Natural Language Processing

第八周 XFormer

庞彦
yanpang@gzhu.edu.cn

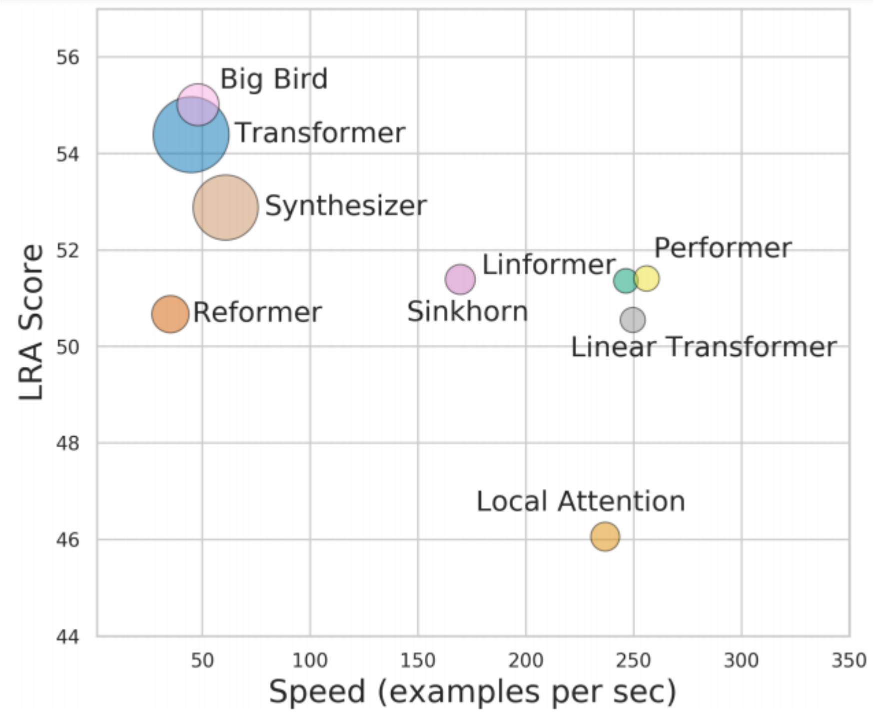
Overview

CONTENTS

01 More Attention Mechanisms

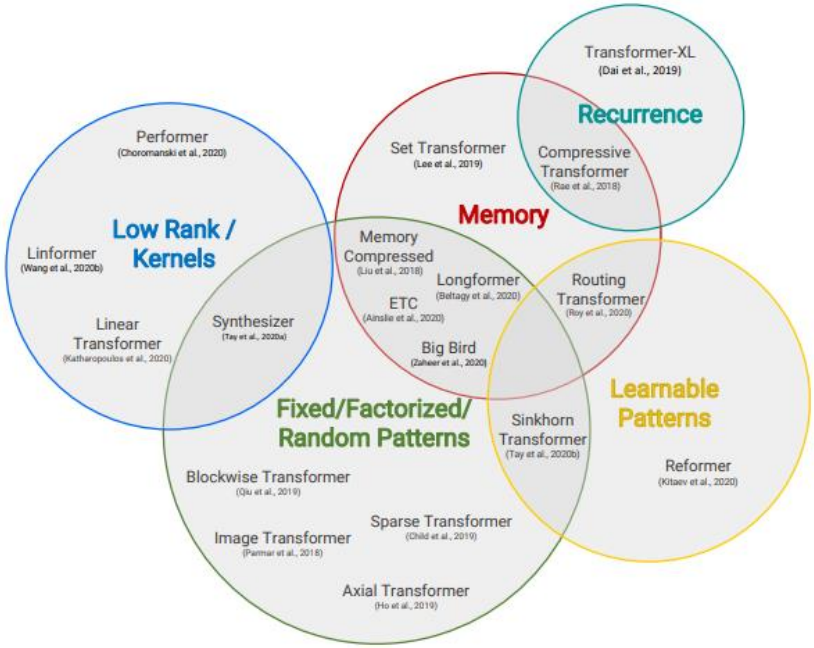
更多的自注意力机制

To Learn More …



Long Range Arena: A Benchmark for Efficient Transformers

https://arxiv.org/abs/2011.04006

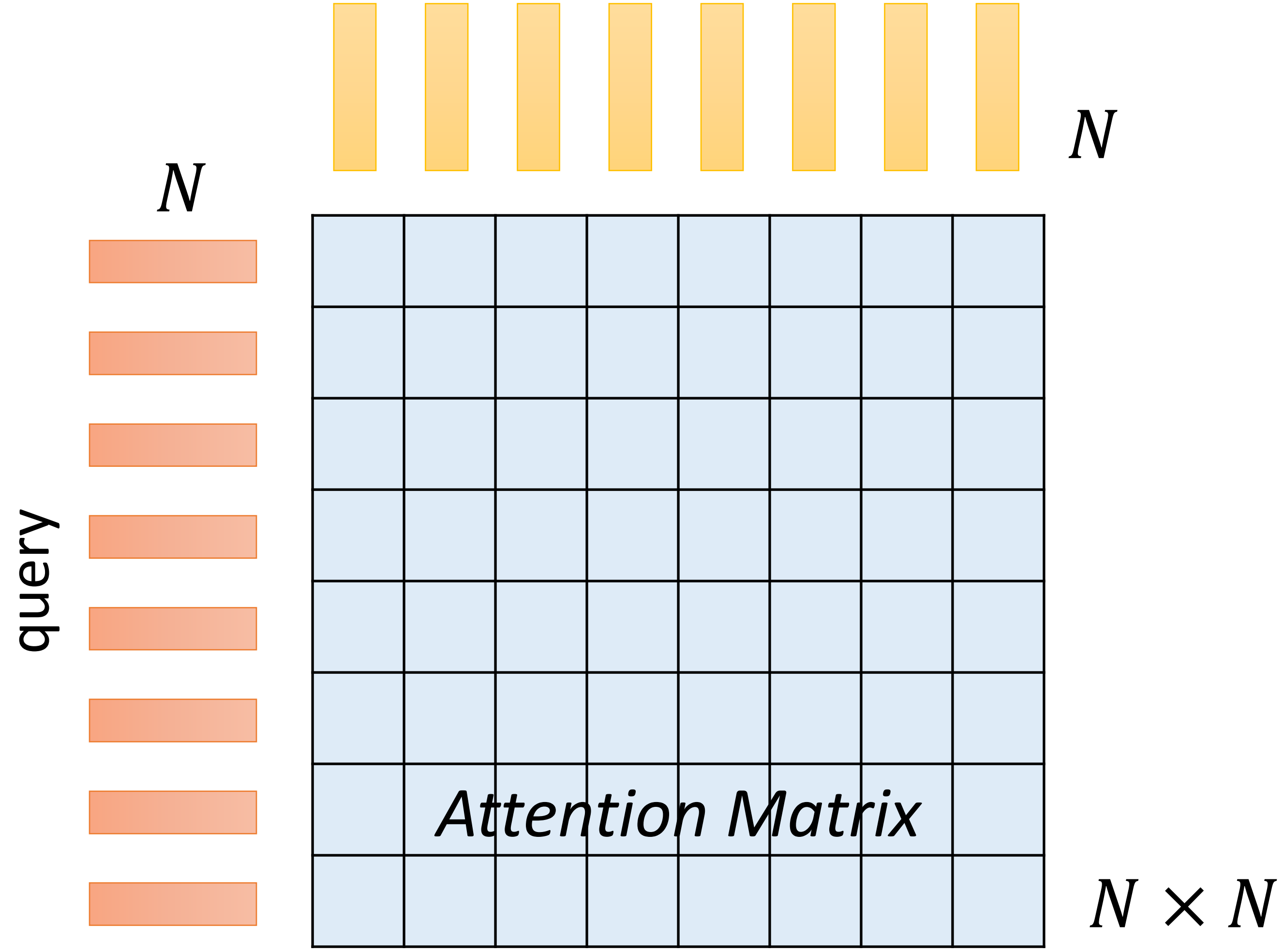


Efficient Transformers: A Survey

https://arxiv.org/abs/2009.06732

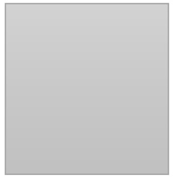
How to make self-attention efficient?

Sequence length = N



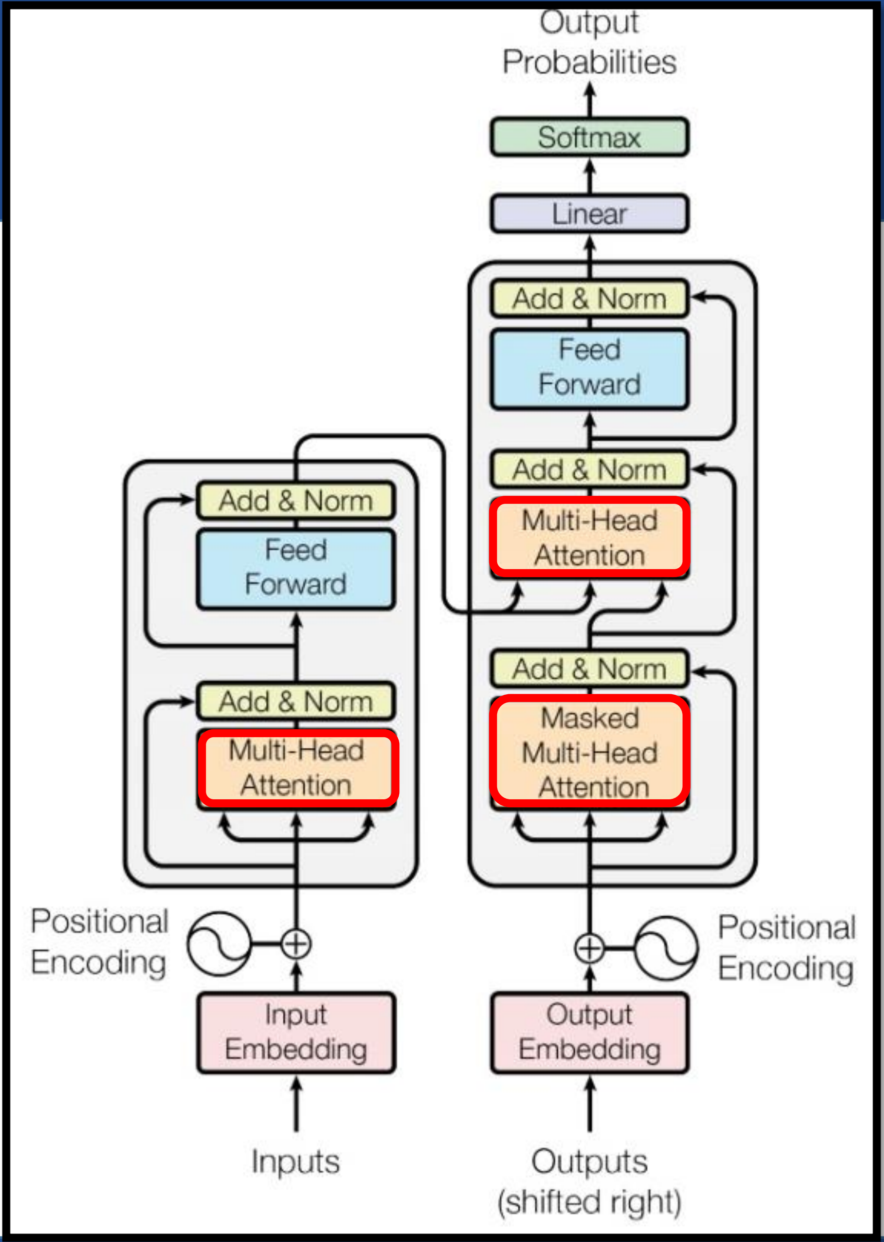
Notice

- Self-attention is only a module in a larger network. 自注意力机制仅仅知识大网络的一个模块。
- Self-attention dominates computation when N is large. 当N非常大的时候，自注意力机制的计算量显著上升。
- Usually developed for image processing

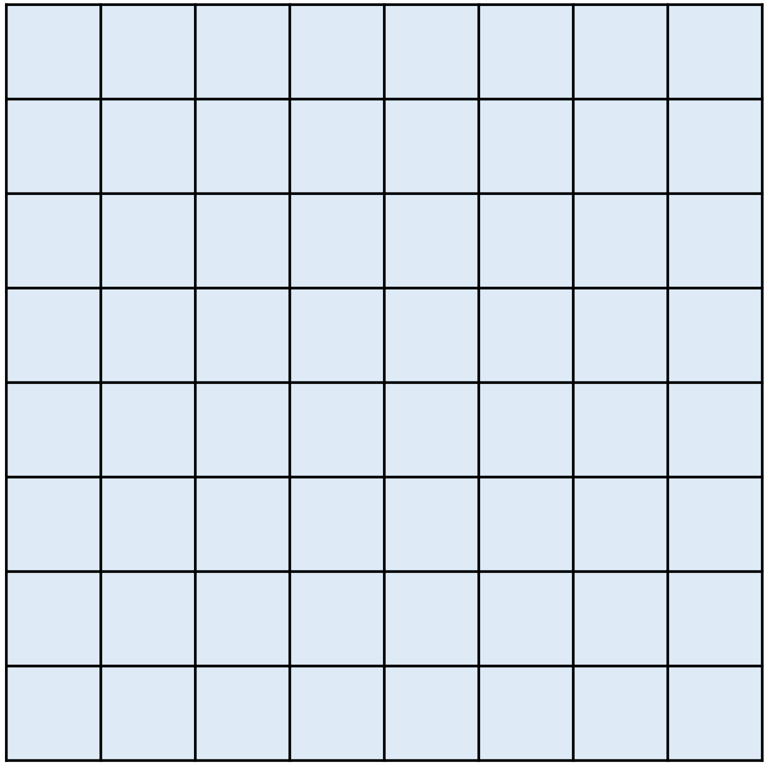
256 

256

$$N = 256 * 256$$

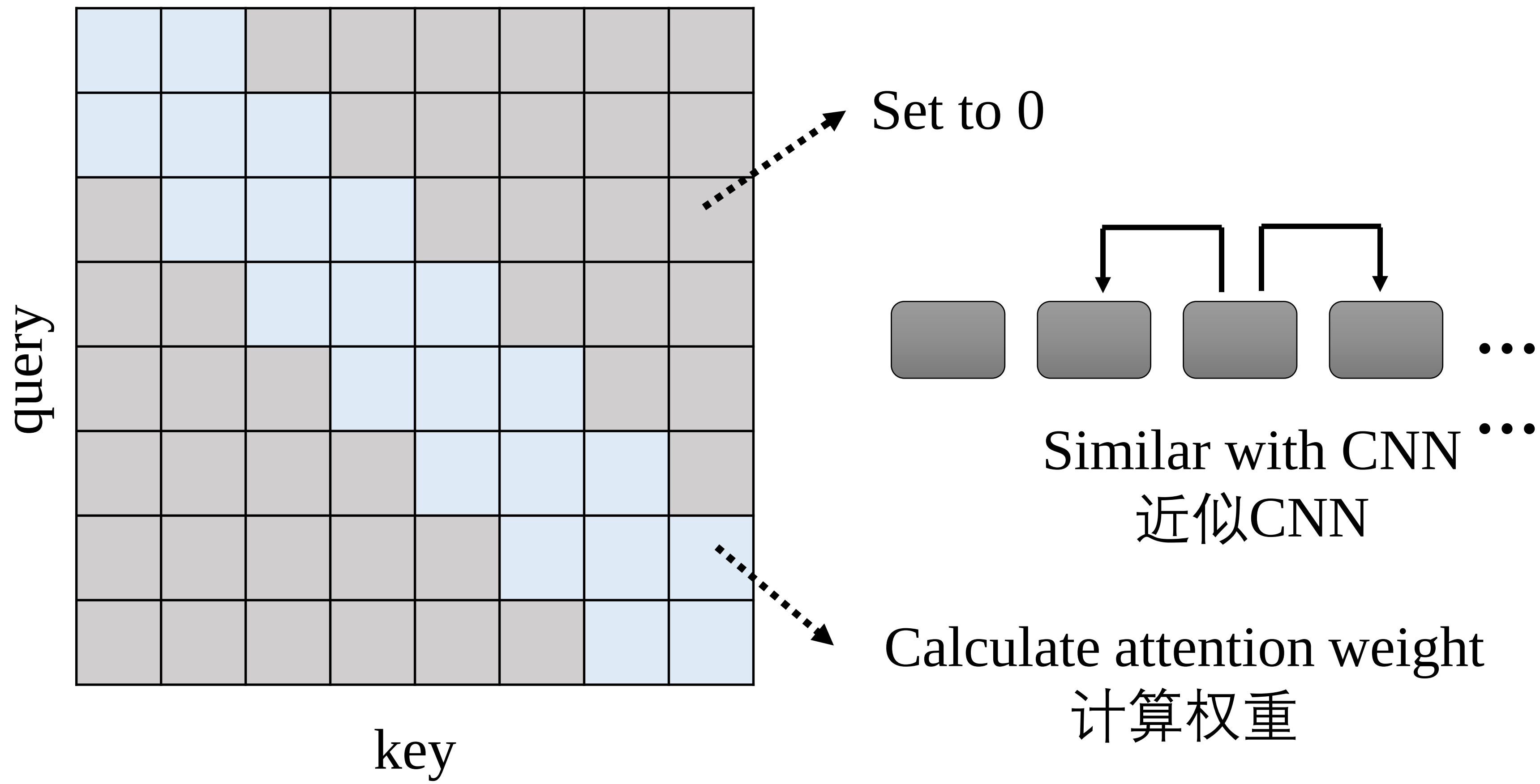


Skip Some Calculations with Human Knowledge

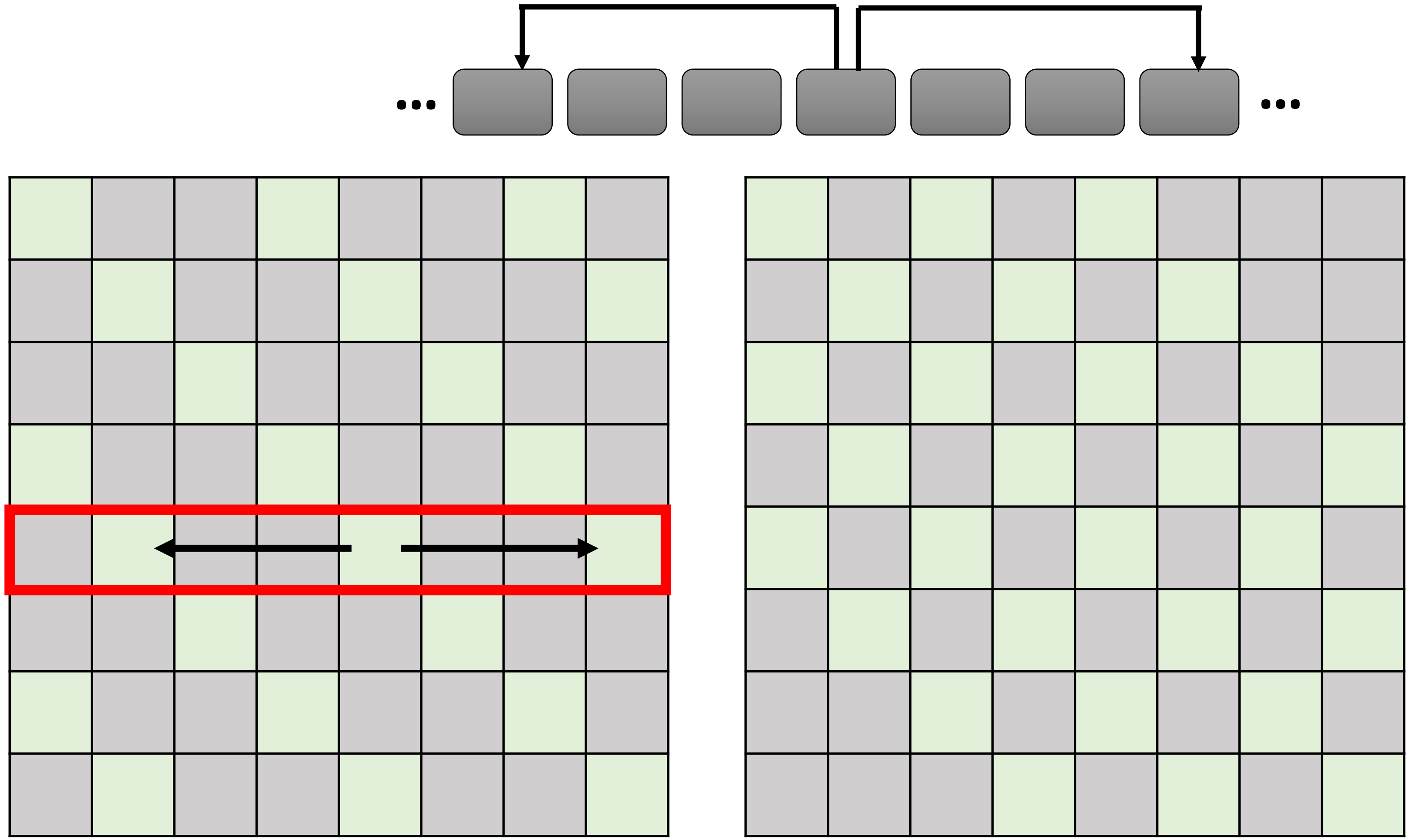


Can we fill in some values with human knowledge?

Local Attention / Truncated Attention



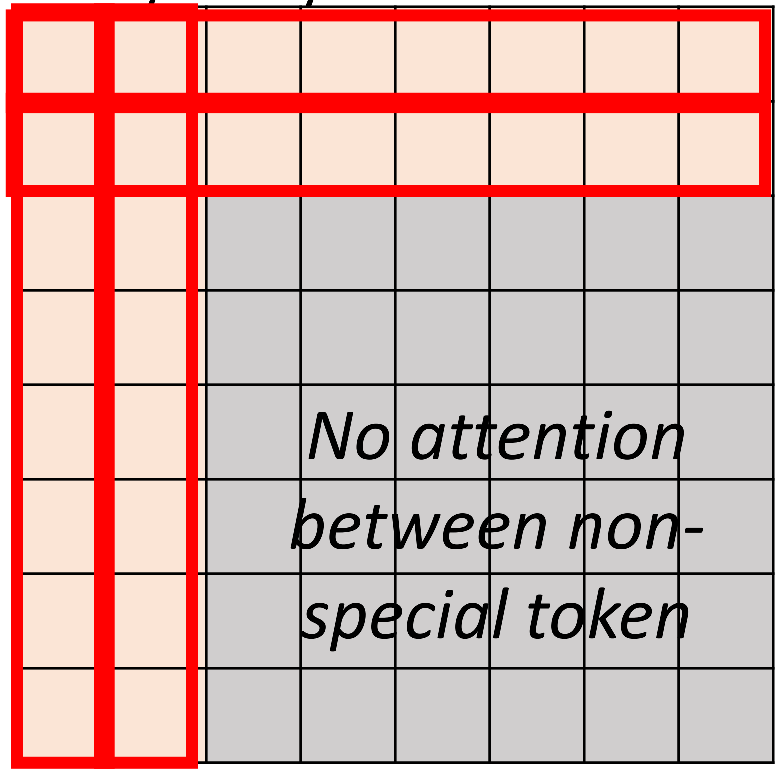
Stride Attention

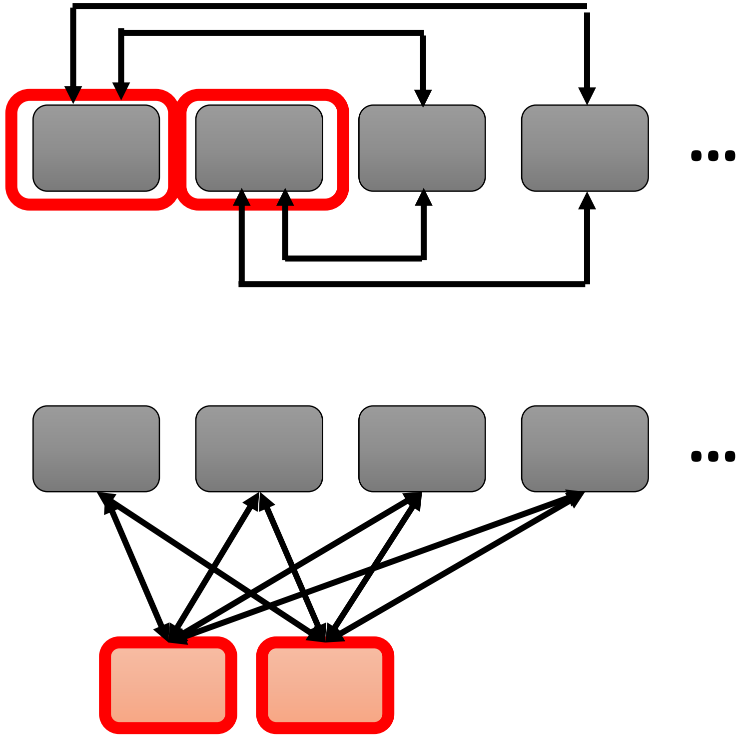


Global Attention

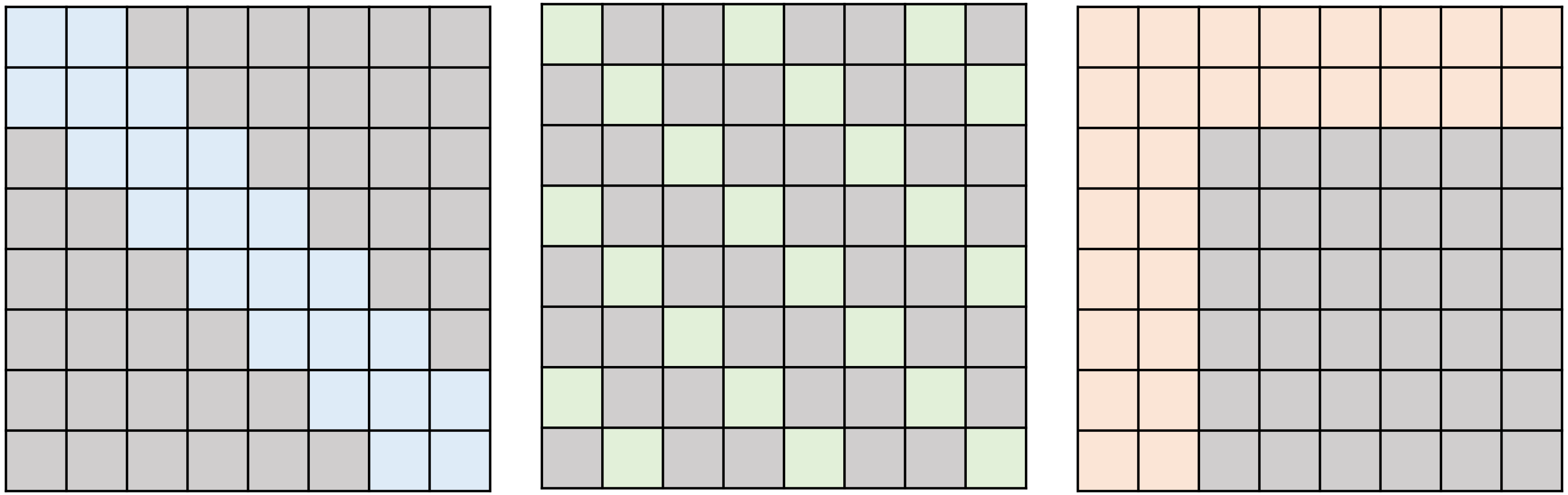
Add special token into original sequence在原句中新增特殊的token

- Attend to every token → collect global information搜集全局信息
- Attended by every token → it knows global information





Many Different Choices …



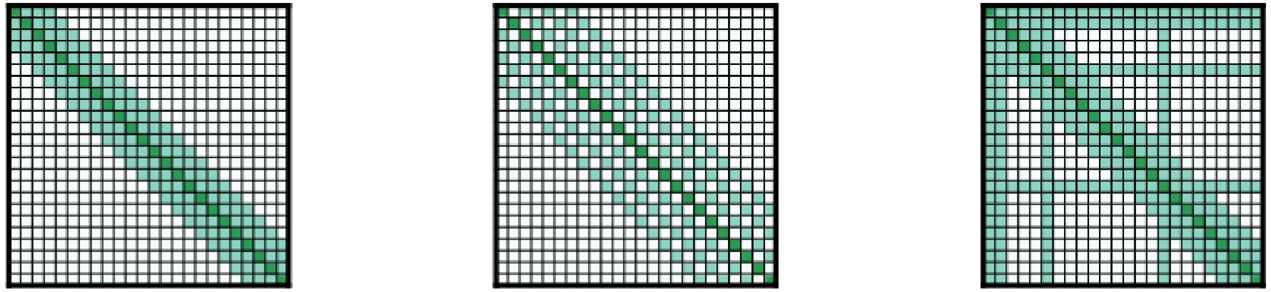
Different heads use different patterns.

不同的头利用不用的特征。

Many Different Choices …

- LongFormer

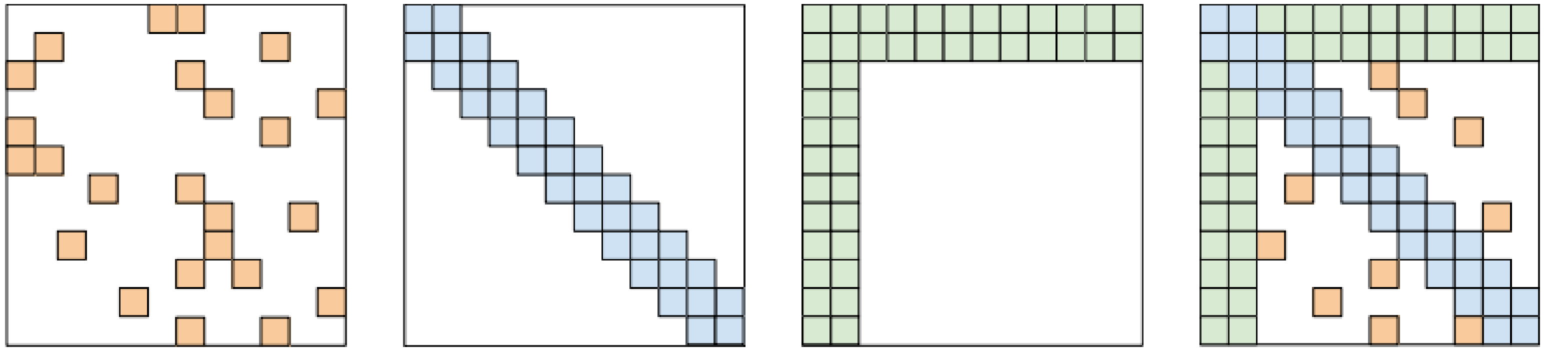
https://arxiv.org/abs/2004.05150



(b) Sliding window attention (c) Dilated sliding window (d) Global+sliding window

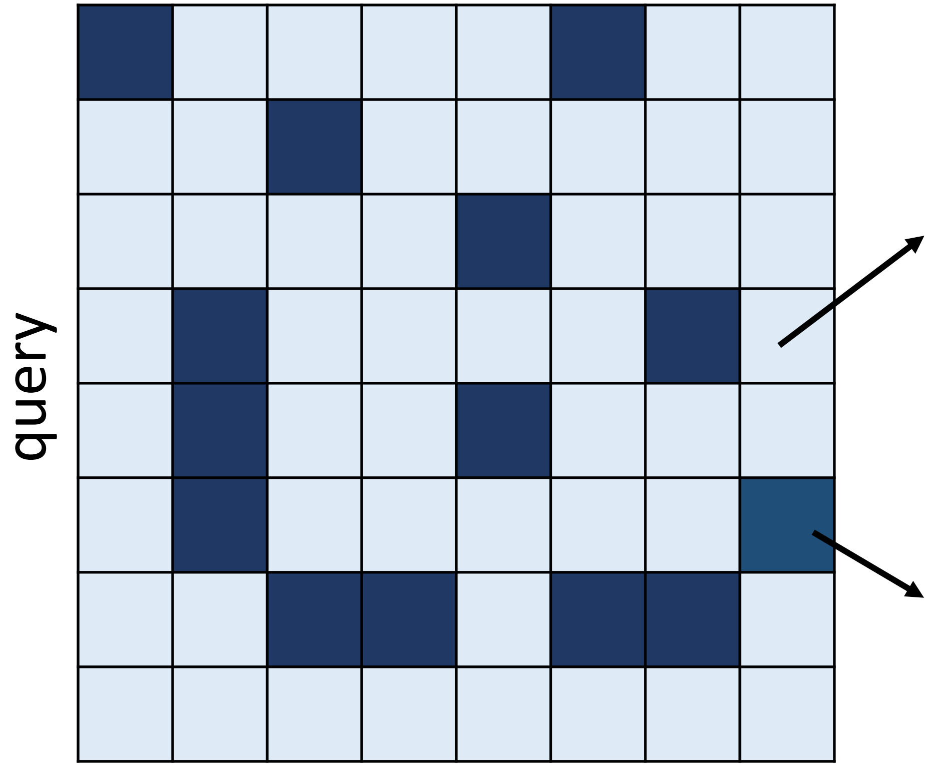
- Big Bird

https://arxiv.org/abs/2007.14062



(a) Random attention (b) Window attention (c) Global Attention (d) BIGBIRD

Can we only focus on Critical Parts?



small value小值

- Directly set to 0直接归零
- Smaller influence on results 对结果影响较小

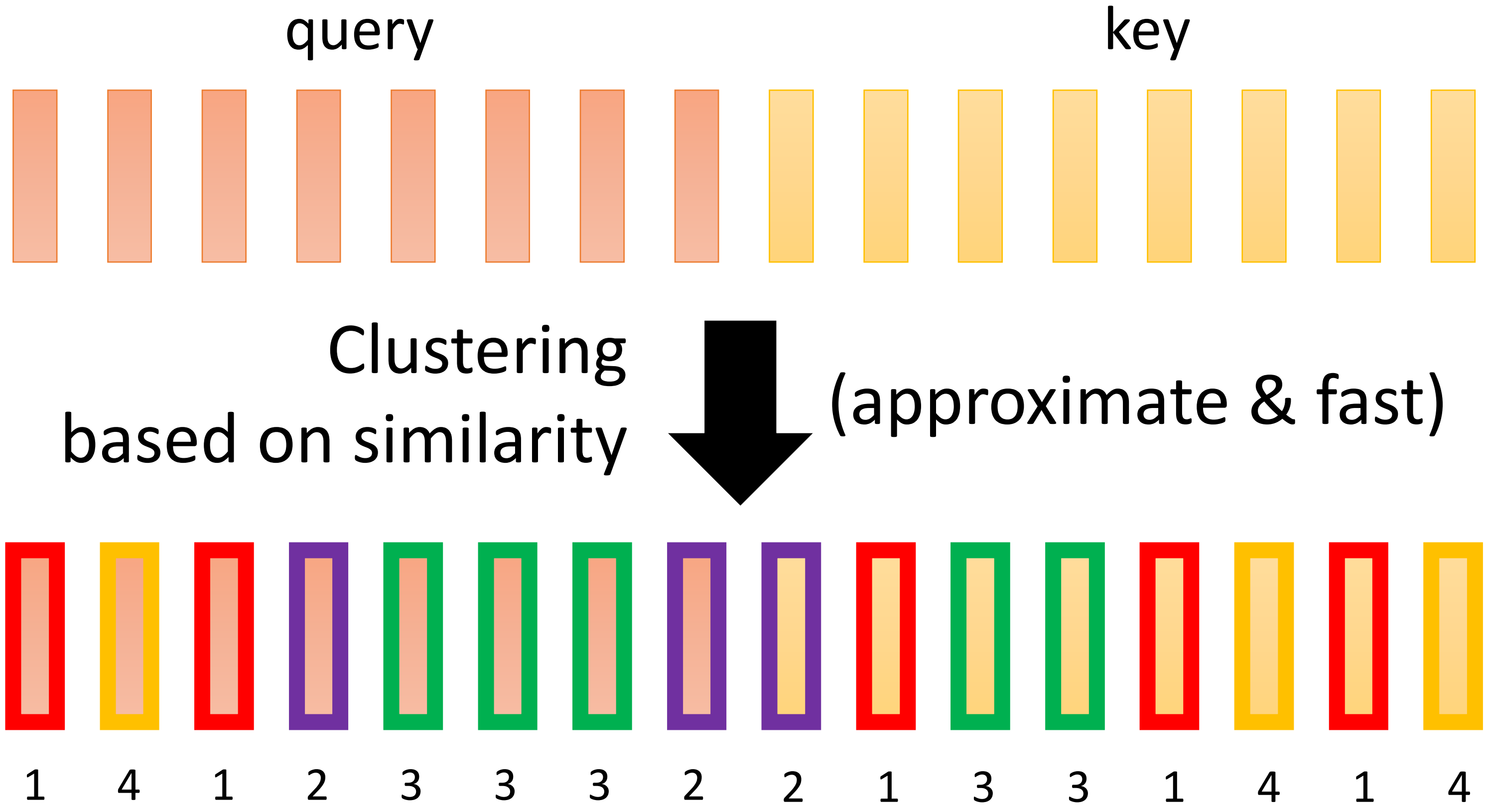
large value大值

How to quickly estimate the portion with small attention weights?

Clustering

Reformer
https://openreview.net/forum?id=rkgNKkHtvB

Step 1

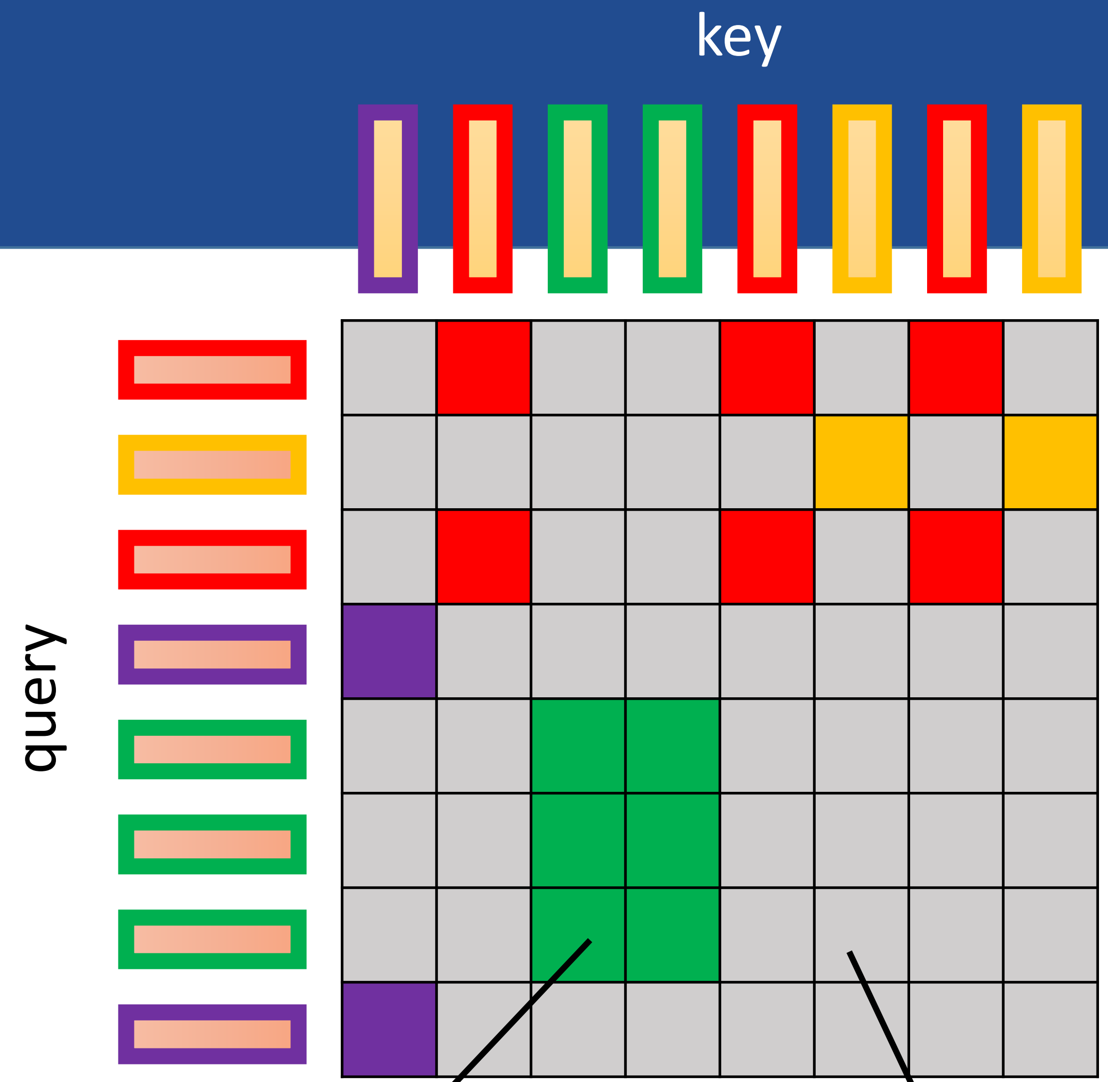


Spring 2023

Routing Transformer
https://arxiv.org/abs/2003.05997

Clustering

Step 2

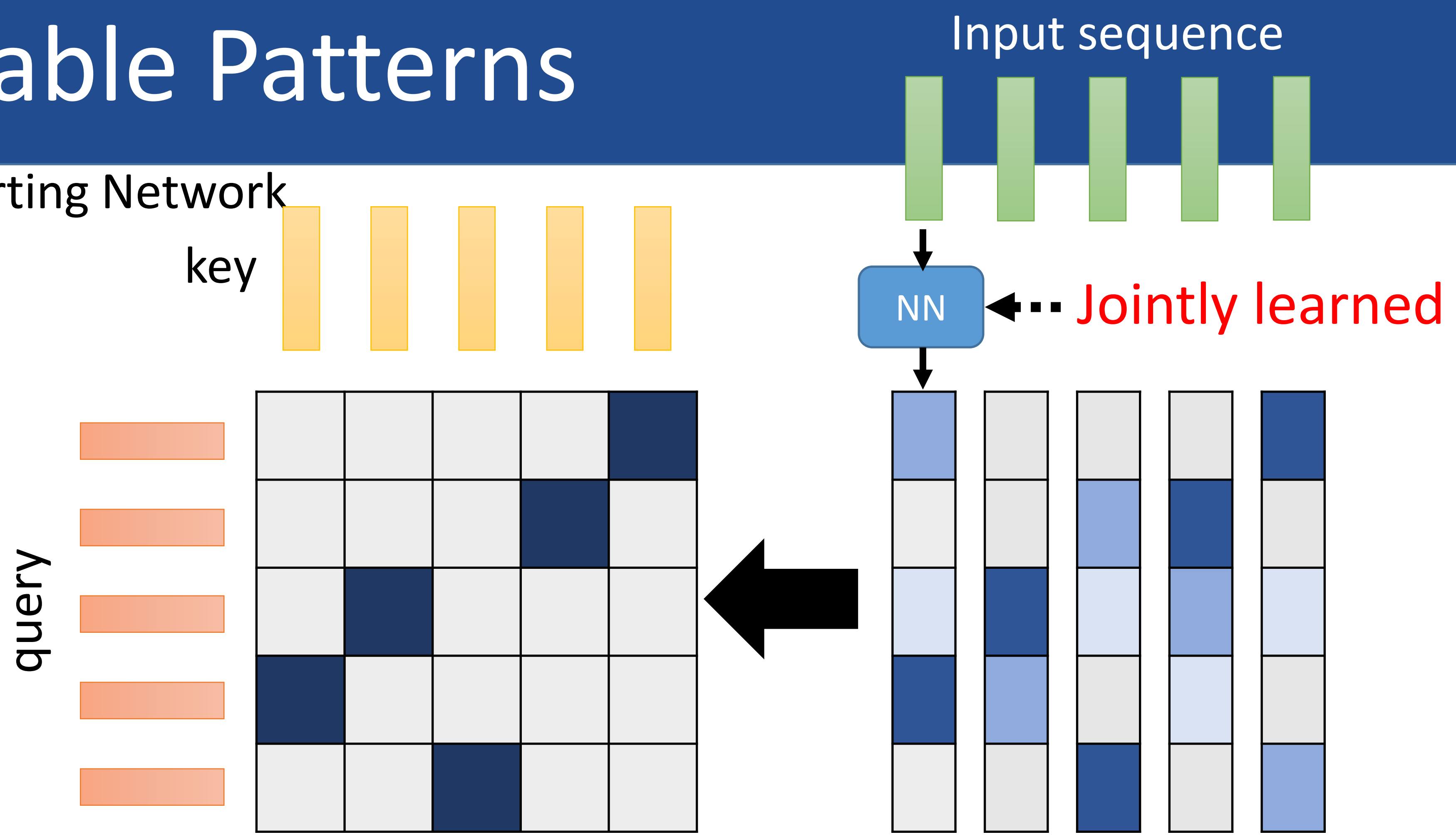


Belong to the same cluster, then calculate attention weight

Not the same cluster, set to 0

Learnable Patterns

Sinkhorn Sorting Network



A grid should be skipped or not is decided by another learned module

https://arxiv.org/abs/2002.11296
(simplified version)

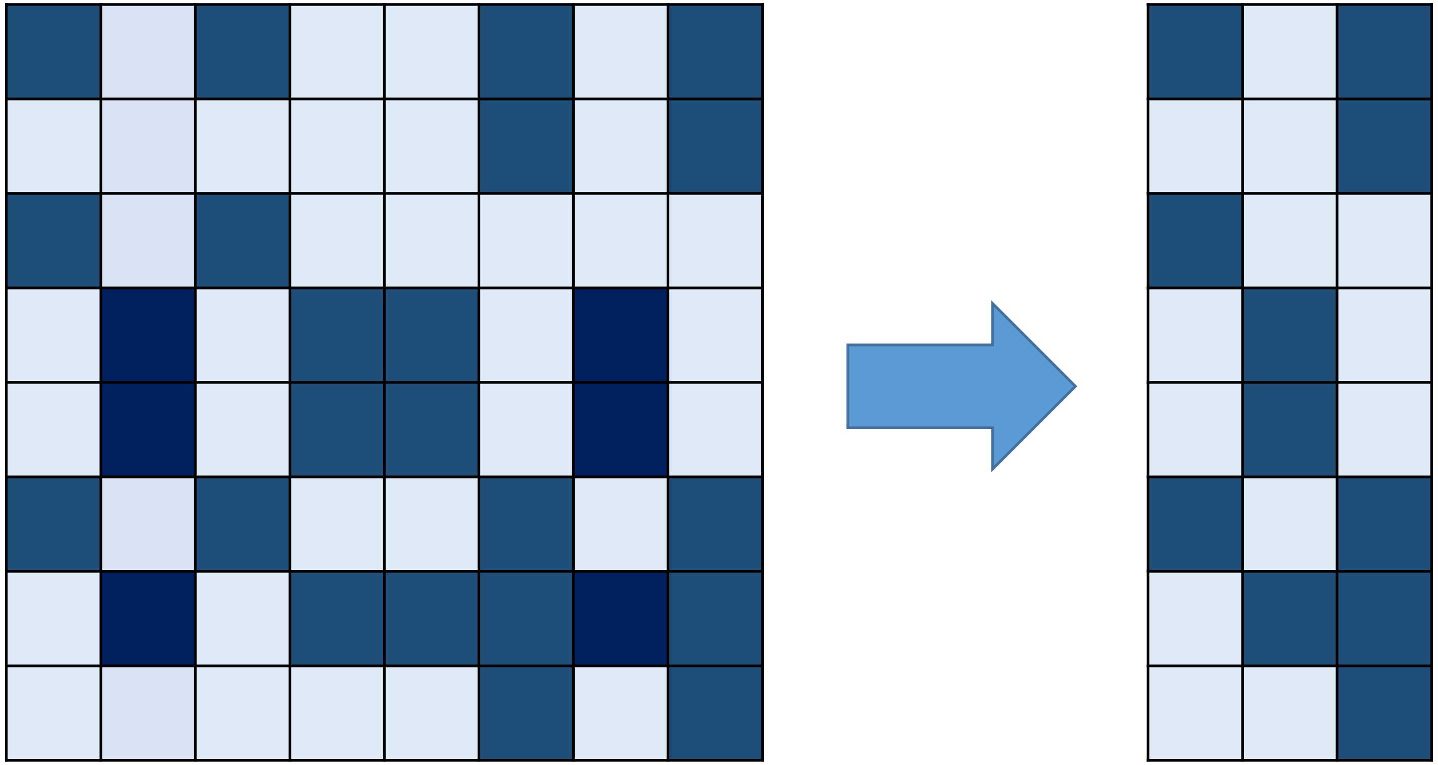
Do we need full attention matrix?

Linformer

https://arxiv.org/abs/2006.04768

Many redundant columns

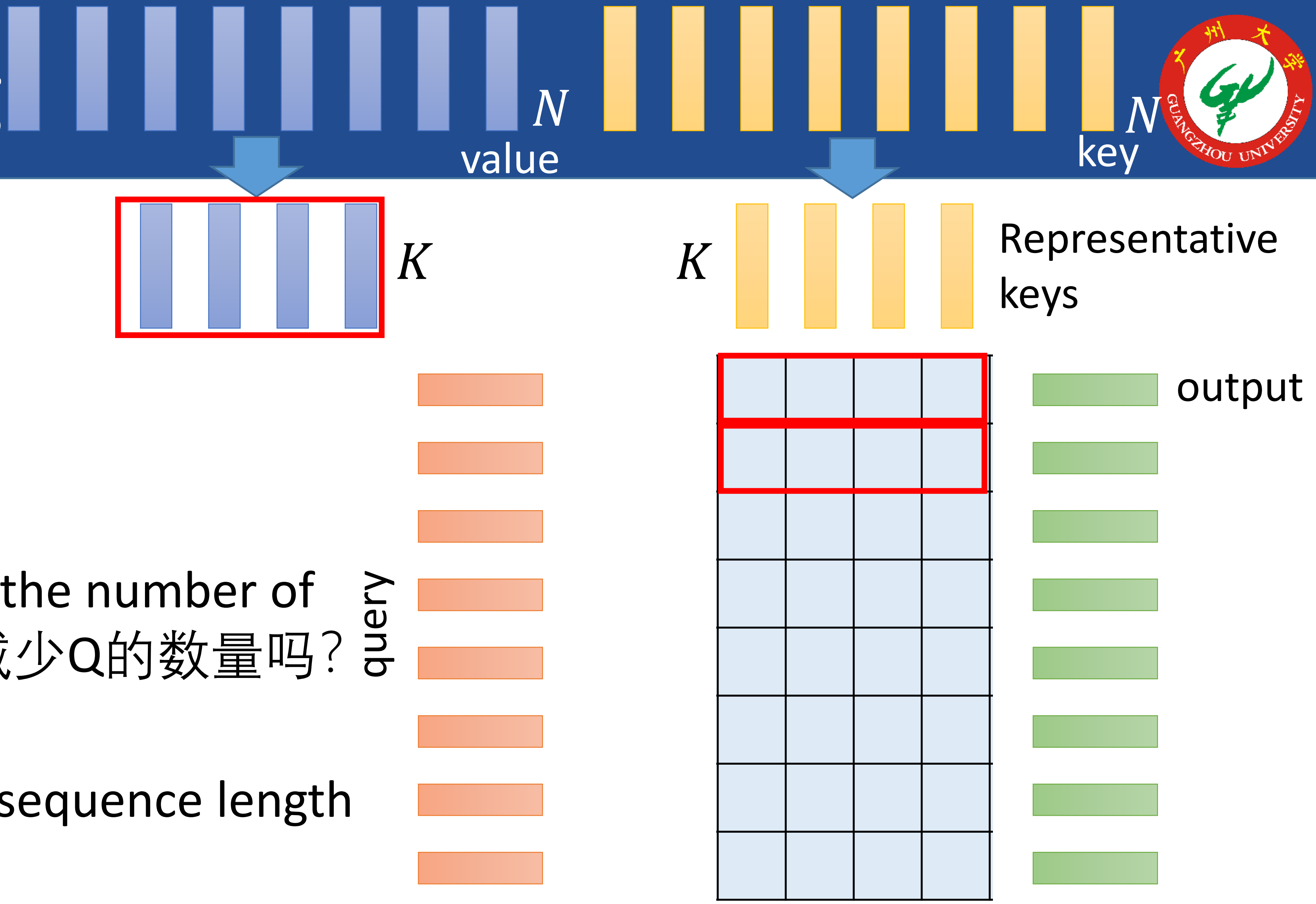
query



key

Low Rank

Clustering

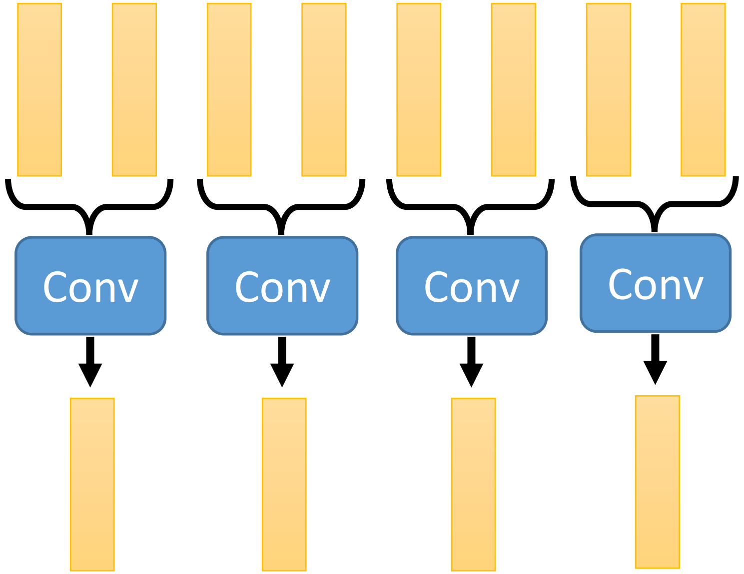


Can we reduce the number of queries?可以减少Q的数量吗？

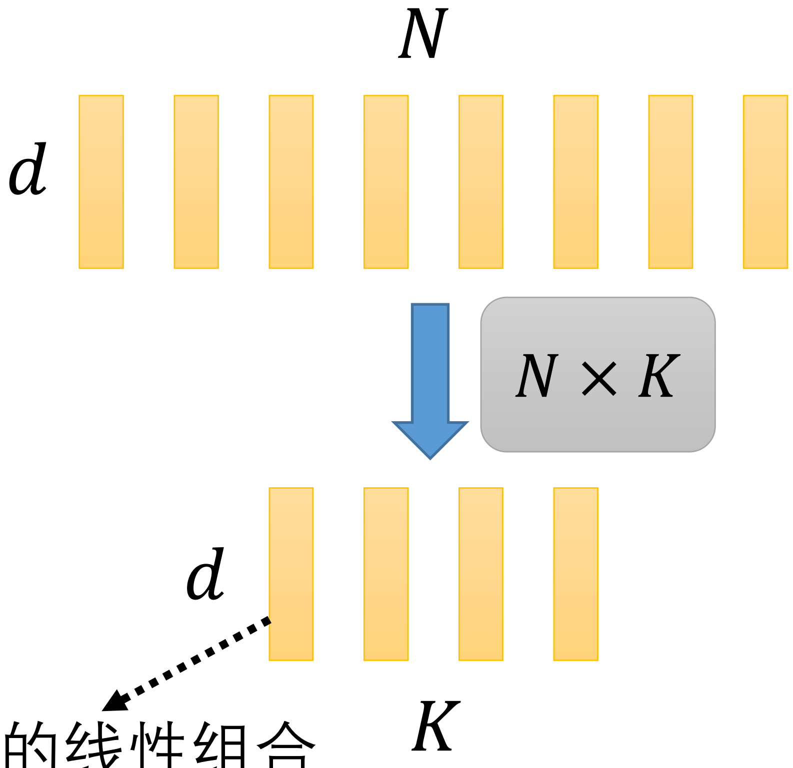
change output sequence length
改变序列长度

Reduce Number of Keys

Compressed Attention
https://arxiv.org/abs/1801.10198



Linformer
https://arxiv.org/abs/2006.04768



Linear combination of N vectors N个矢量的线性组合

Attention Mechanism is three-matrix Multiplication

Review

$$\underset{d \times N}{Q} = W^q \, I$$

$$\underset{d \times N}{K} = W^k \, I$$

$$\underset{d' \times N}{V} = W^v \, I$$

$$A' \xleftarrow{\text{softmax}} A = \boxed{K^T \, Q}$$

ignore

$$O = V \, A'$$

Attention Mechanism is three-matrix Multiplication

Review

$$\underset{d \times N}{Q} = W^q \, I$$

$$\underset{d \times N}{K} = W^k \, I$$

$$\underset{d' \times N}{V} = W^v \, I$$

$$\underset{d' \times N}{O} \approx \underbrace{\underset{d' \times N}{V} \; \underbrace{\underset{N \times d}{K^T} \; \underset{d \times N}{Q}}}$$

Attention Mechanism is three-matrix Multiplication

$$\underset{d' \times N}{O} \approx \underbrace{\underset{d' \times N}{V}\ \underbrace{\underset{N \times d}{K^T}\ \underset{d \times N}{Q}}}$$

$$\underset{d' \times N}{O} \approx \underbrace{\underbrace{\underset{d' \times N}{V}\ \underset{N \times d}{K^T}}\ \underset{d \times N}{Q}}$$

What is the difference?
区别?

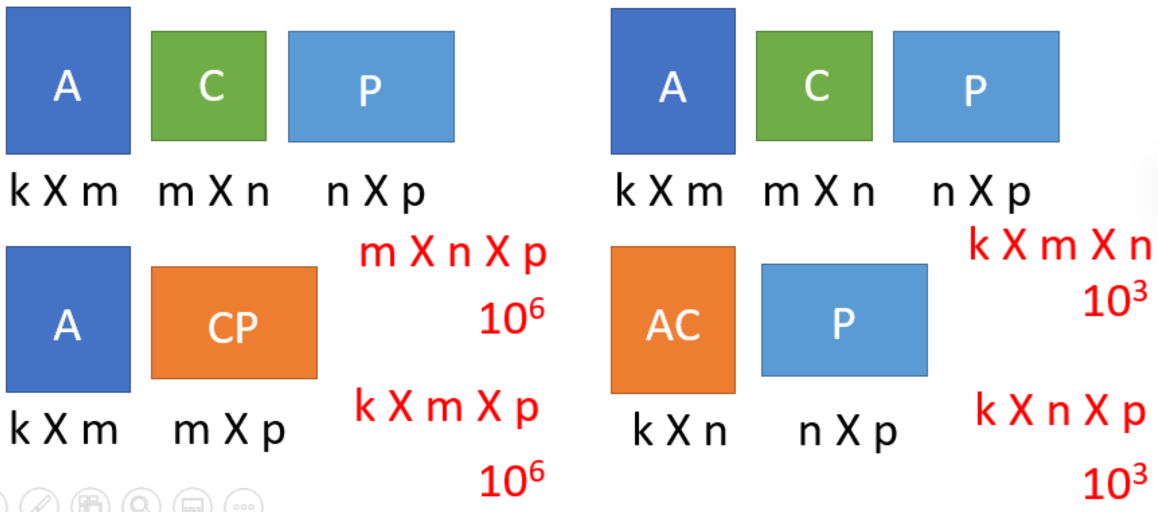
Review Linear Algebra

Practical Issue

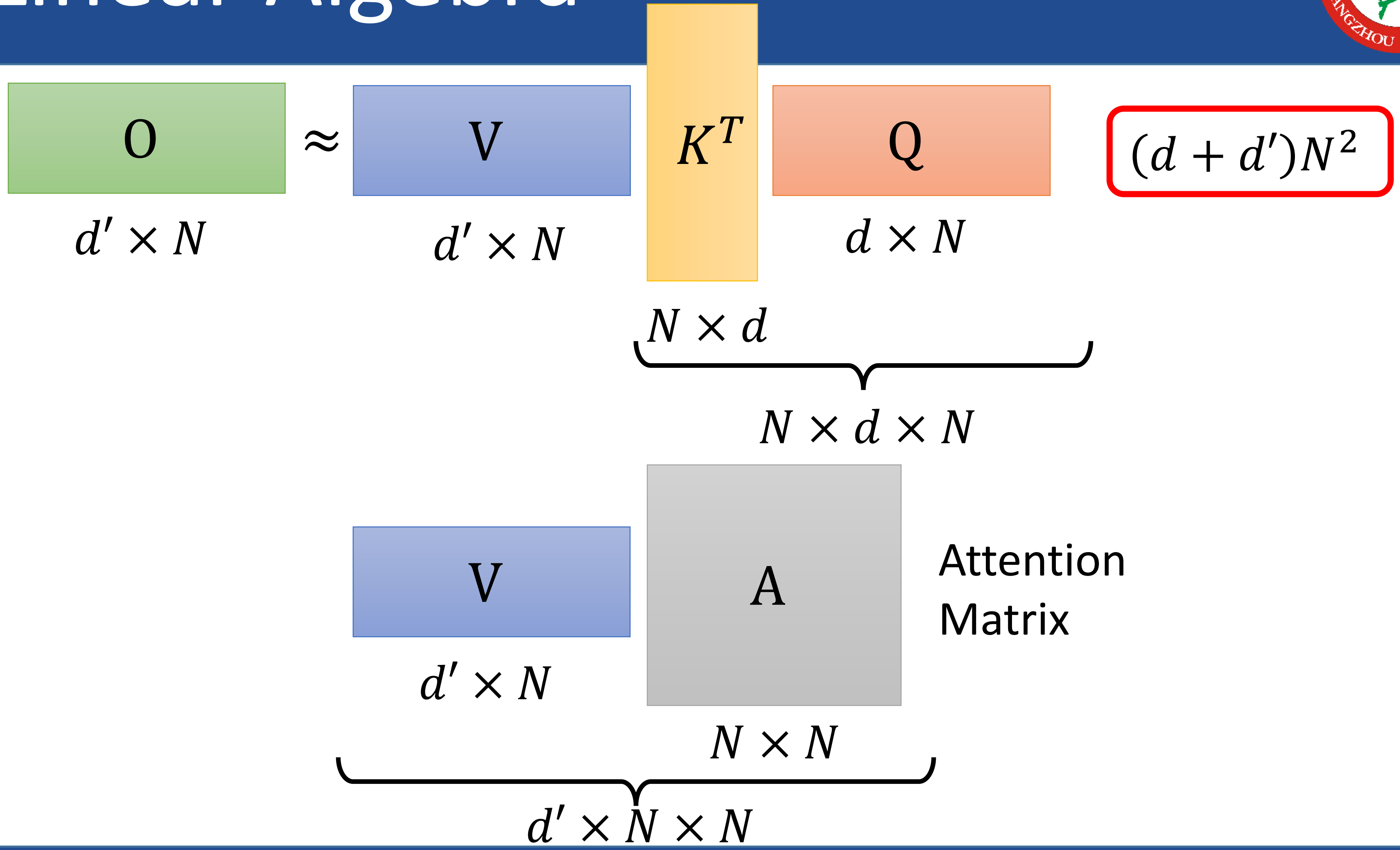
k=1 m=1000

n=1 p=1000

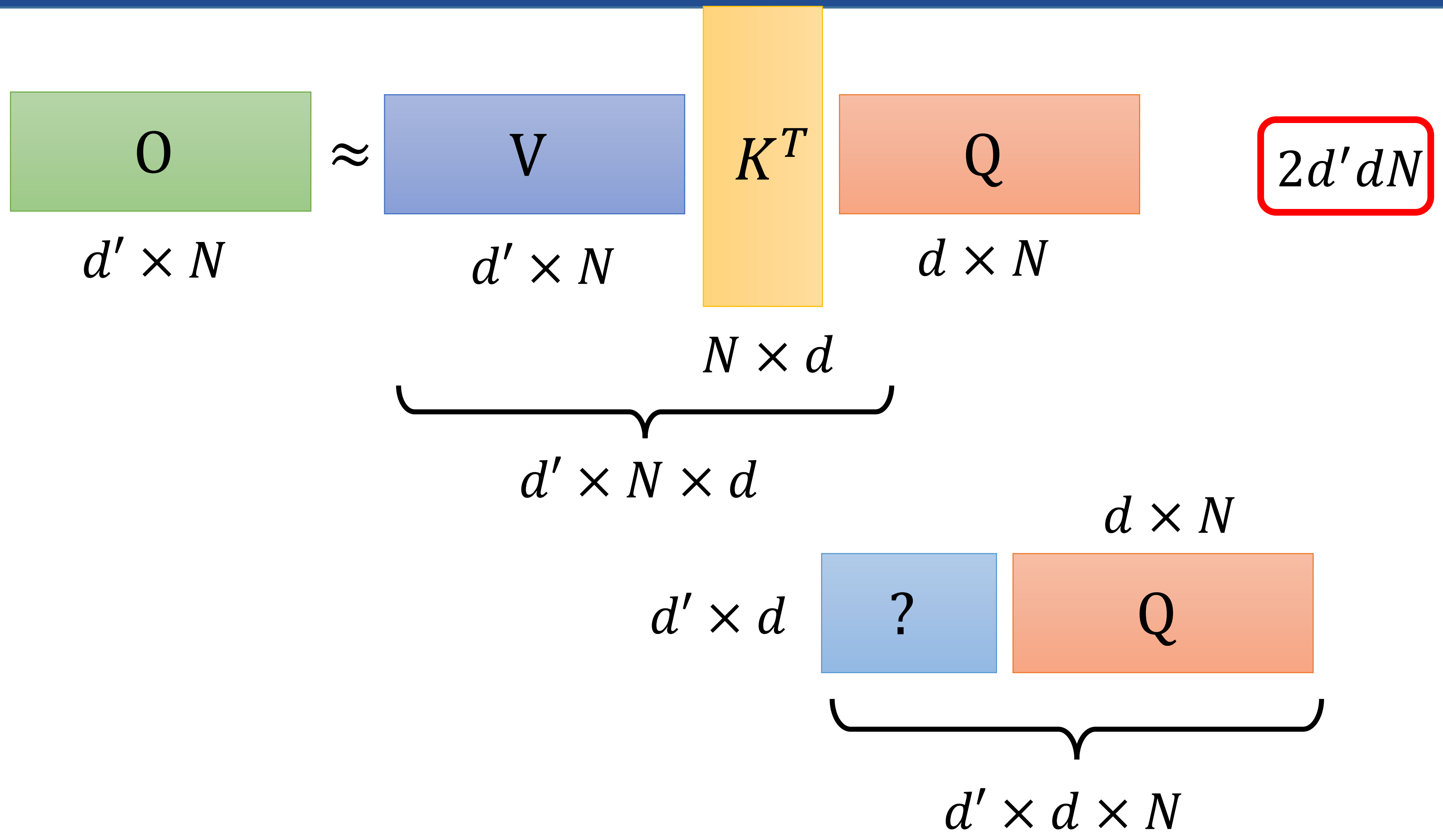
- Let A and B be k x m matrices, C be an m x n matrix, and P and Q be n x p matrices
  - A(CP) = (AC)P



Review Linear Algebra



Review Linear Algebra



Review Linear Algebra

$$\underset{d \times N}{\mathrm{O}} \approx \underbrace{\underset{d \times N}{\mathrm{V}} \; \underbrace{\underset{N \times d}{K^T} \; \underset{d \times N}{\mathrm{Q}}}} \qquad \boxed{(d + d')N^2}$$

$$\vee$$

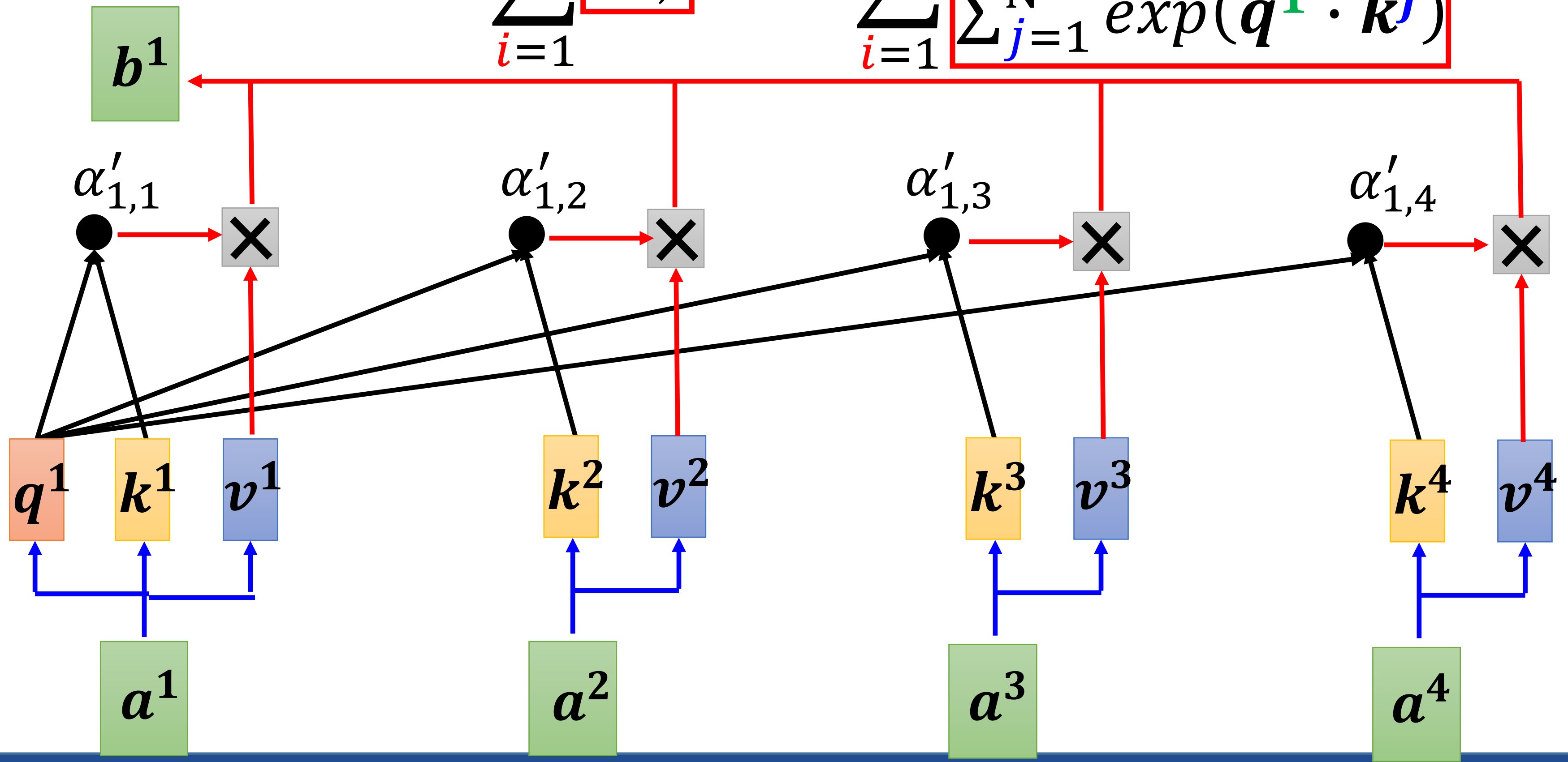
$$\underset{d \times N}{\mathrm{O}} \approx \underbrace{\underbrace{\underset{d \times N}{\mathrm{V}} \; \underset{N \times d}{K^T}} \; \underset{d \times N}{\mathrm{Q}}} \qquad \boxed{2d'dN}$$

Review Linear Algebra

Let’s put softmax back …
把Softmax激活函数放回来

Softmax

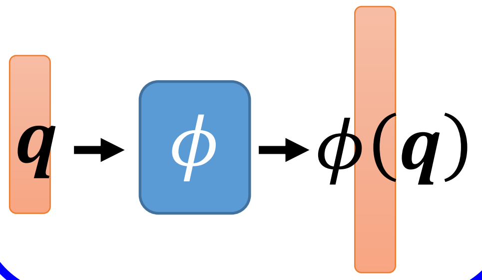
$$b^{1} = \sum_{i=1}^{N} \boxed{\alpha'_{1,i}} v^{i} = \sum_{i=1}^{N} \boxed{\frac{exp(q^{1} \cdot k^{i})}{\sum_{j=1}^{N} exp(q^{1} \cdot k^{j})}} v^{i}$$



Softmax

$$\boldsymbol{b}^1 = \sum_{i=1}^{N} \alpha'_{1,i} \boldsymbol{v}^i = \sum_{i=1}^{N} \frac{exp(\boldsymbol{q}^1 \cdot \boldsymbol{k}^i)}{\sum_{j=1}^{N} exp(\boldsymbol{q}^1 \cdot \boldsymbol{k}^j)} \boldsymbol{v}^i$$

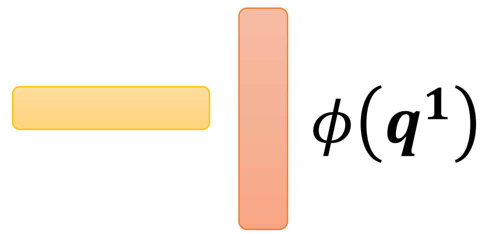
$$exp(\boldsymbol{q} \cdot \boldsymbol{k}) \approx \phi(\boldsymbol{q}) \cdot \phi(\boldsymbol{k})$$



$$= \sum_{i=1}^{N} \frac{\phi(\boldsymbol{q}^1) \cdot \phi(\boldsymbol{k}^i)}{\sum_{j=1}^{N} \phi(\boldsymbol{q}^1) \cdot \phi(\boldsymbol{k}^j)} \boldsymbol{v}^i$$

$$= \frac{\sum_{i=1}^{N} [\phi(\boldsymbol{q}^1) \cdot \phi(\boldsymbol{k}^i)] \boldsymbol{v}^i}{\boxed{\sum_{j=1}^{N} \phi(\boldsymbol{q}^1) \cdot \phi(\boldsymbol{k}^j)}}$$

$$\phi(\boldsymbol{q}^1) \cdot \boxed{\sum_{j=1}^{N} \phi(\boldsymbol{k}^j)}$$



Softmax

$$\boldsymbol{b}^1 = \sum_{i=1}^{N} \alpha'_{1,i}\boldsymbol{v}^i = \frac{\boxed{\sum_{i=1}^{N}[\phi(\boldsymbol{q}^1)\cdot\phi(\boldsymbol{k}^i)]\boldsymbol{v}^i}}{\phi(\boldsymbol{q}^1)\cdot\sum_{j=1}^{N}\phi(\boldsymbol{k}^j)}$$

$$\boxed{\sum_{i=1}^{N}[\phi(\boldsymbol{q}^1)\cdot\phi(\boldsymbol{k}^i)]\boldsymbol{v}^i}$$

$$\phi(\boldsymbol{q}^1) = \begin{bmatrix} q_1^1 \\ q_2^1 \\ \vdots \end{bmatrix} \qquad \phi(\boldsymbol{k}^1) = \begin{bmatrix} k_1^1 \\ k_2^1 \\ \vdots \end{bmatrix}$$

$$= [\phi(\boldsymbol{q}^1)\cdot\phi(\boldsymbol{k}^1)]\boldsymbol{v}^1 + [\phi(\boldsymbol{q}^1)\cdot\phi(\boldsymbol{k}^2)]\boldsymbol{v}^2 + \cdots$$

$$= (q_1^1 k_1^1 + q_2^1 k_2^1 + \cdots)\boldsymbol{v}^1 + (q_1^1 k_1^2 + q_2^1 k_2^2 + \cdots)\boldsymbol{v}^2 + \cdots$$

$$= \underline{q_1^1 k_1^1 \boldsymbol{v}^1} + \underline{q_2^1 k_2^1 \boldsymbol{v}^1} + \cdots + \underline{q_1^1 k_1^2 \boldsymbol{v}^2} + \underline{q_2^1 k_2^2 \boldsymbol{v}^2} + \cdots + \cdots$$

$$= q_1^1(k_1^1\boldsymbol{v}^1 + k_1^2\boldsymbol{v}^2 + \cdots) + q_2^1(k_2^1\boldsymbol{v}^1 + k_2^2\boldsymbol{v}^2 + \cdots)$$

Softmax

$$\boldsymbol{b}^1 = \sum_{i=1}^{N} \alpha'_{1,i} \boldsymbol{v}^i = \frac{\boxed{\sum_{i=1}^{N} [\phi(\boldsymbol{q}^1) \cdot \phi(\boldsymbol{k}^i)] \boldsymbol{v}^i}}{\phi(\boldsymbol{q}^1) \cdot \sum_{j=1}^{N} \phi(\boldsymbol{k}^j)}$$

$$\boxed{\sum_{i=1}^{N} [\phi(\boldsymbol{q}^1) \cdot \phi(\boldsymbol{k}^i)] \boldsymbol{v}^i}$$

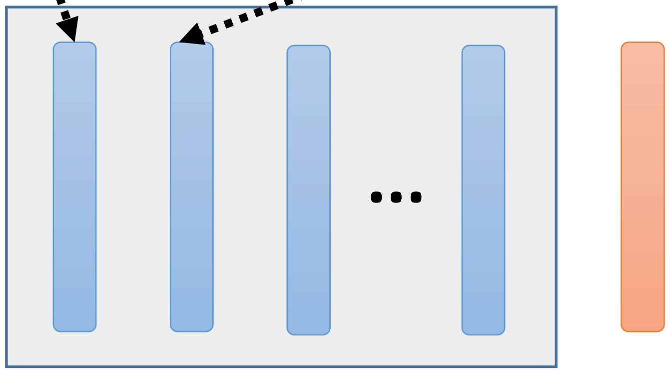
$$\phi(\boldsymbol{q}^1) = \begin{bmatrix} q_1^1 \\ q_2^1 \\ \vdots \end{bmatrix} \qquad \phi(\boldsymbol{k}^1) = \begin{bmatrix} k_1^1 \\ k_2^1 \\ \vdots \end{bmatrix}$$

M dim

$$= \boxed{q_1^1}(k_1^1 \boldsymbol{v}^1 + k_1^2 \boldsymbol{v}^2 + \cdots) + \boxed{q_2^1}(k_2^1 \boldsymbol{v}^1 + k_2^2 \boldsymbol{v}^2 + \cdots)$$

$$\sum_{j=1}^{N} k_1^j \boldsymbol{v}^j \qquad \sum_{j=1}^{N} k_2^j \boldsymbol{v}^j$$

M vectors



$\phi(\boldsymbol{q}^1)$

Softmax

$v^1 \quad v^2 \quad v^3 \quad \cdots \quad v^N$

$\phi(k^1) \quad \phi(k^2) \quad \phi(k^3) \quad \cdots\cdots \quad \phi(k^N)$

M dim

$\sum_{j=1}^{N} k_1^j v^j \qquad \sum_{j=1}^{N} k_2^j v^j$

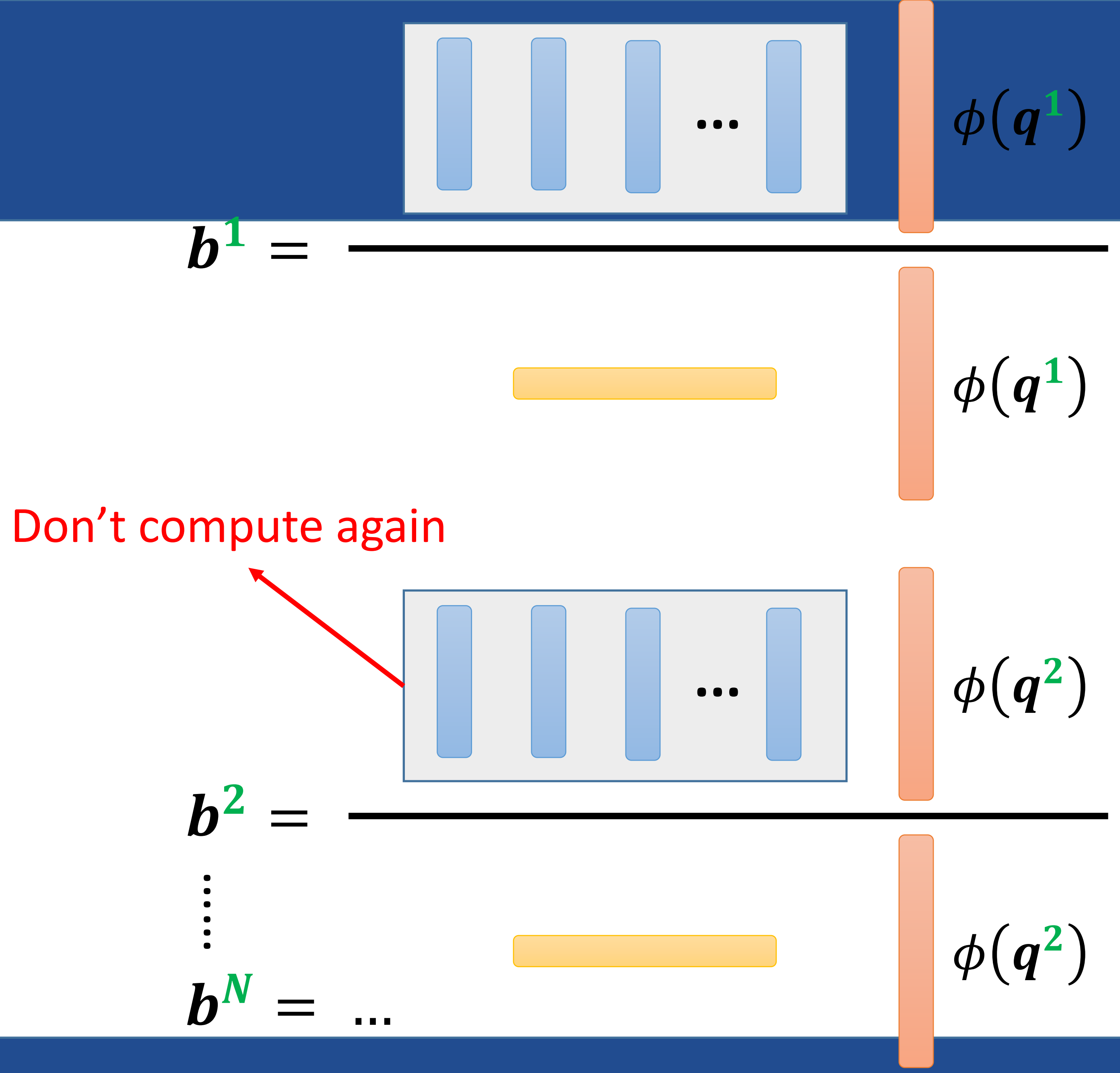
M vectors

$\phi(q^{\boxed{1}})$

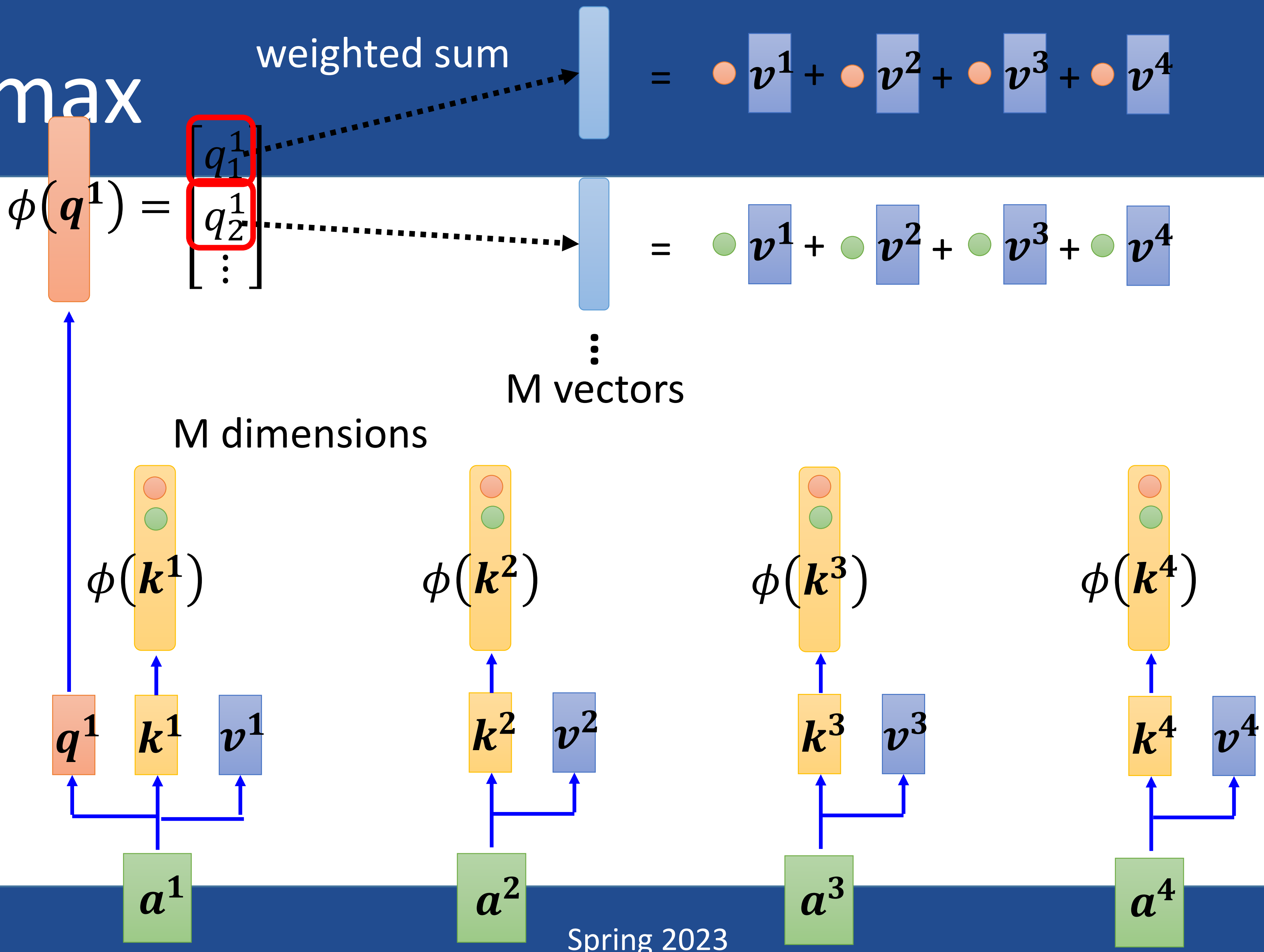
M dim

$$b^{\boxed{1}} = \frac{\left[\sum_{j=1}^{N} k_1^j v^j \quad \sum_{j=1}^{N} k_2^j v^j \quad \cdots \right] \phi(q^{\boxed{1}})}{\sum_{j=1}^{N} \phi(k^j) \, \phi(q^{\boxed{1}})}$$

Softmax


$\boldsymbol{b}^1 =$
$\phi(\boldsymbol{q}^1)$
$\phi(\boldsymbol{q}^1)$
Don't compute again
$\phi(\boldsymbol{q}^2)$
$\boldsymbol{b}^2 =$
$\phi(\boldsymbol{q}^2)$
$\boldsymbol{b}^N = \dots$

Softmax



weighted sum
$\phi(\boldsymbol{q}^1) = \begin{bmatrix} q_1^1 \\ q_2^1 \\ \vdots \end{bmatrix}$
$= \boldsymbol{v}^1 + \boldsymbol{v}^2 + \boldsymbol{v}^3 + \boldsymbol{v}^4$
$= \boldsymbol{v}^1 + \boldsymbol{v}^2 + \boldsymbol{v}^3 + \boldsymbol{v}^4$
M vectors
M dimensions
$\phi(\boldsymbol{k}^1)$
$\phi(\boldsymbol{k}^2)$
$\phi(\boldsymbol{k}^3)$
$\phi(\boldsymbol{k}^4)$
$\boldsymbol{q}^1$
$\boldsymbol{k}^1$
$\boldsymbol{v}^1$
$\boldsymbol{k}^2$
$\boldsymbol{v}^2$
$\boldsymbol{k}^3$
$\boldsymbol{v}^3$
$\boldsymbol{k}^4$
$\boldsymbol{v}^4$
$\boldsymbol{a}^1$
$\boldsymbol{a}^2$
$\boldsymbol{a}^3$
$\boldsymbol{a}^4$

Softmax

$$\phi(\boldsymbol{q}^1) = \begin{bmatrix} \boxed{q_1^1} \\ \boxed{q_2^1} \\ \vdots \end{bmatrix}$$

weighted sum

$$= \boldsymbol{v}^1 + \boldsymbol{v}^2 + \boldsymbol{v}^3 + \boldsymbol{v}^4$$

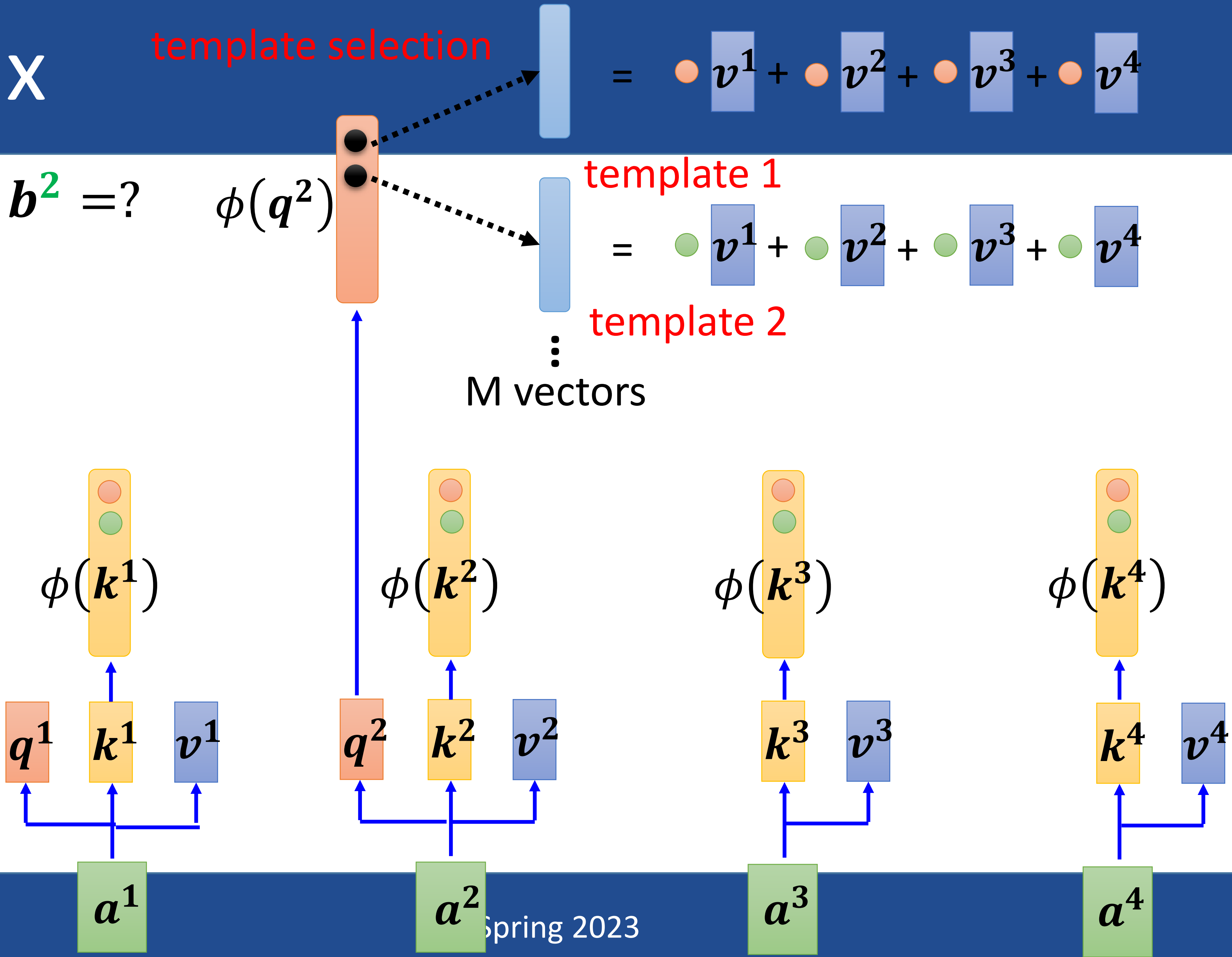
$$= \boldsymbol{v}^1 + \boldsymbol{v}^2 + \boldsymbol{v}^3 + \boldsymbol{v}^4$$

$\vdots$

M vectors

$$\boldsymbol{b}^1 = \frac{\cdots \quad \phi(\boldsymbol{q}^1)}{\sum_{j=1}^{N} \phi(\boldsymbol{k}^j) \quad \phi(\boldsymbol{q}^1)}$$

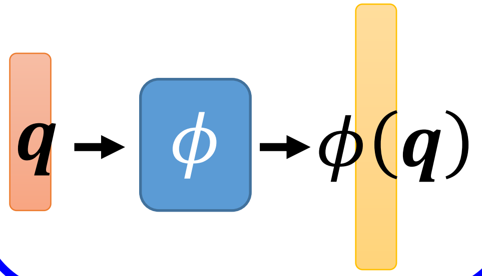
Softmax



template selection
template 1
template 2
M vectors
$b^2 = ?$
$\phi(q^2)$
$= \bullet v^1 + \bullet v^2 + \bullet v^3 + \bullet v^4$
$= \bullet v^1 + \bullet v^2 + \bullet v^3 + \bullet v^4$
$\phi(k^1)$
$\phi(k^2)$
$\phi(k^3)$
$\phi(k^4)$
q^1 k^1 v^1
q^2 k^2 v^2
k^3 v^3
k^4 v^4
a^1
a^2
a^3
a^4

Realization

$$exp(\boldsymbol{q} \cdot \boldsymbol{k}) \approx \phi(\boldsymbol{q}) \cdot \phi(\boldsymbol{k})$$



- Efficient attention

  https://arxiv.org/pdf/1812.01243.pdf

- Linear Transformer

  https://linear-transformers.com/

- Random Feature Attention

  https://arxiv.org/pdf/2103.02143.pdf

- Performer

  https://arxiv.org/pdf/2009.14794.pdf

Do we need q and k to compute attention? Synthesizer!

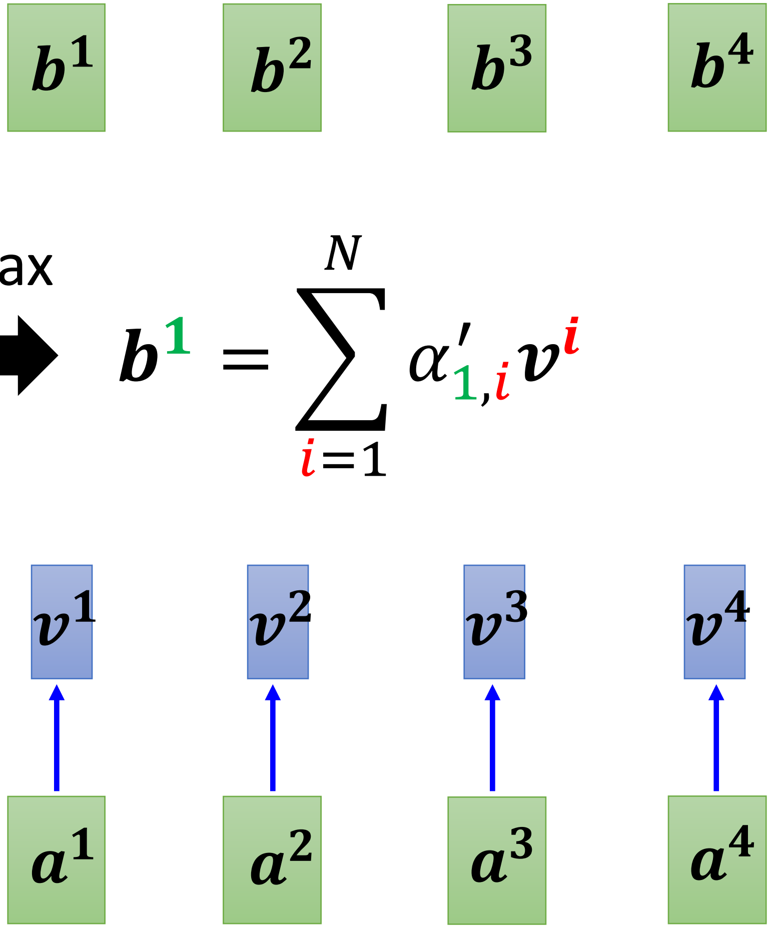
$\alpha_{1,1}$	$\alpha_{1,2}$	$\alpha_{1,3}$	$\alpha_{1,4}$
$\alpha_{1,2}$	$\alpha_{2,2}$	$\alpha_{2,3}$	$\alpha_{2,4}$
$\alpha_{1,3}$	$\alpha_{2,3}$	$\alpha_{3,3}$	$\alpha_{3,4}$
$\alpha_{1,4}$	$\alpha_{2,4}$	$\alpha_{3,4}$	$\alpha_{4,4}$

~~From q and k?~~

They are network parameters!

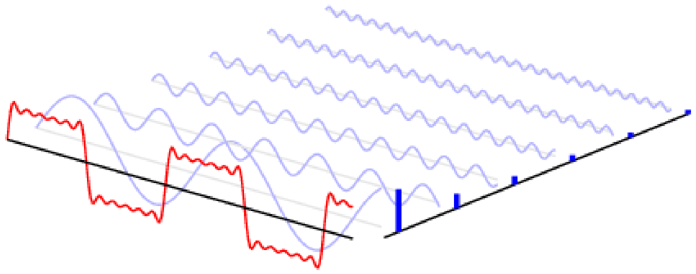
softmax

$$\boldsymbol{b}^1 = \sum_{i=1}^{N} \alpha'_{1,i} \boldsymbol{v}^i$$



https://arxiv.org/abs/2005.00743

Attention-Free



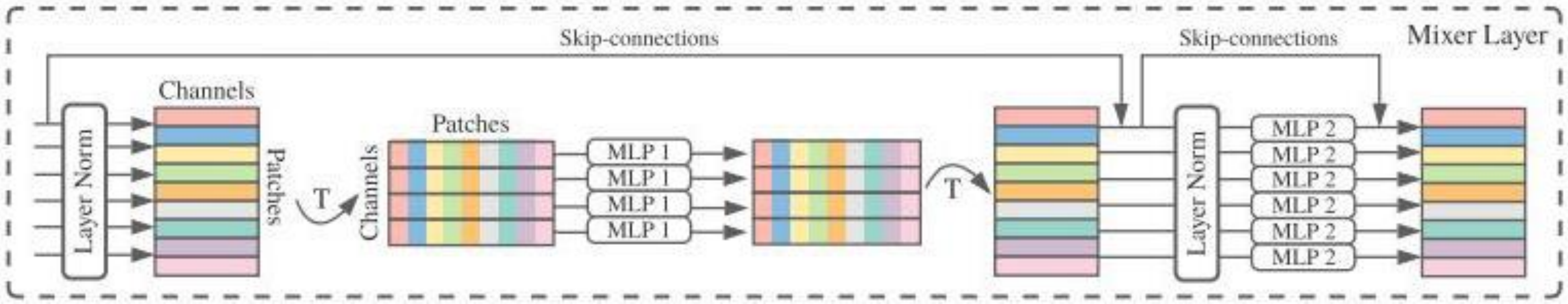
- Fnet: Mixing tokens with fourier transforms

https://arxiv.org/abs/2105.03824

- Pay Attention to MLPs https://arxiv.org/abs/2105.08050

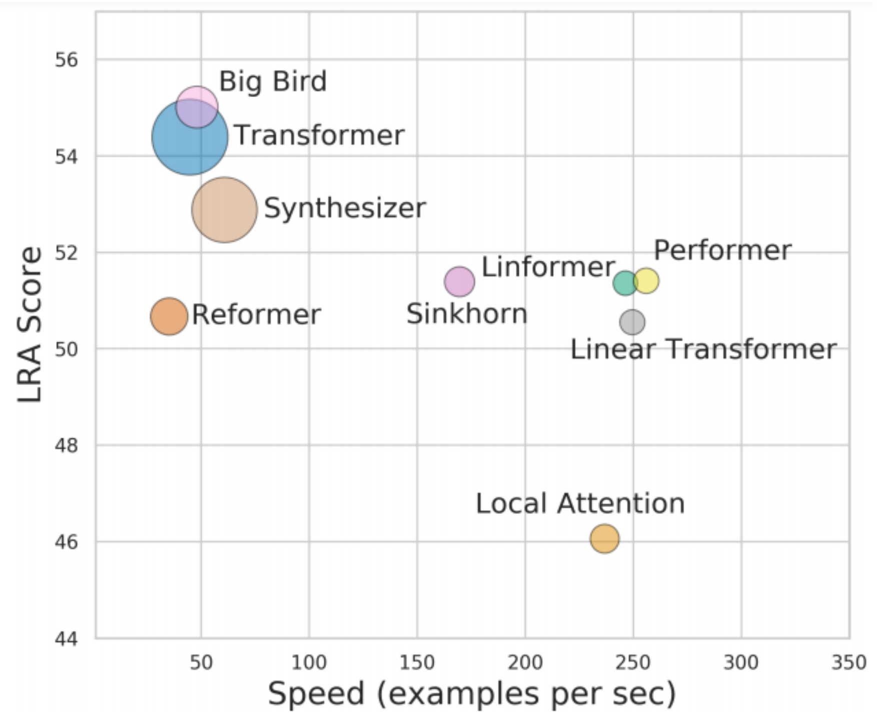
- MLP-Mixer: An all-MLP Architecture for Vision

https://arxiv.org/abs/2105.01601



Summary

- Human knowledge
  - Local Attention, Big Bird
- Clustering
  - Reformer
- Learnable Pattern
  - SinFform
- Representative key
  - LinFormer
- k,q first → v,k first
  - Linear Transformer, Performer
- New framework
  - Synthesizer



Q&A