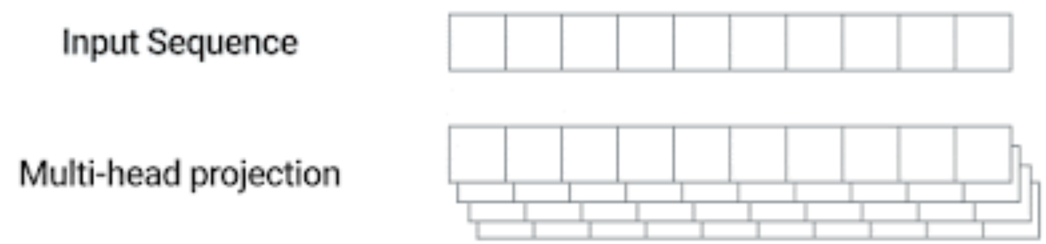
- Content-based sparse attention

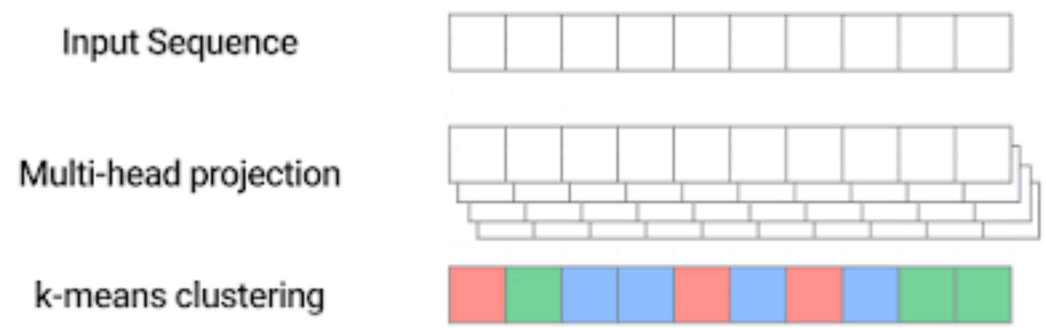


Tokens are assigned to different clusters
Attention is performed only within each cluster

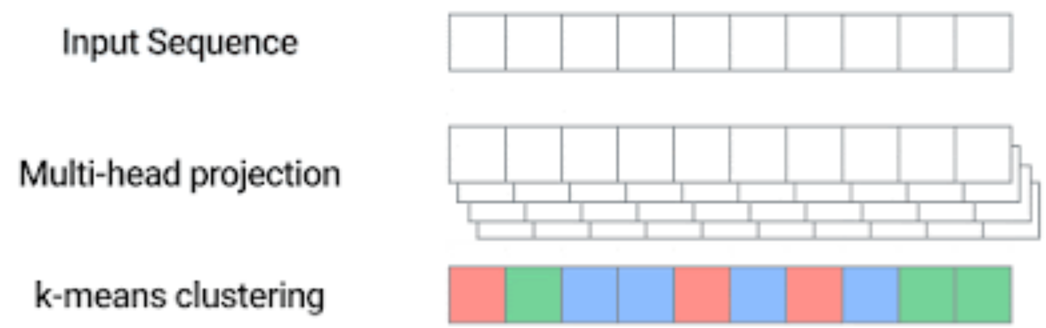
Pic screenshot from: https://arxiv.org/pdf/2003.05997.pdf

Input Sequence 

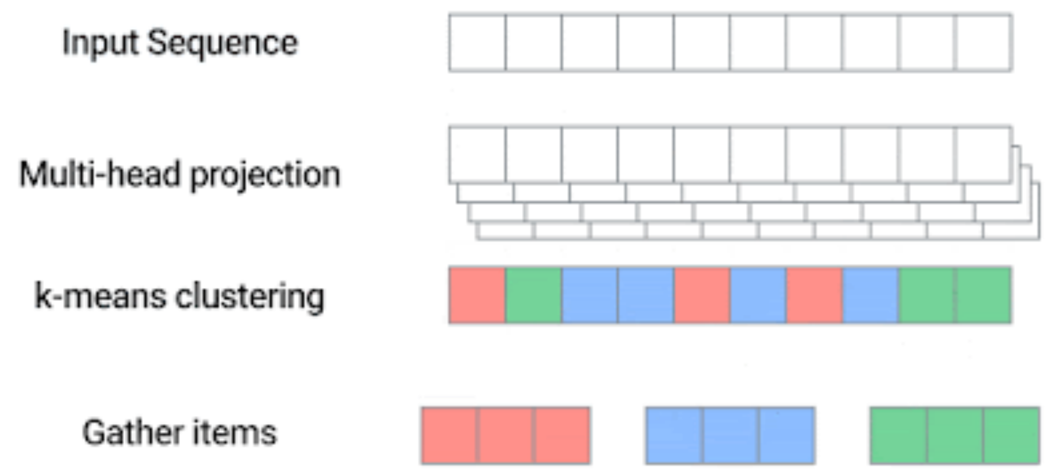

Input Sequence
Multi-head projection

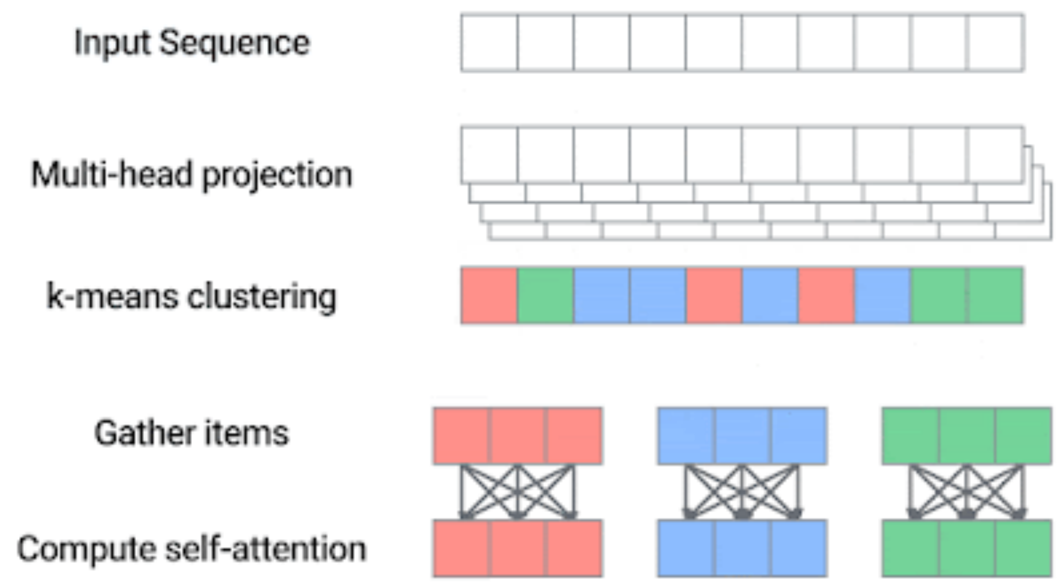


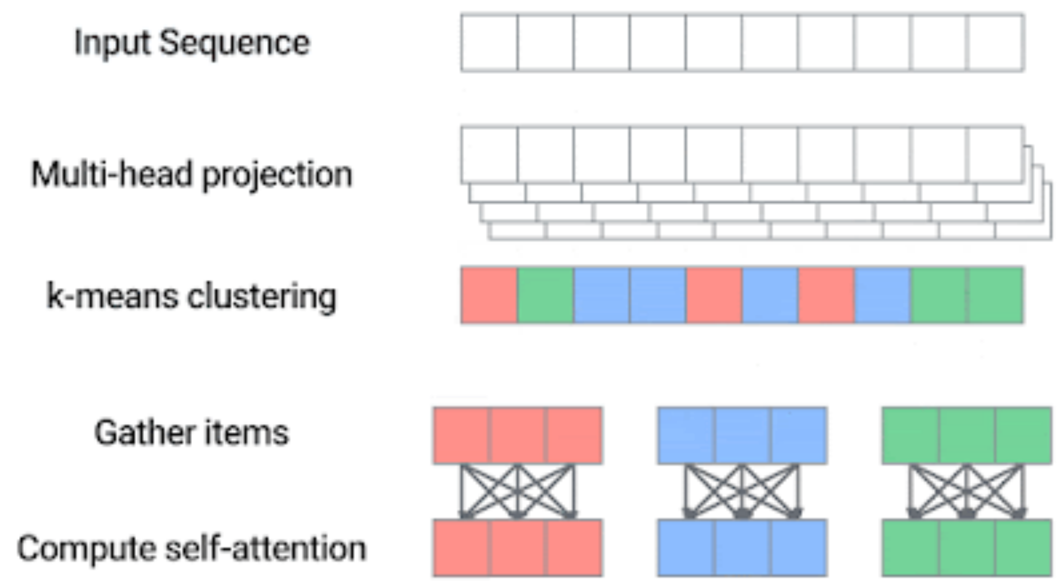
Input Sequence
Multi-head projection
k-means clustering



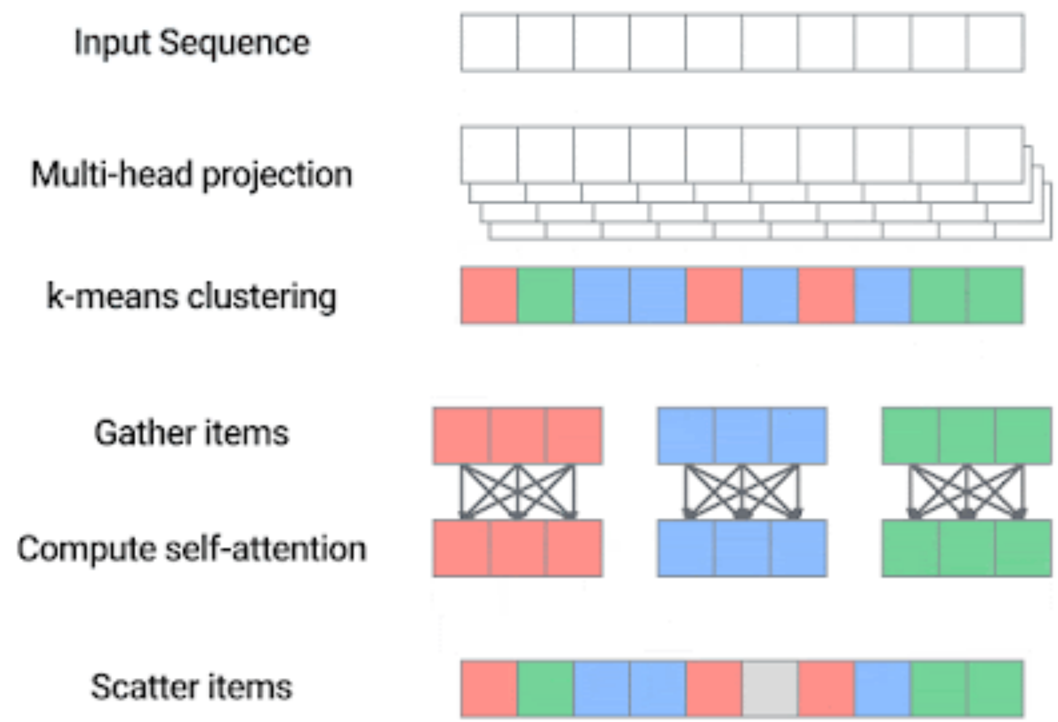
Both queries and keys are clustered with the same centroid vectors!

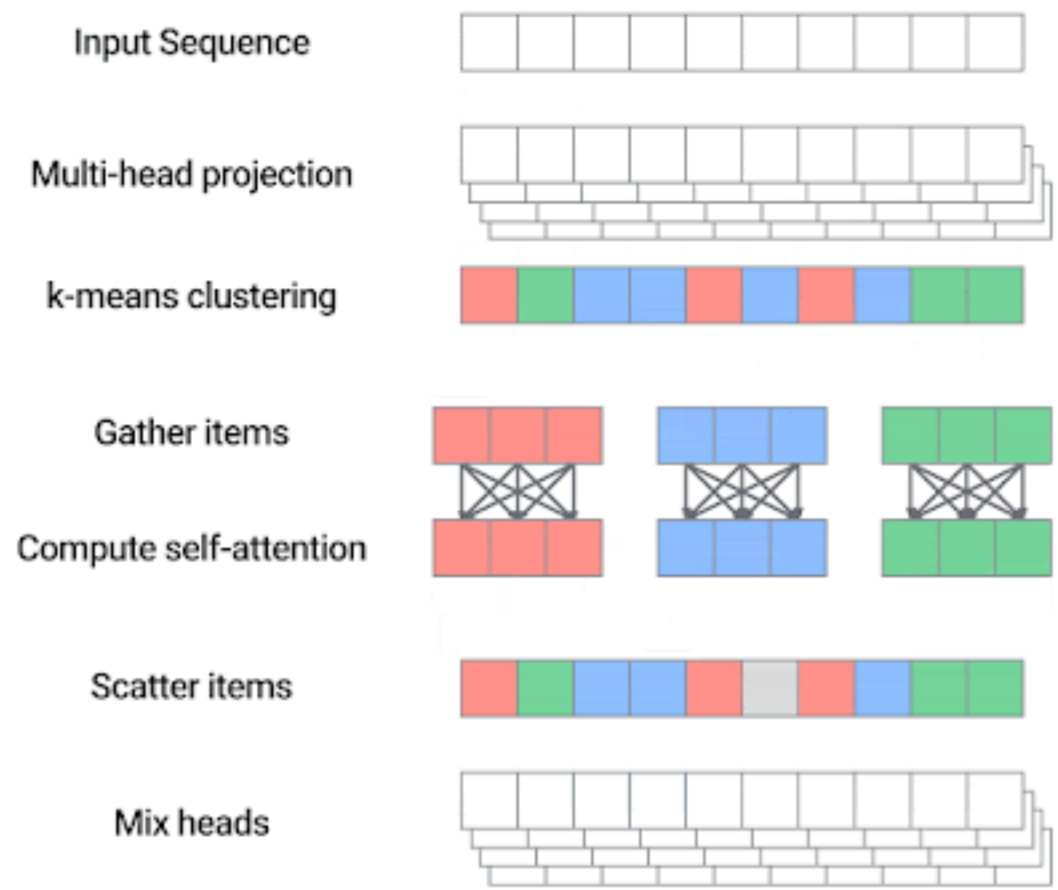

Input Sequence
Multi-head projection
k-means clustering
Gather items

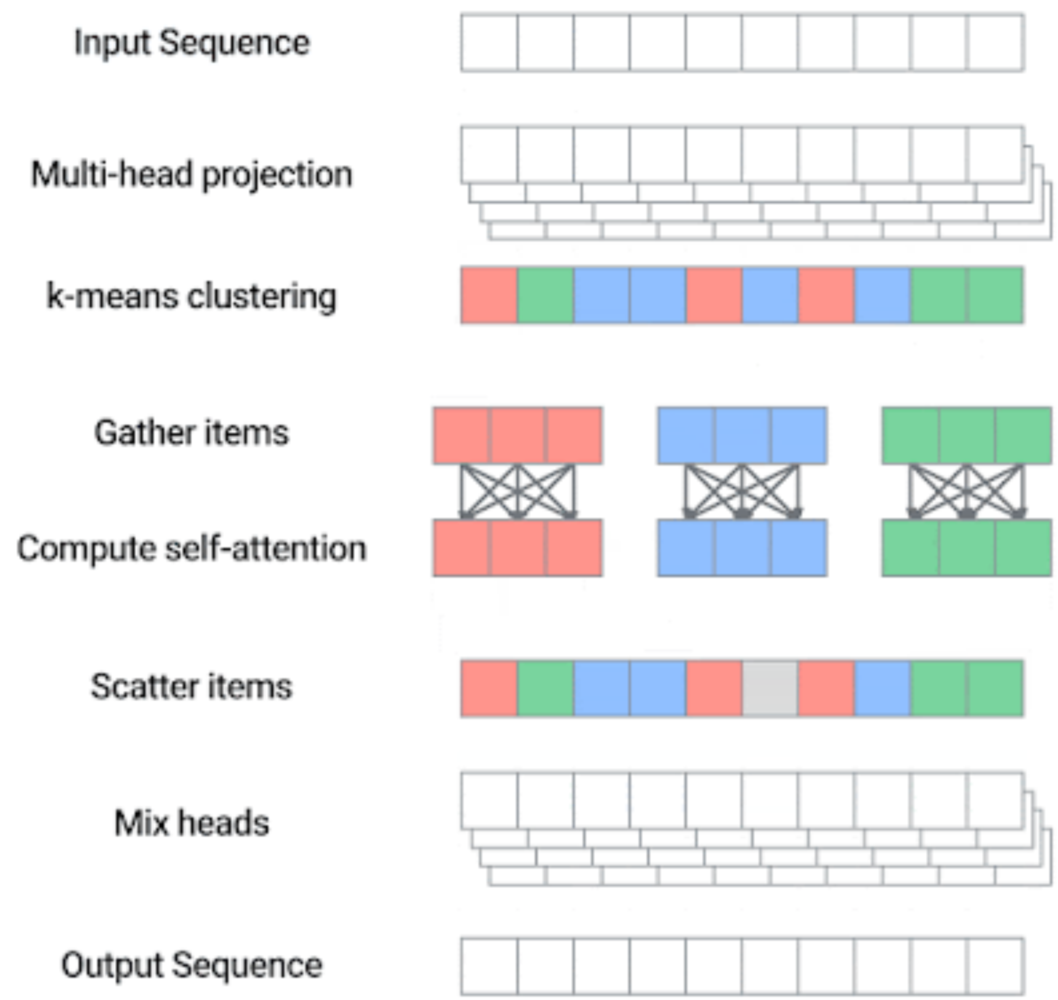

Input Sequence
Multi-head projection
k-means clustering
Gather items
Compute self-attention



Only queries and keys that belong to the same cluster are considered in self-attention


Input Sequence
Multi-head projection
k-means clustering
Gather items
Compute self-attention
Scatter items


Input Sequence
Multi-head projection
k-means clustering
Gather items
Compute self-attention
Scatter items
Mix heads



Input Sequence
Multi-head projection
k-means clustering
Gather items
Compute self-attention
Scatter items
Mix heads
Output Sequence

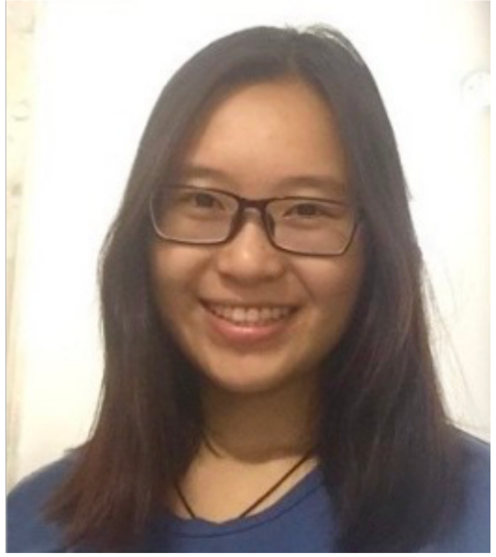
Model	Layers	Heads	Perplexity
Local Transformer	24	8	39.3
TransformerXL (Dai et al., 2019)	36	–	36.3
Compressive Transformer (Rae et al., 2020)	36	–	33.6
Routing Transformer	22	8	**33.2**

Table 5: Results on language modeling on `PG-19` data-set. Local Transformer refers to Transformer (Vaswani et al., 2017) with relative position encoding (Shaw et al., 2018) together with local attention. Perplexity is normalized by the number of tokens reported in (Rae et al., 2020) and is reported on the test set.

What do these models do with these longer contexts?

Do long-range language models actually use long-range context?

EMNLP 2021

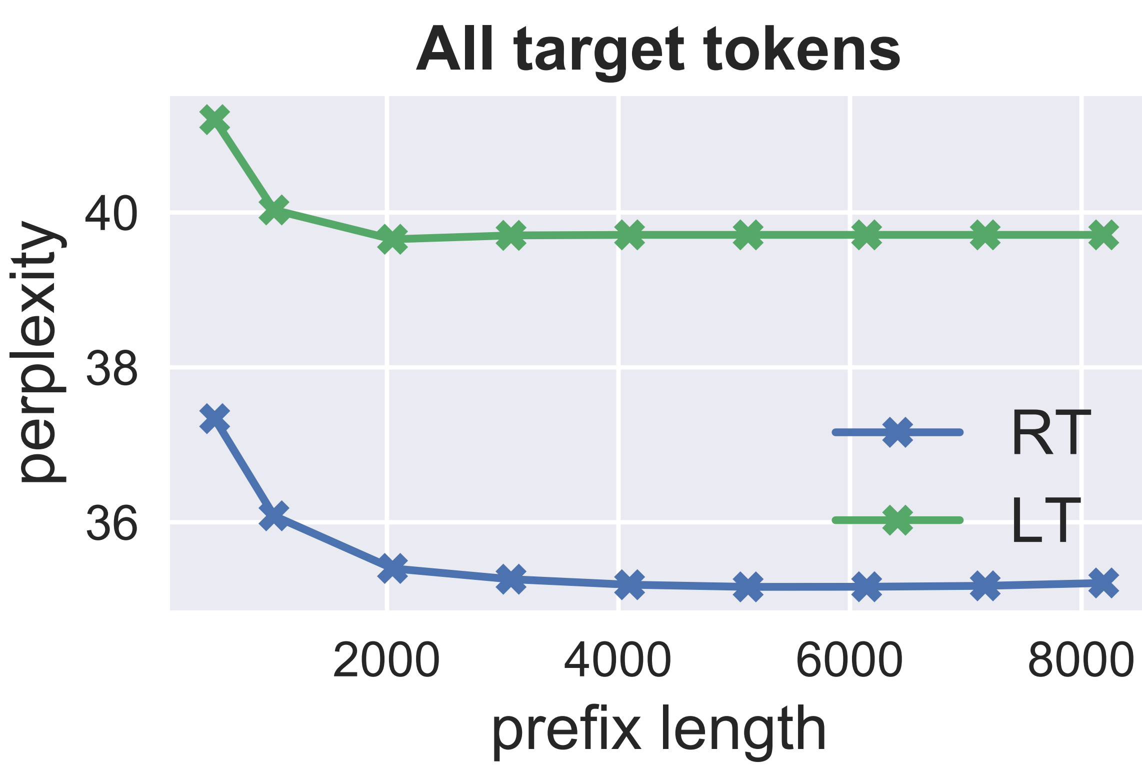


Simeng Sun

Kalpesh Krishna

The effect of longer context

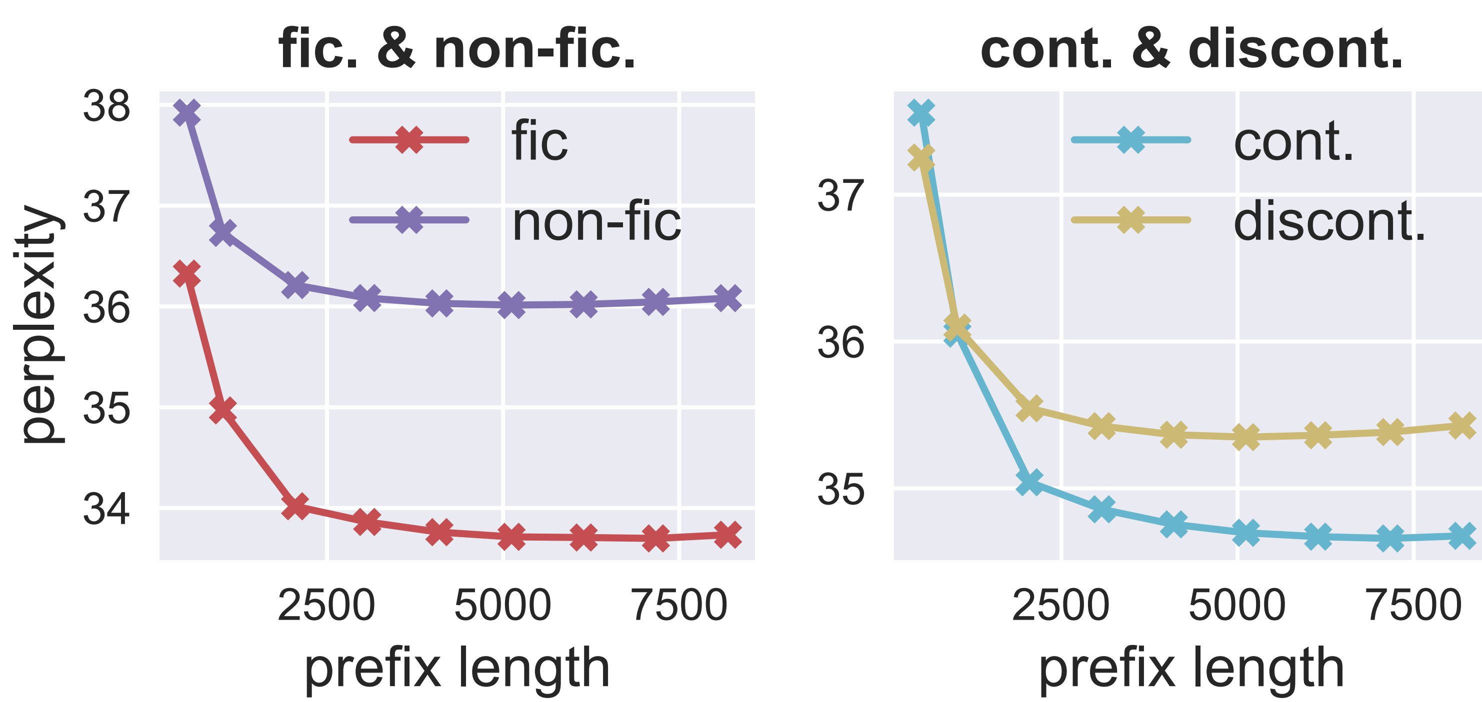
- Target tokens sampled from PG-19 validation set
- Compute perplexity of target tokens preceded by prefix of various lengths



Providing long-range context (i.e., further than 2K tokens away) has negligible effect on the perplexity of target tokens in aggregate

The effect of longer context

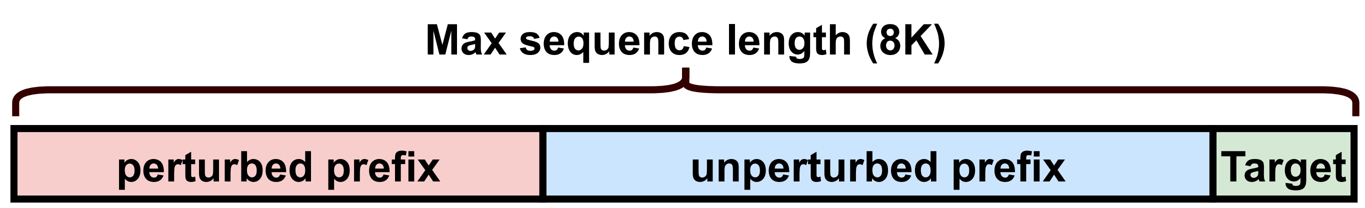
- **Fictional** vs. **non-fictional**
- **Continuous** vs. **discontinuous**



RT takes better advantage of context beyond 2K tokens for fictional and continuous books than non-fictional and discontinuous books.

The perturbation of longer context

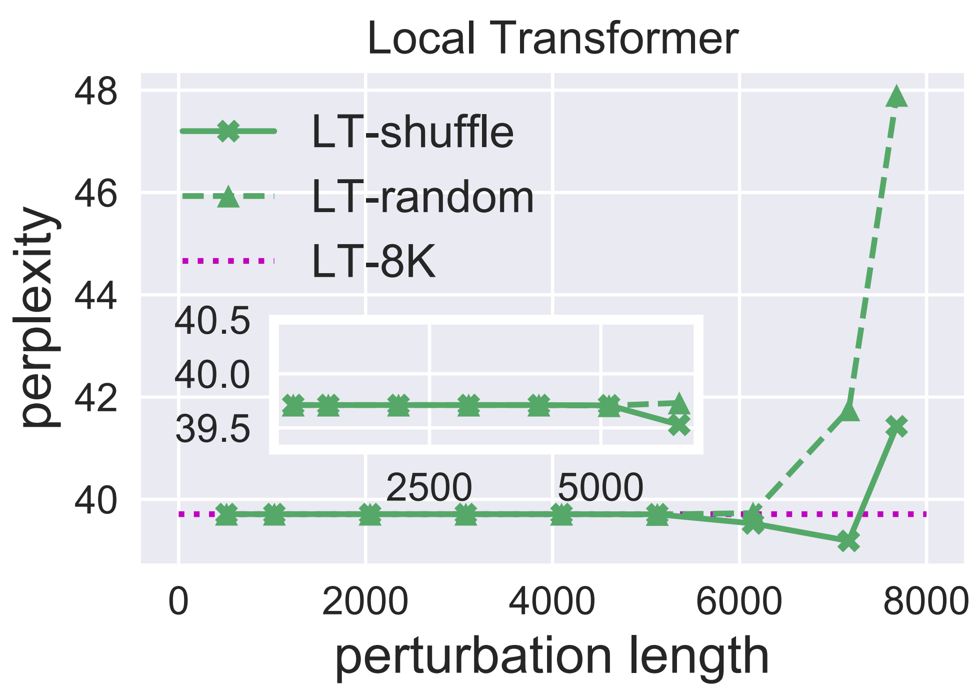
- Fixed context size, perturb distant context

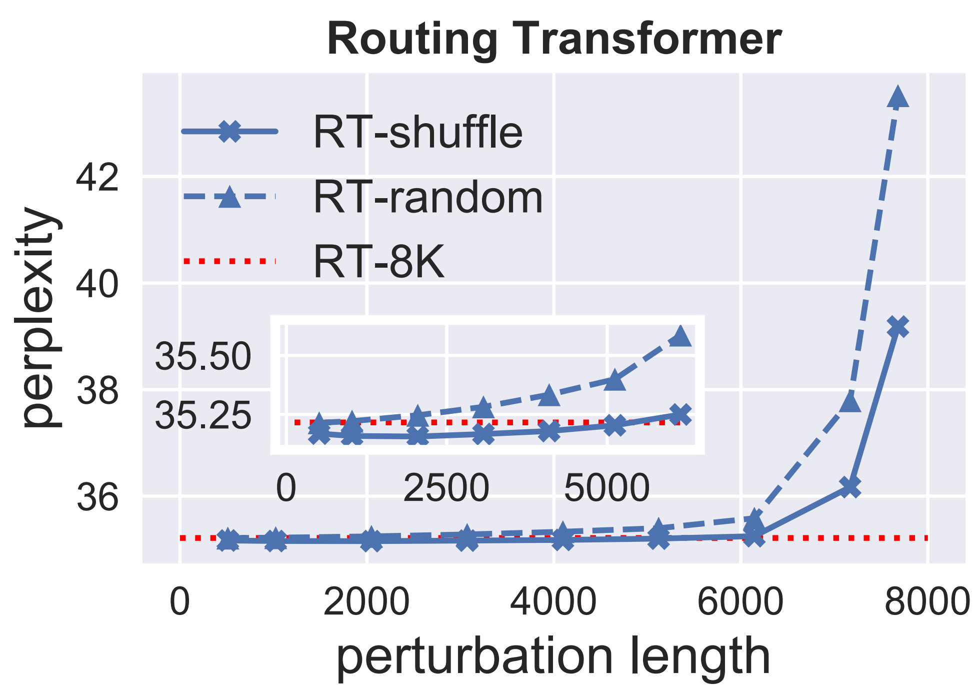


- Types of perturbation
    - Shuffling
    - Random replacement
    - Specific token dropout

The perturbation of longer context

- Evaluate over all target tokens

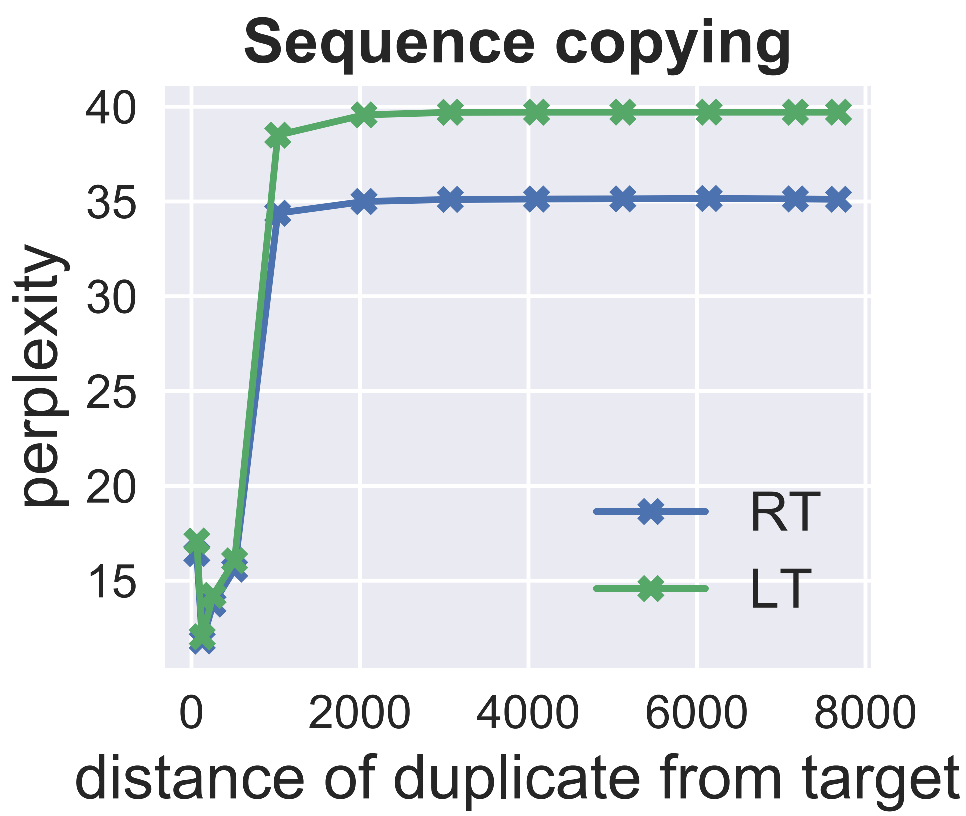




Sequence-level perturbations further than 2K tokens from the target have minimal impact on perplexity

Sequence-level tasks

- How well does the model memorize a sequence that occurs in the distant context?



Word order in the long-range context is not well encoded by both models

While these LMs can *computationally* handle long contexts, they have a long way to go before they can take advantage of the information within these contexts!