Self-Attention, Graph Networks, Transformer

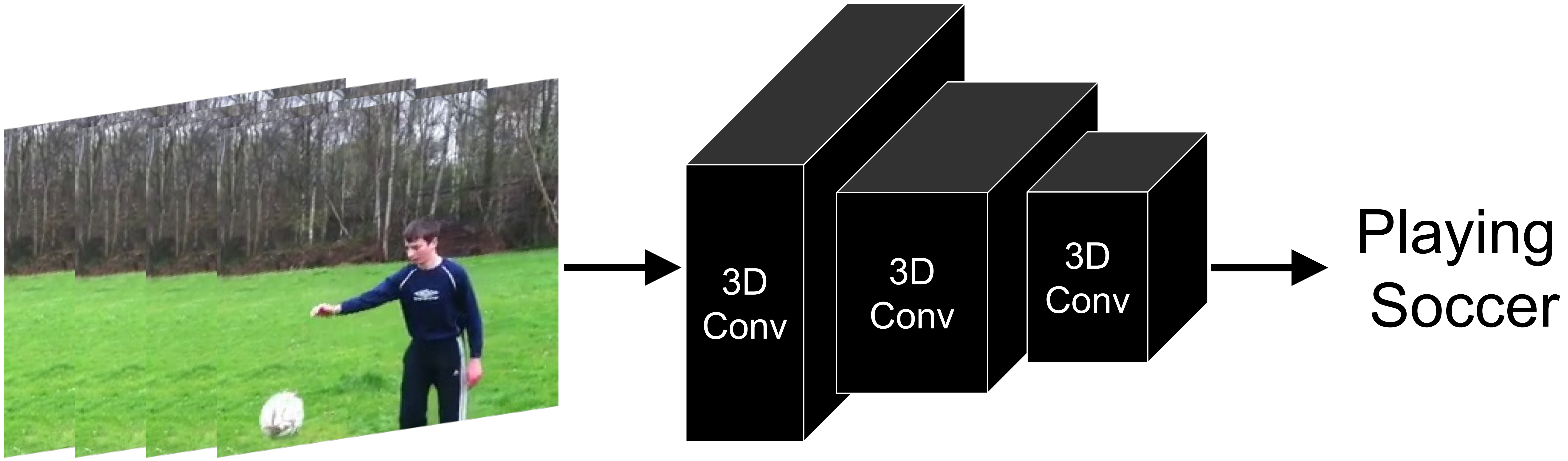
Xiaolong Wang

This Class

- Non-local Neural Network for Videos
- Self-Attention and Transformer for NLP
- Graph Neural Networks

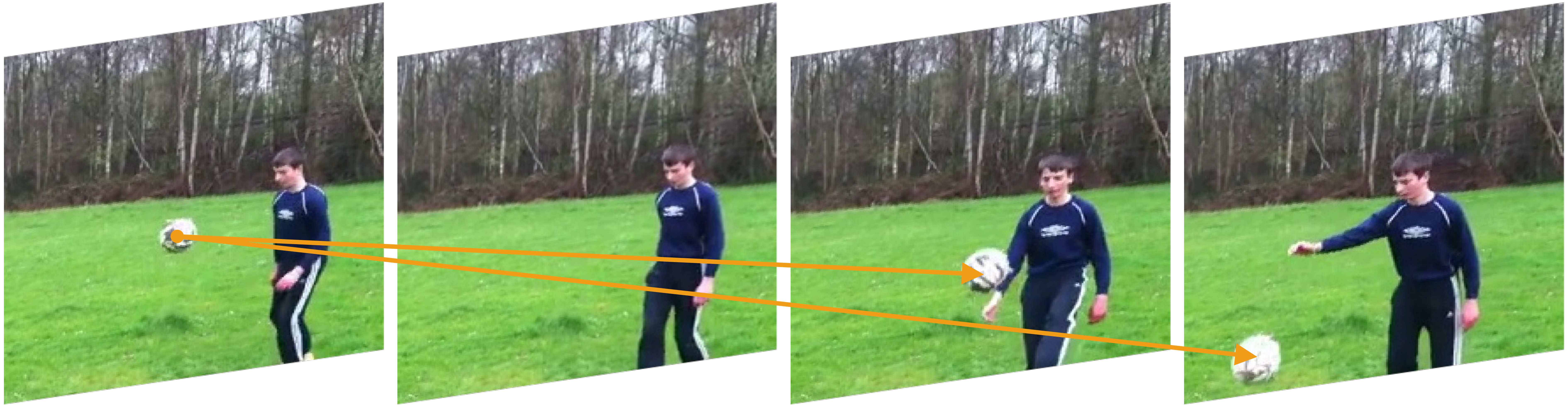
Non-local Neural Network for Videos

Video Recognition



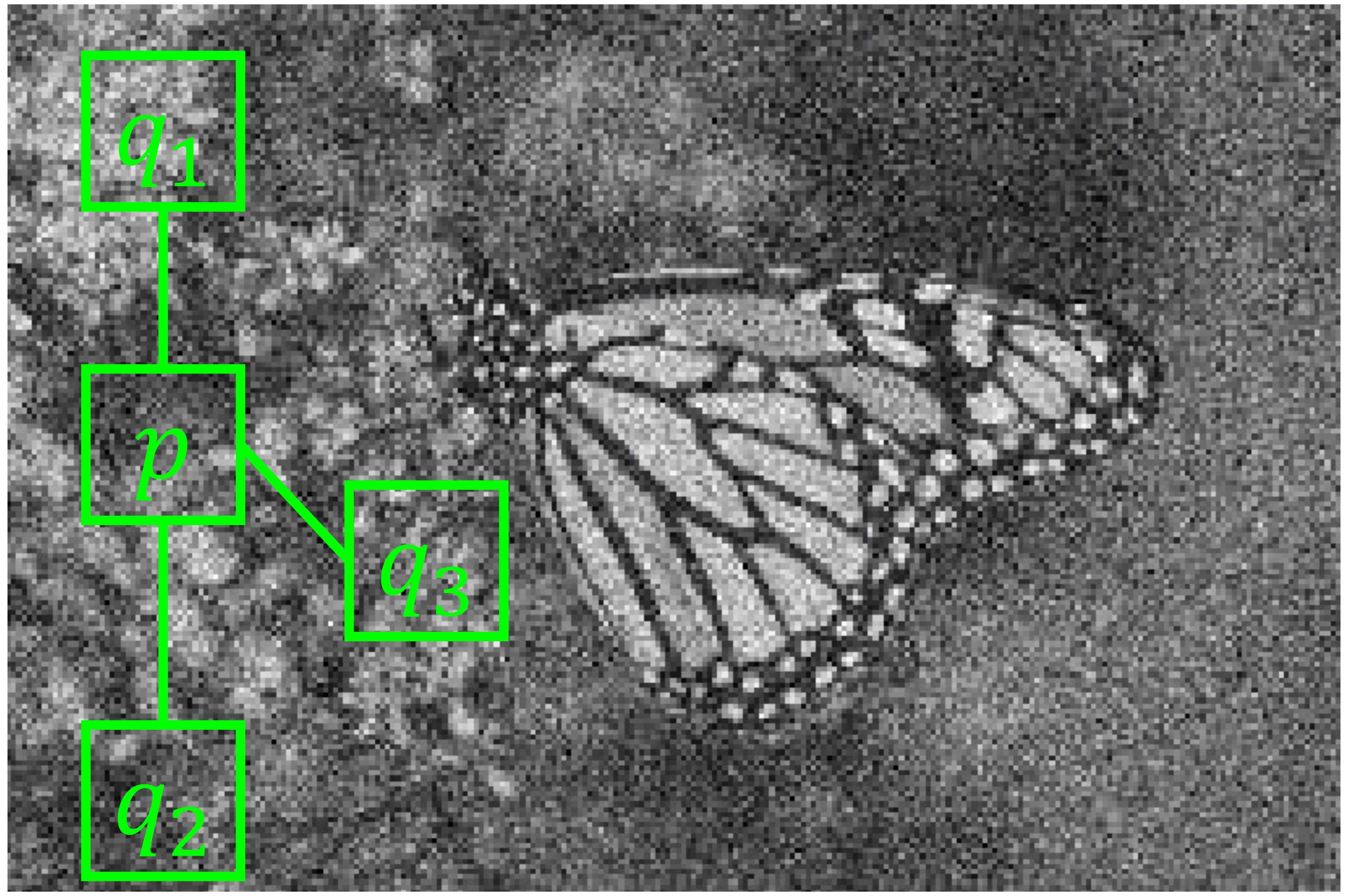
Reasoning for Action Recognition

Long-rang explicit reasoning



Wang et al., 2018.

Non-local Means

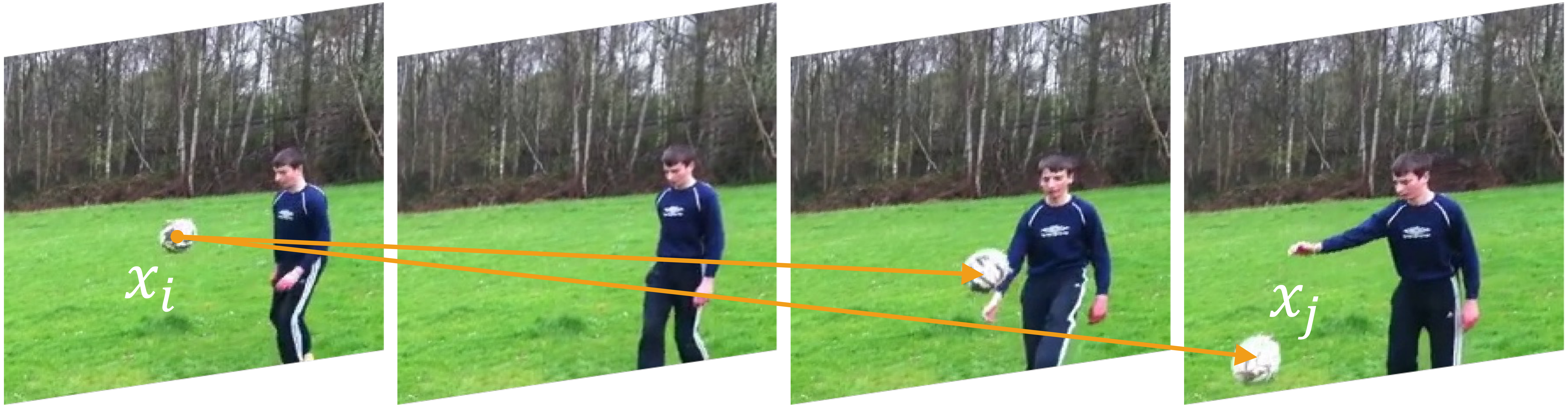


Buades et al., 2005.

Non-local Operator

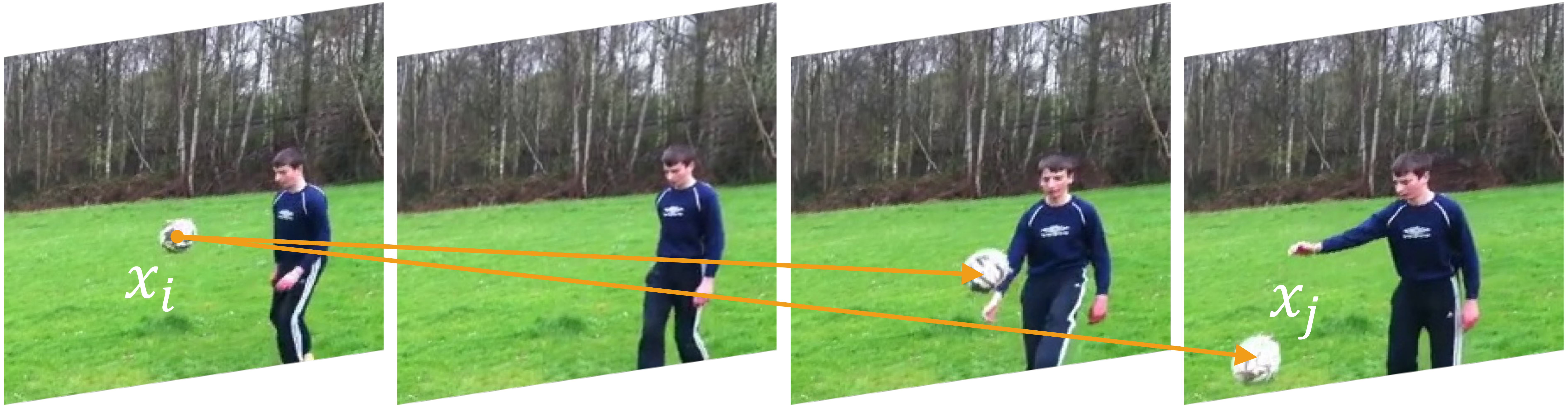
Operation in feature space

Can be embedded into any ConvNets



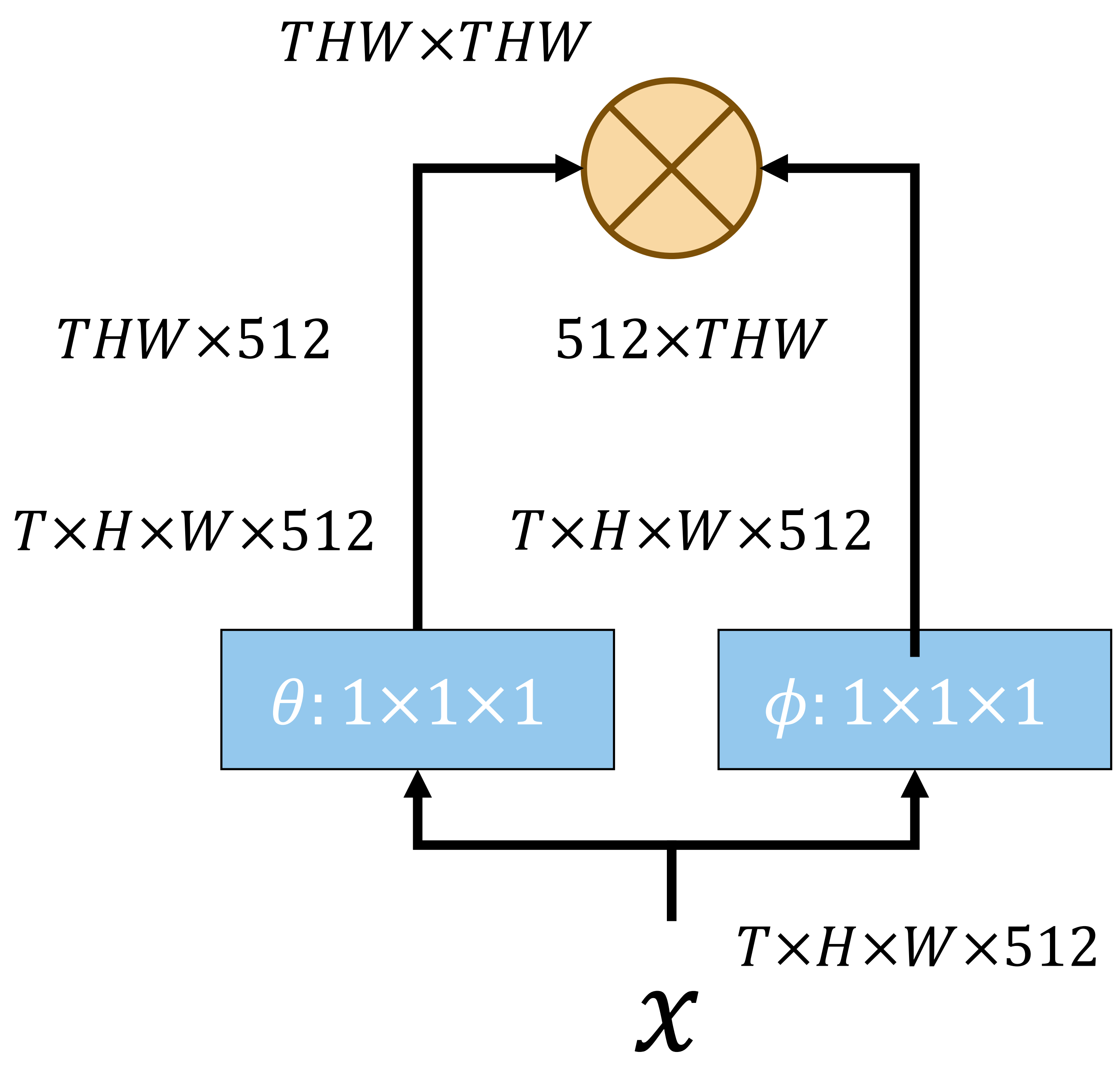
Non-local Operator

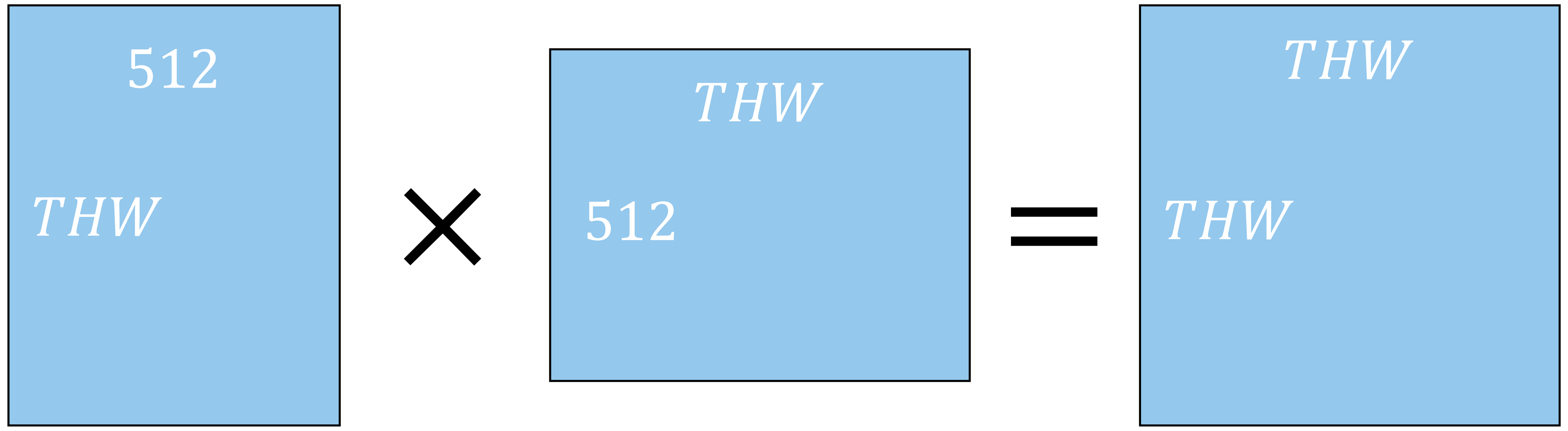
$$y_i = \frac{1}{C(x)} \sum_{\forall j} \underbrace{f(x_i, x_j)}_{\text{Affinity}} \; \underbrace{g(x_j)}_{\text{Features}}$$



Non-local Operator

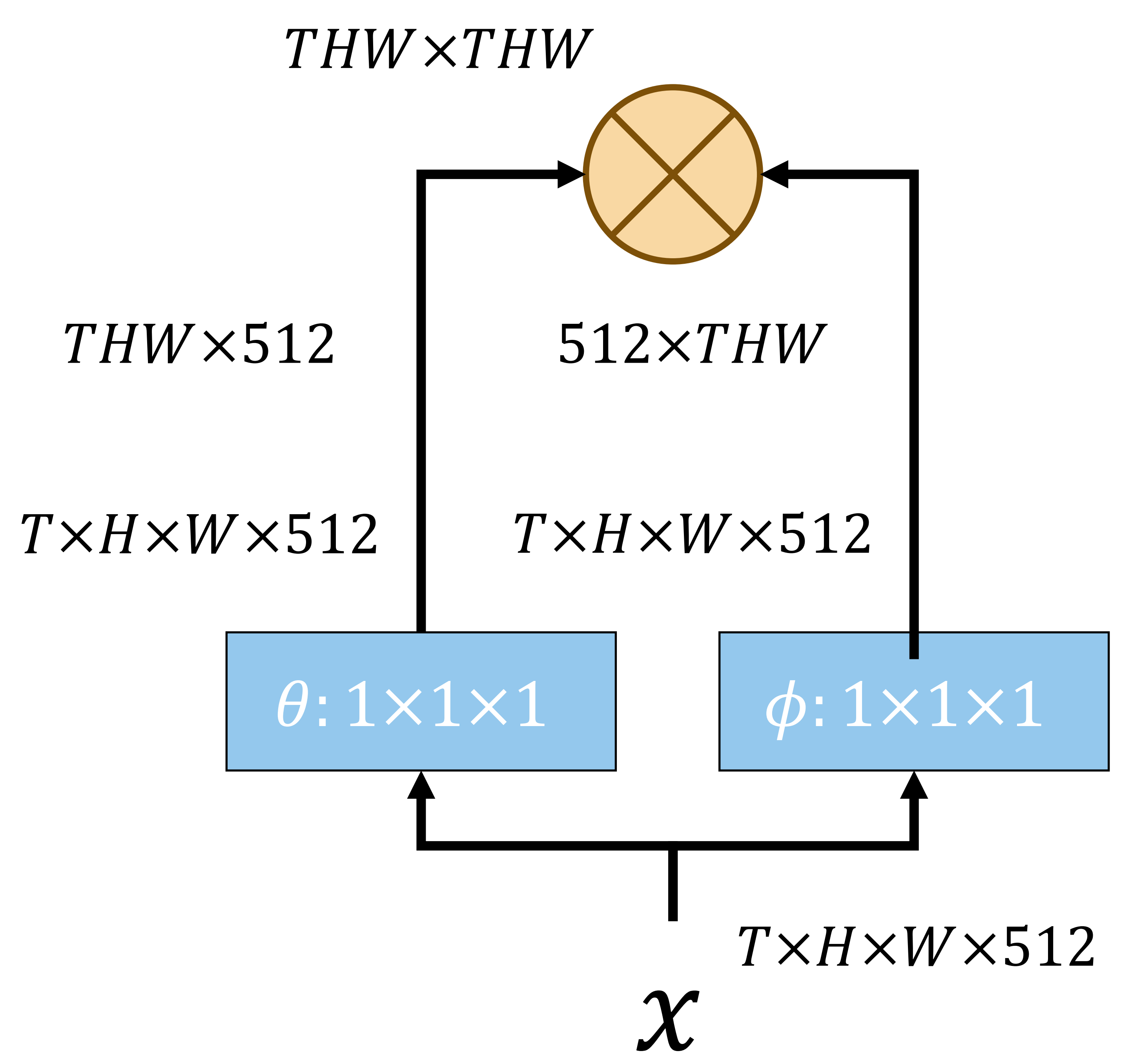
$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j)\ g(x_j)$$

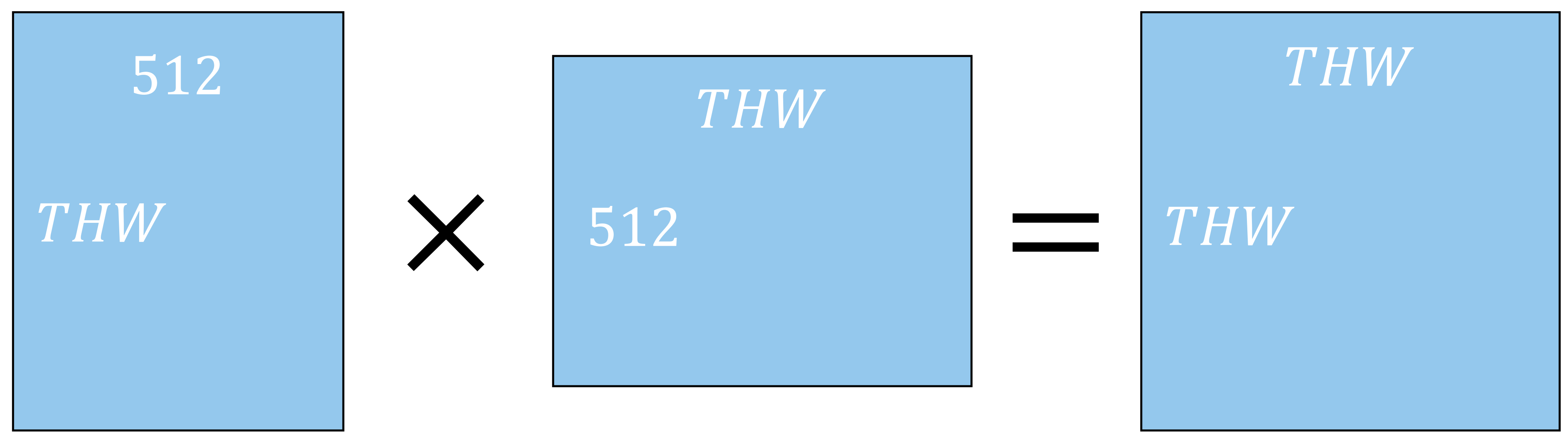




Non-local Operator

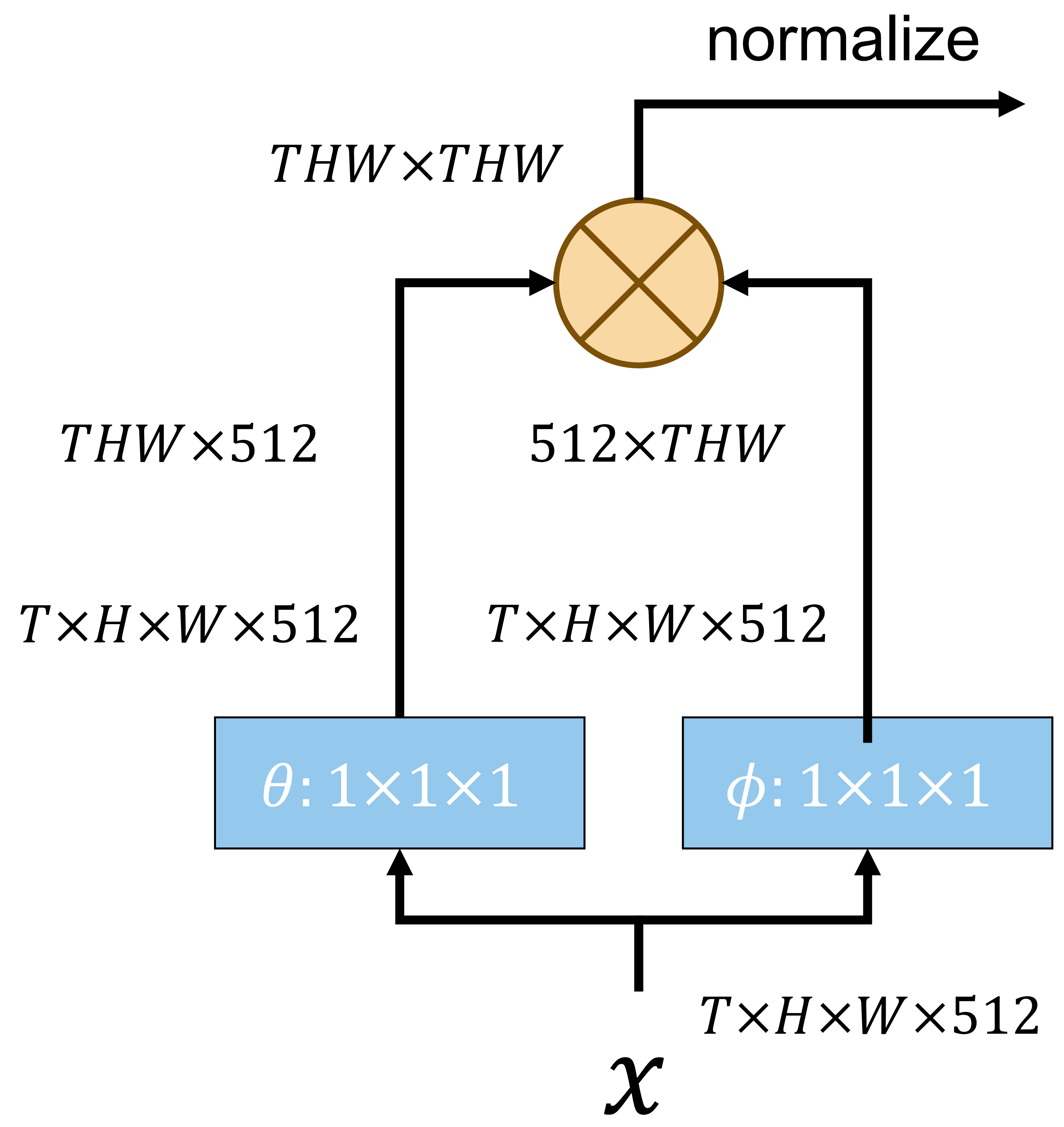
$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j)\ g(x_j)$$





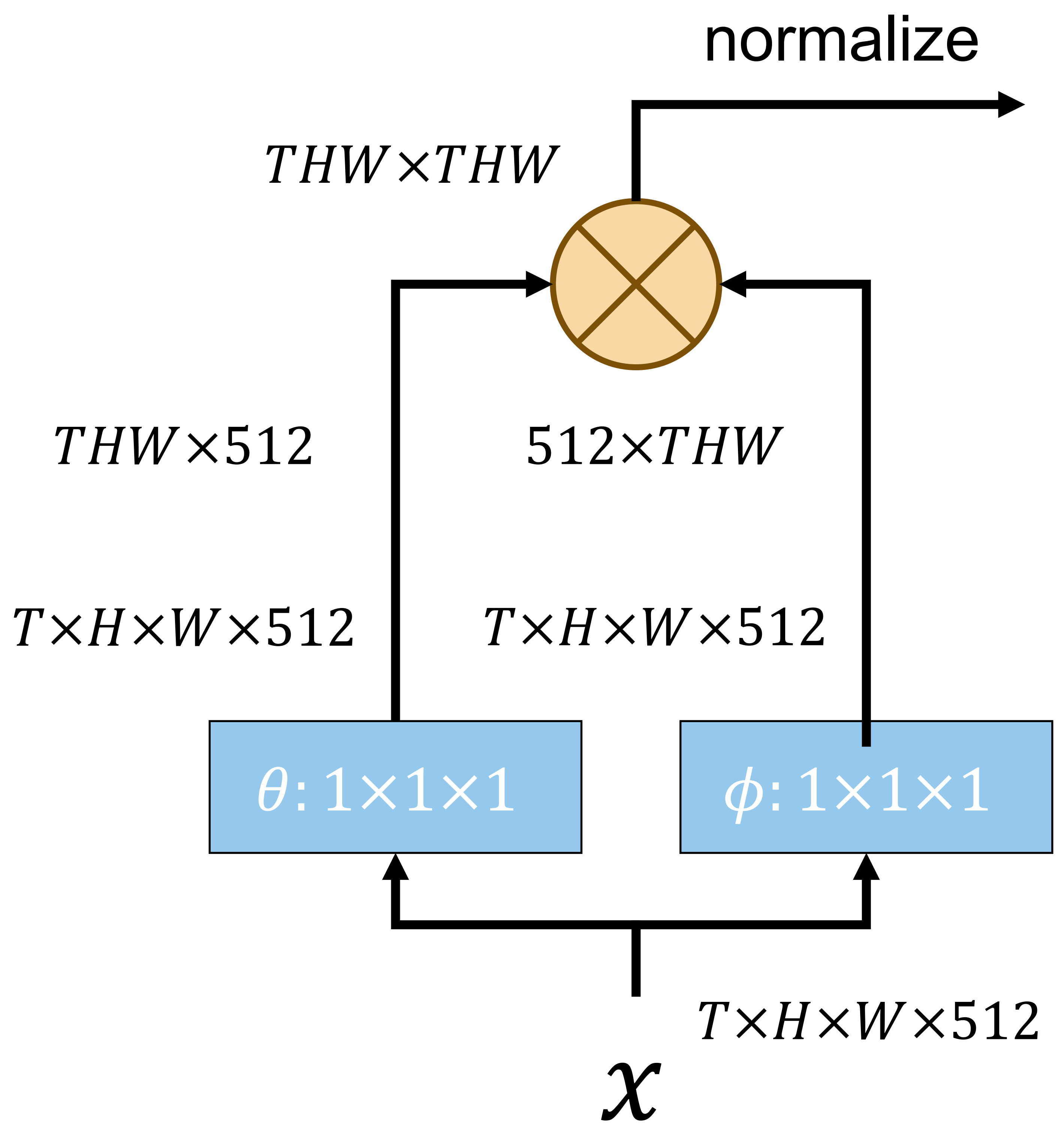
Non-local Operator

$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j)\ g(x_j)$$



Non-local Operator

$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j)\ g(x_j)$$



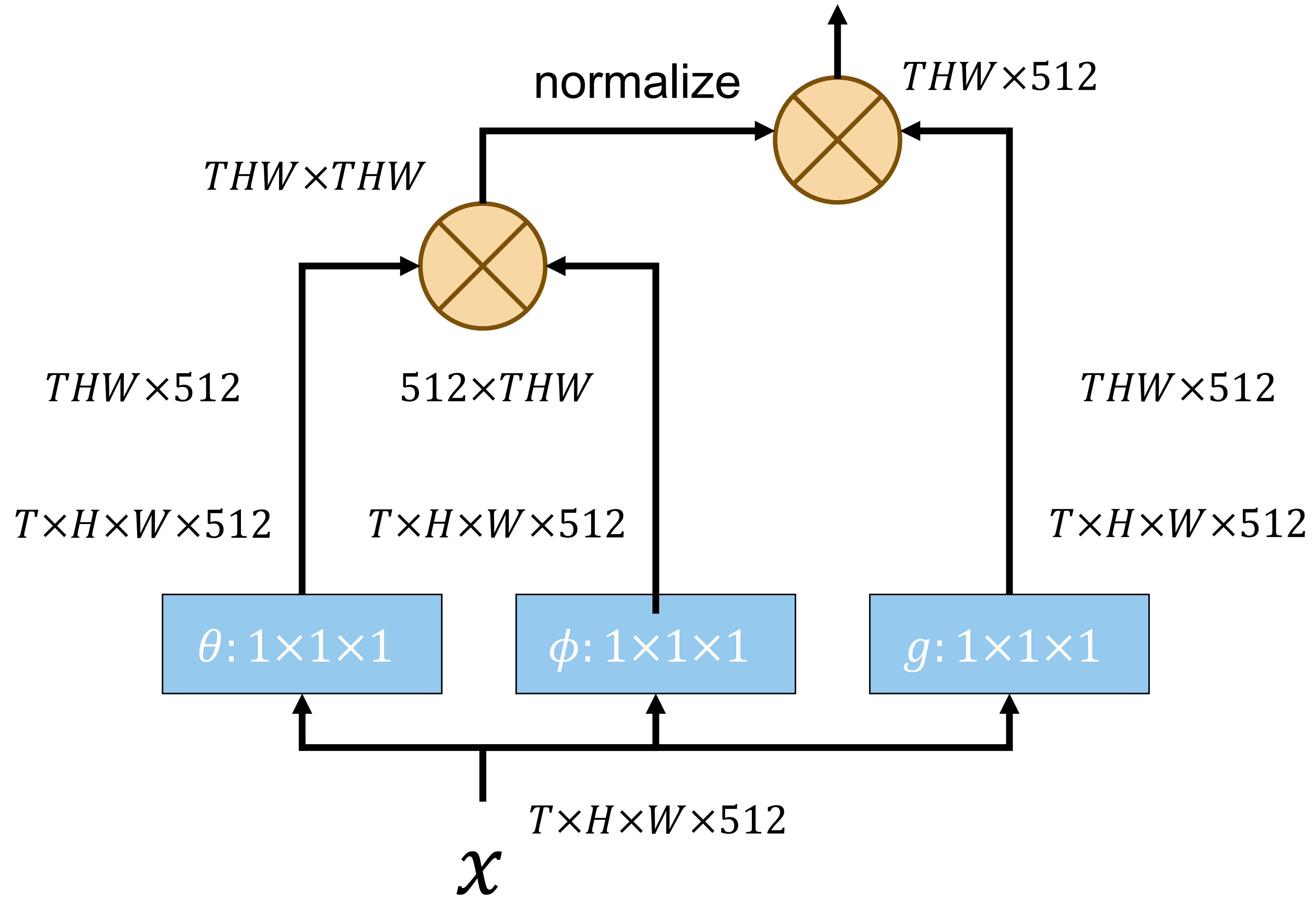
$$f(x_i, x_j) = \exp(x_i^T x_j)$$

$$C(x) = \sum_{\forall j} f(x_i, x_j)$$

$$\frac{f(x_i, x_j)}{C(x)} = \frac{\exp(x_i^T x_j)}{\sum_{\forall j} \exp(x_i^T x_j)}$$

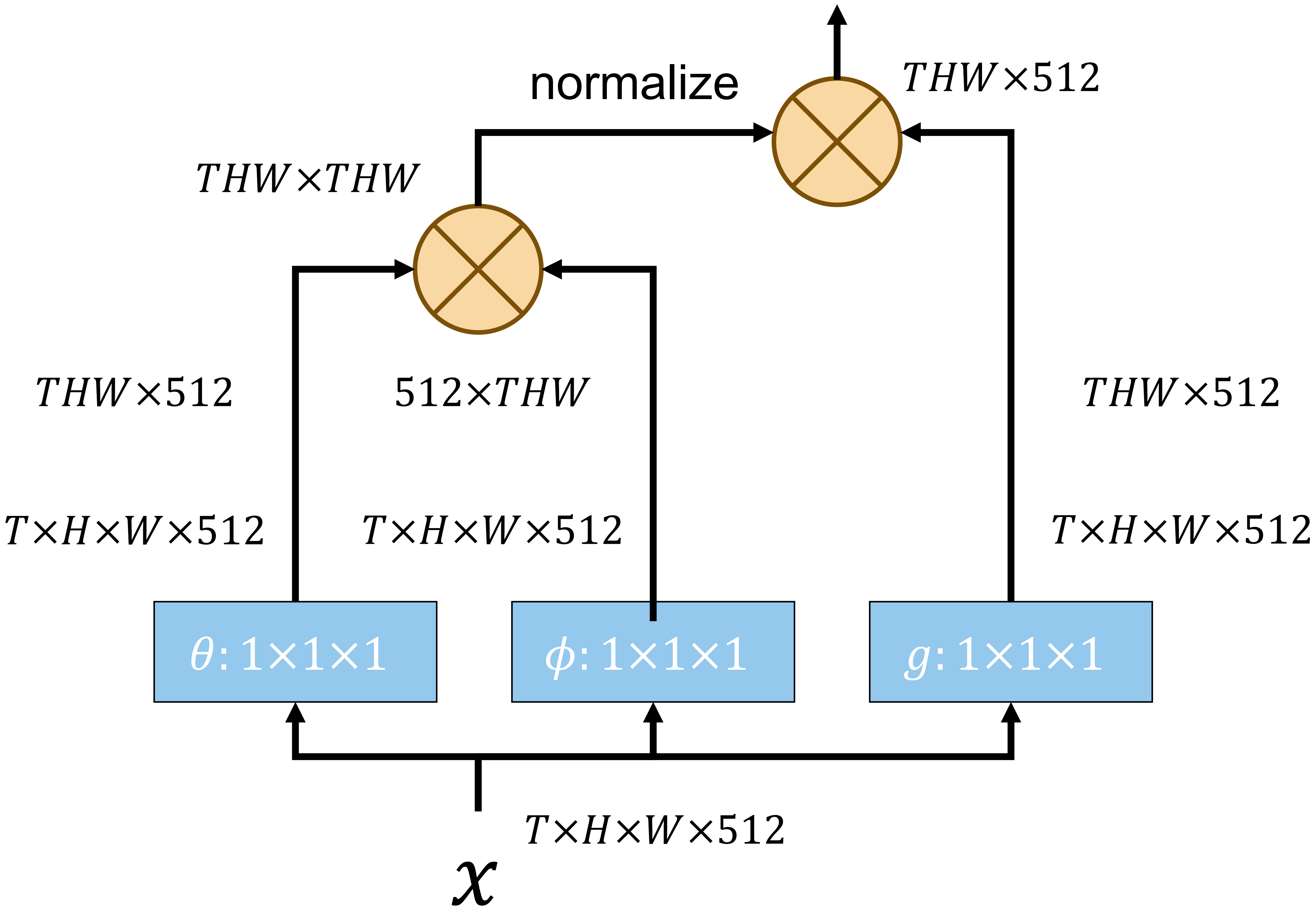
Non-local Operator

$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j)\ g(x_j)$$



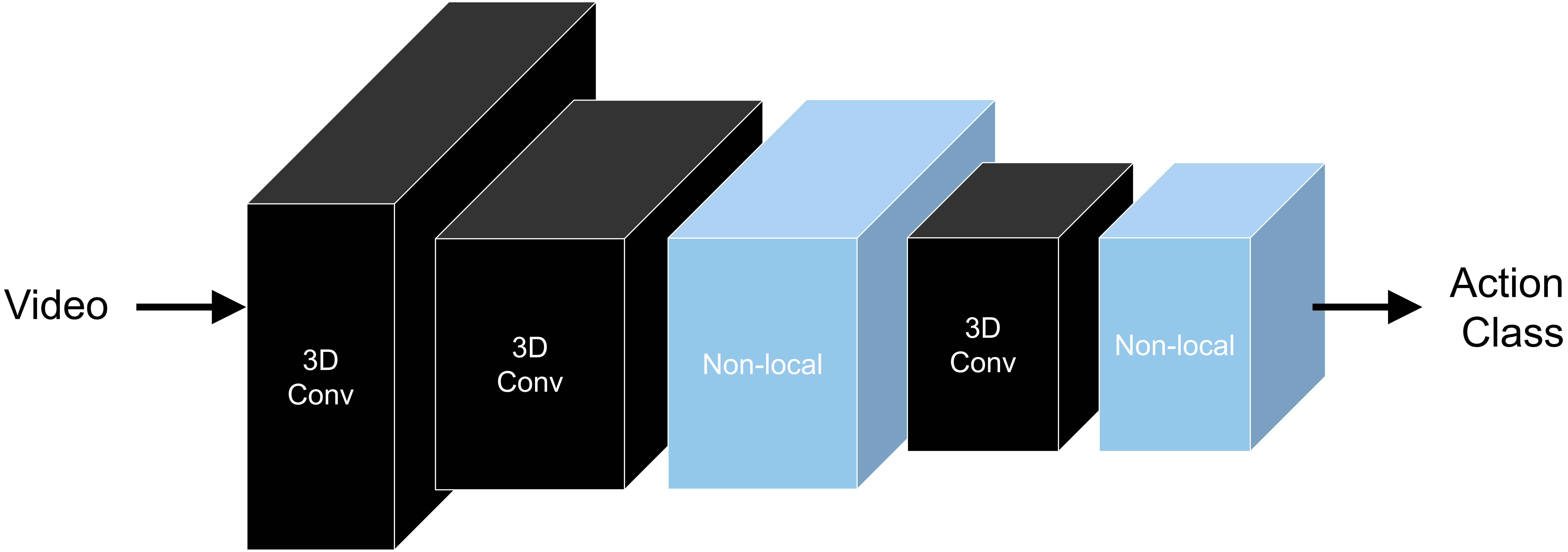
Non-local Operator

$$y_i = \frac{1}{C(x)} \sum_{\forall j} f\left(x_i, x_j\right)\ g(x_j)$$

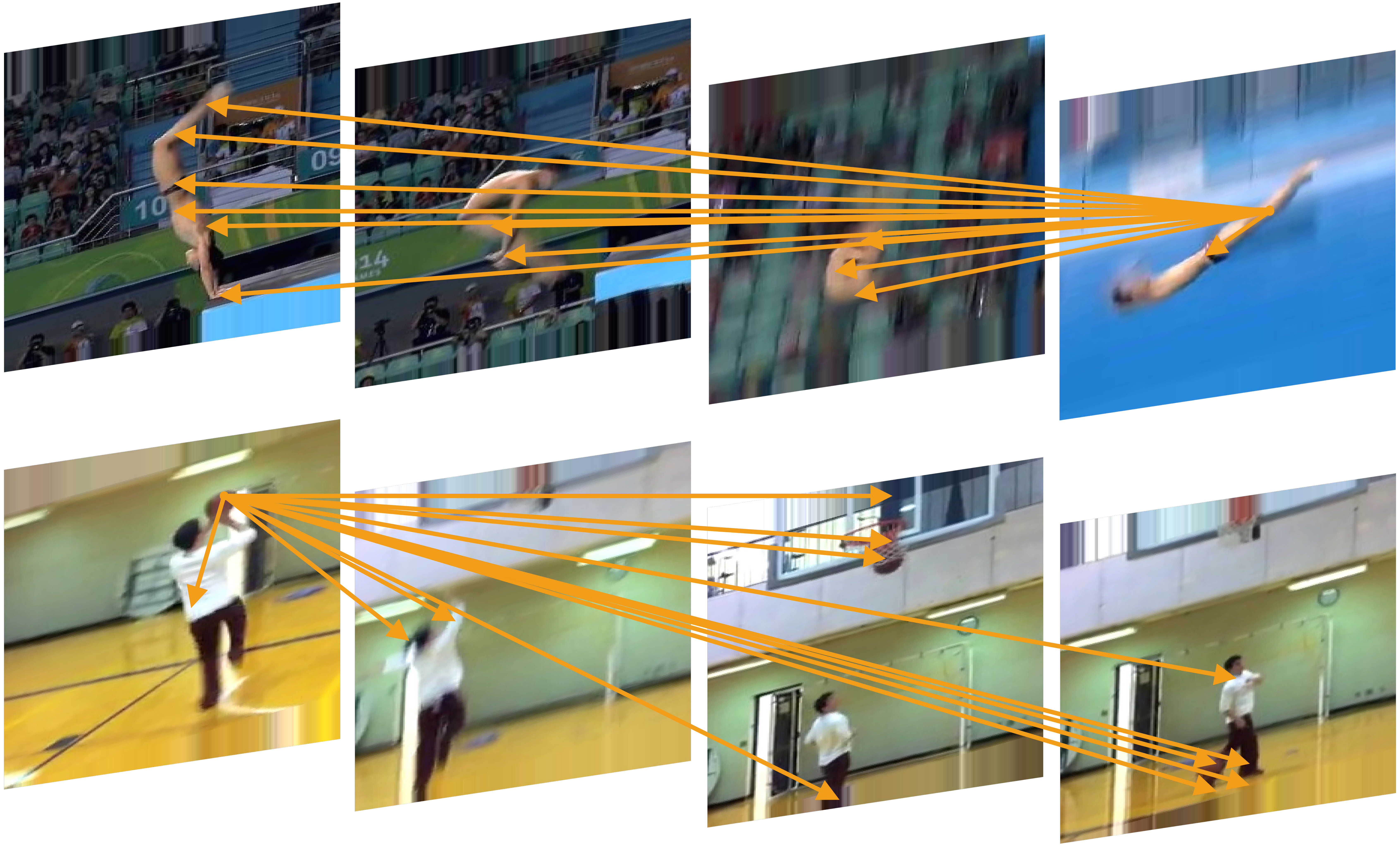


Non-local Operator as A Residual Block

$$z_i = y_i W + x_i$$



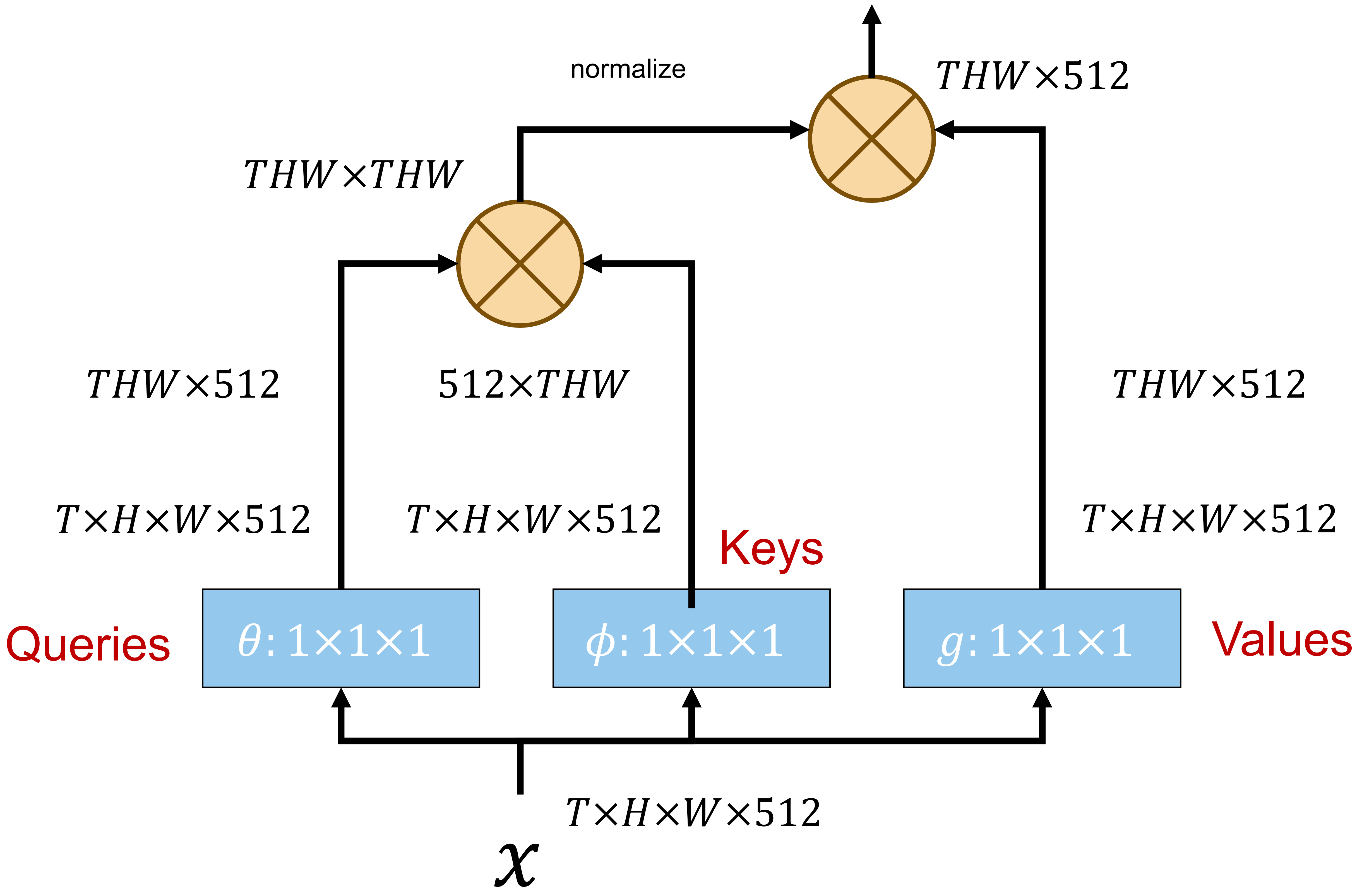
Examples



Self-Attention and Transformer for NLP

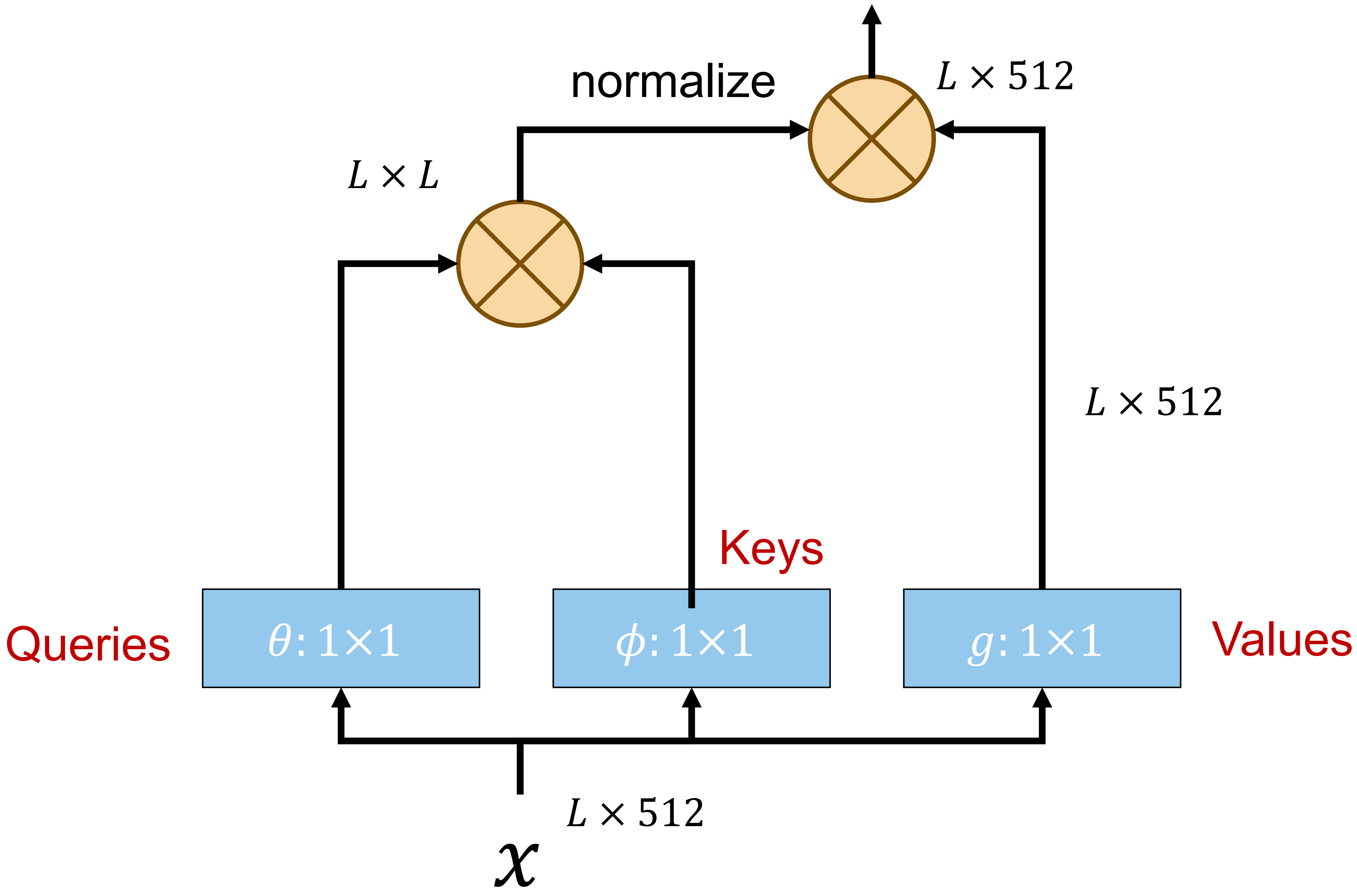
Self-Attention Operator

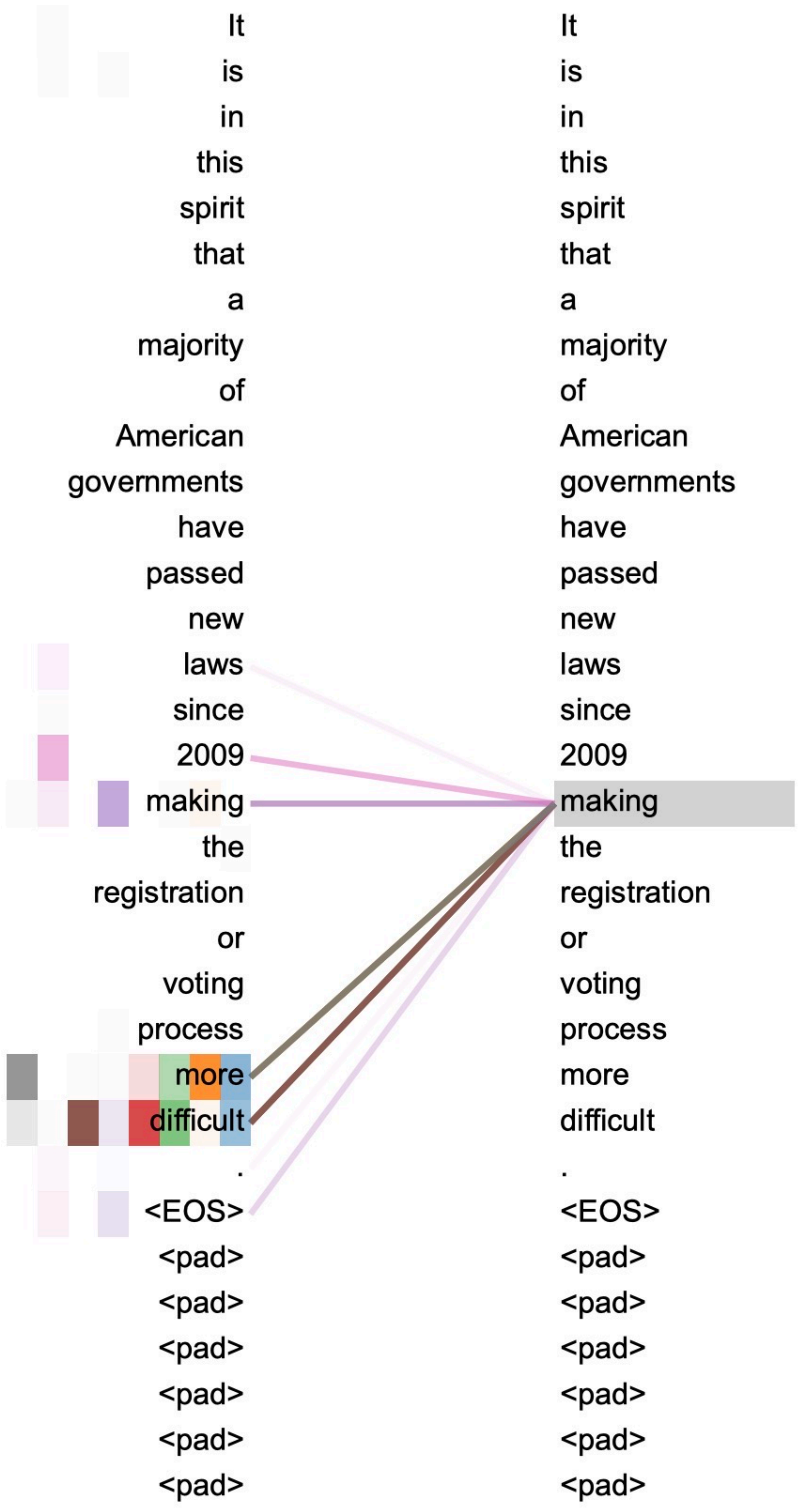
$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j)\ g(x_j)$$



Self-Attention Operator

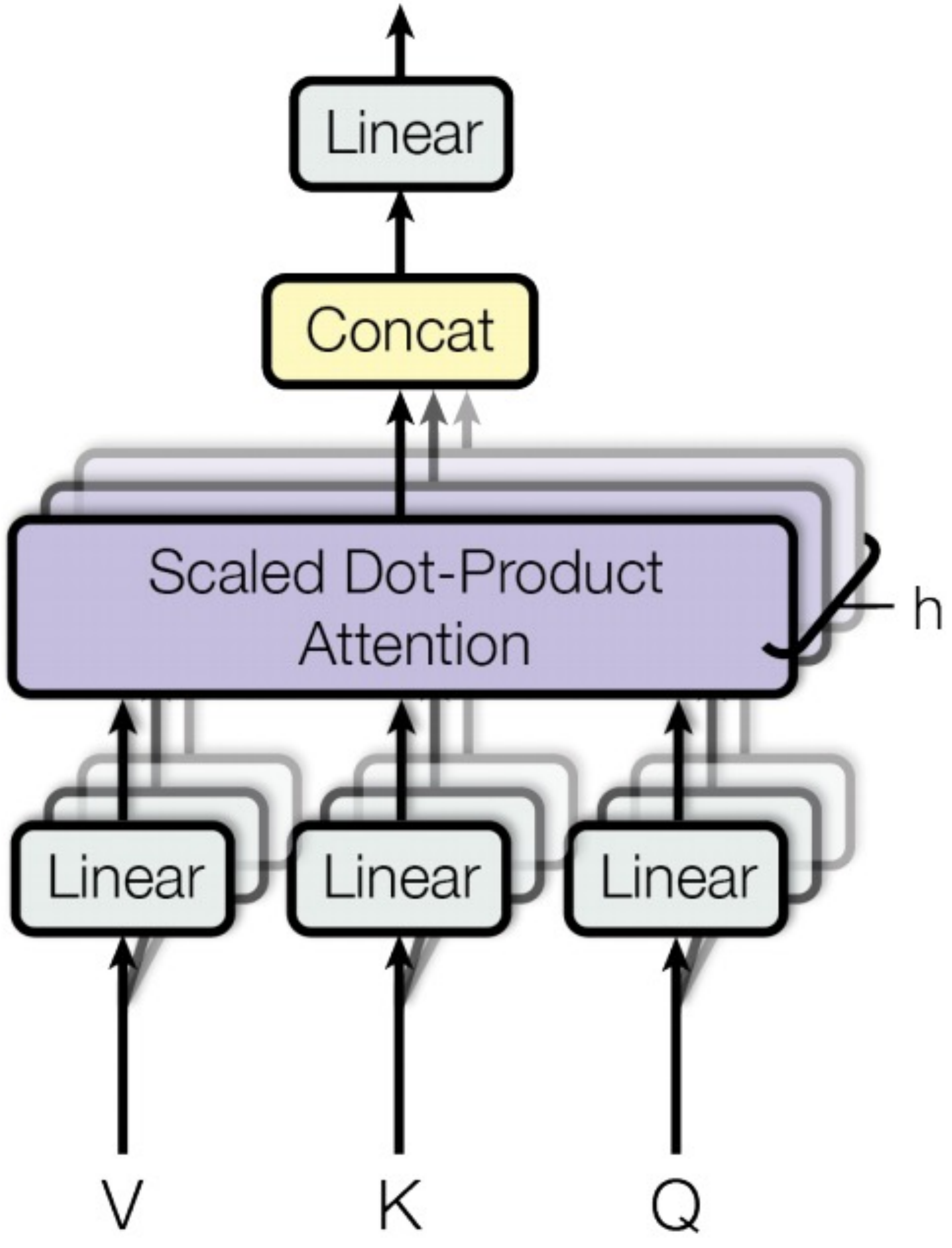
$$y_i = \frac{1}{C(x)} \sum_{\forall j} f\left(x_i, x_j\right)\ g(x_j)$$




It
is
in
this
spirit
that
a
majority
of
American
governments
have
passed
new
laws
since
2009
making
the
registration
or
voting
process
more
difficult
.
<EOS>
<pad>
<pad>
<pad>
<pad>
<pad>
<pad>
It
is
in
this
spirit
that
a
majority
of
American
governments
have
passed
new
laws
since
2009
making
the
registration
or
voting
process
more
difficult
.
<EOS>
<pad>
<pad>
<pad>
<pad>
<pad>
<pad>

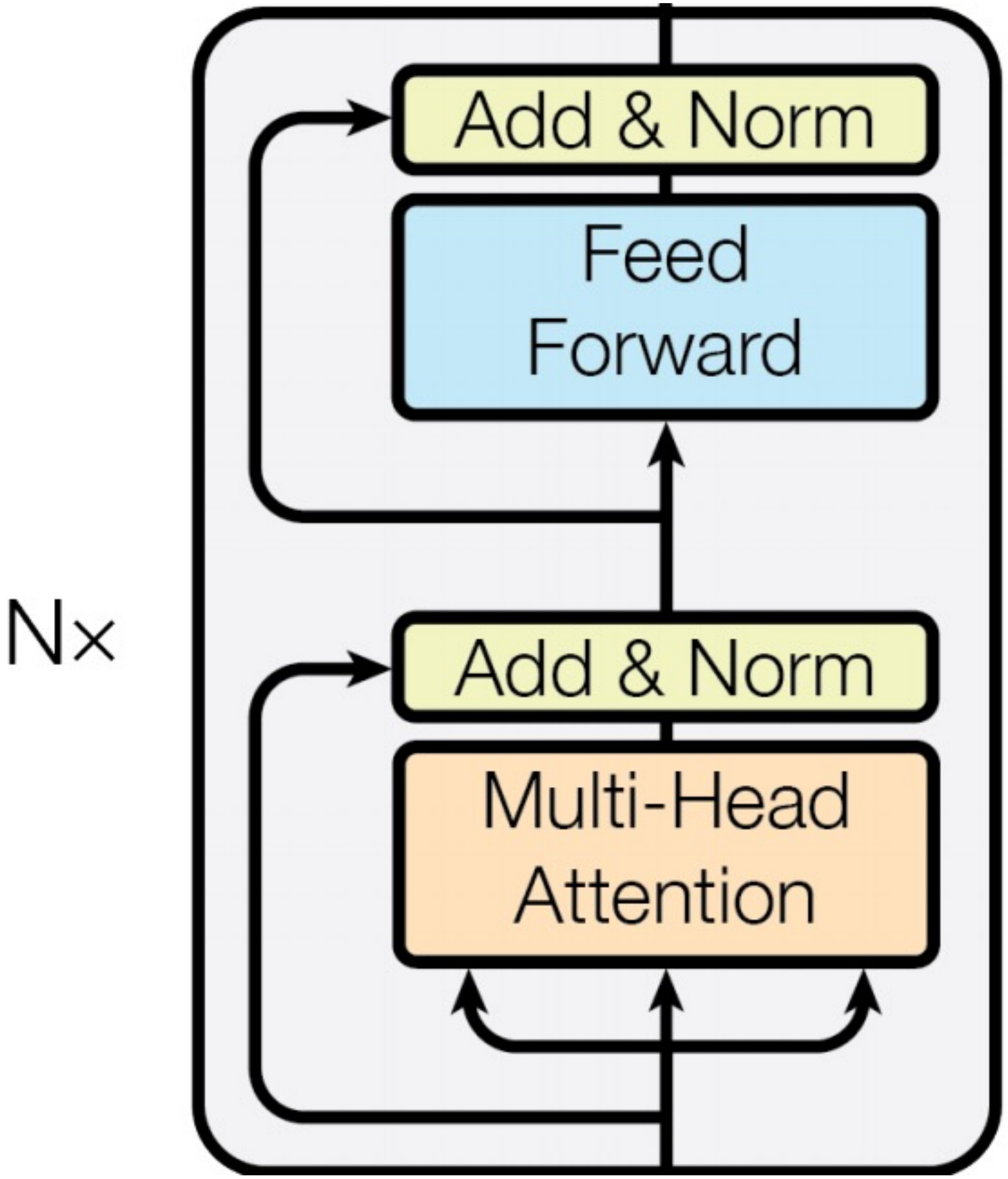
Multi-head attention

- Run h attention models in parallel on top of different linearly projected versions of Q, K, V; concatenate and linearly project the results
- Intuition: enables model to attend to different kinds of information at different positions



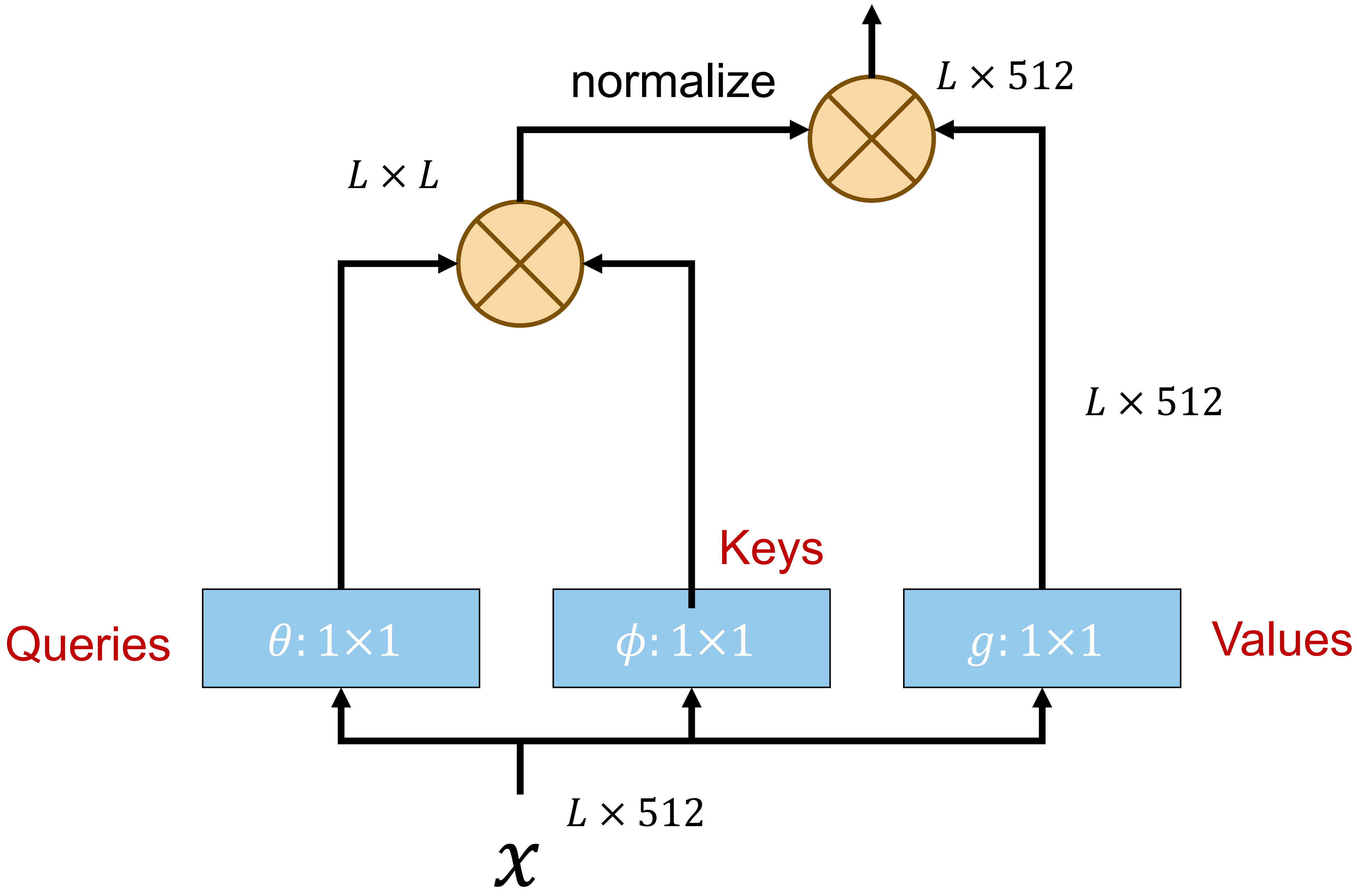
Transformer blocks

- A **Transformer** is a sequence of transformer blocks
  - Vaswani et al.: N=12 blocks, embedding dimension = 512, 6 attention heads
  - **Add & Norm:** residual connection followed by layer normalization
  - **Feedforward:** two linear layers with ReLUs in between, applied independently to each vector
- Attention is the only interaction between inputs!



Positional encoding

Self-attention does not encode the order of the inputs.



$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j)\ g(x_j)$$

Positional encoding

To give transformer information about ordering of tokens, add function of position (based on sines and cosines) to every input

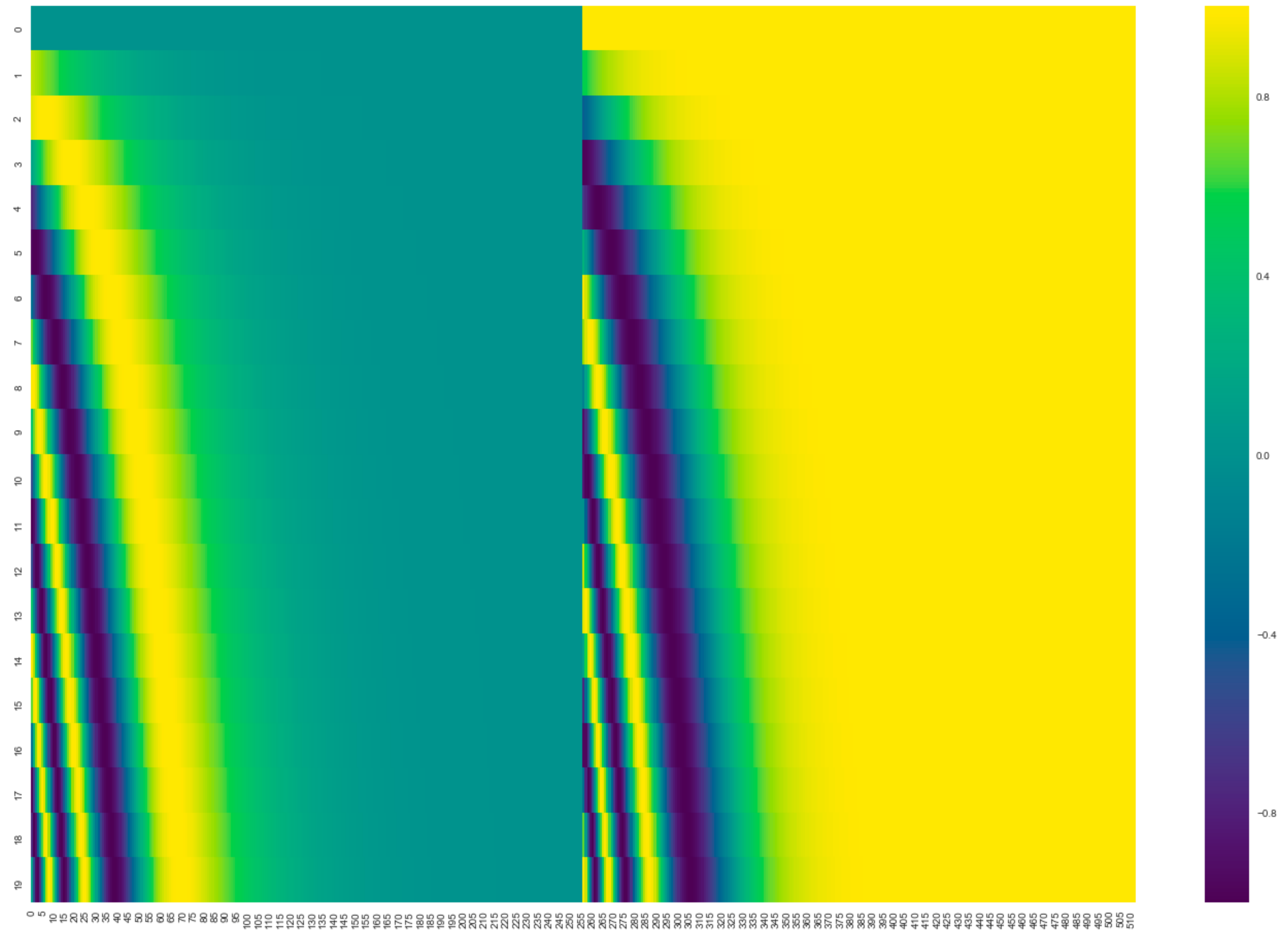
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

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

Positional encoding

To give transformer information about ordering of tokens, add function of position (based on sines and cosines) to every input

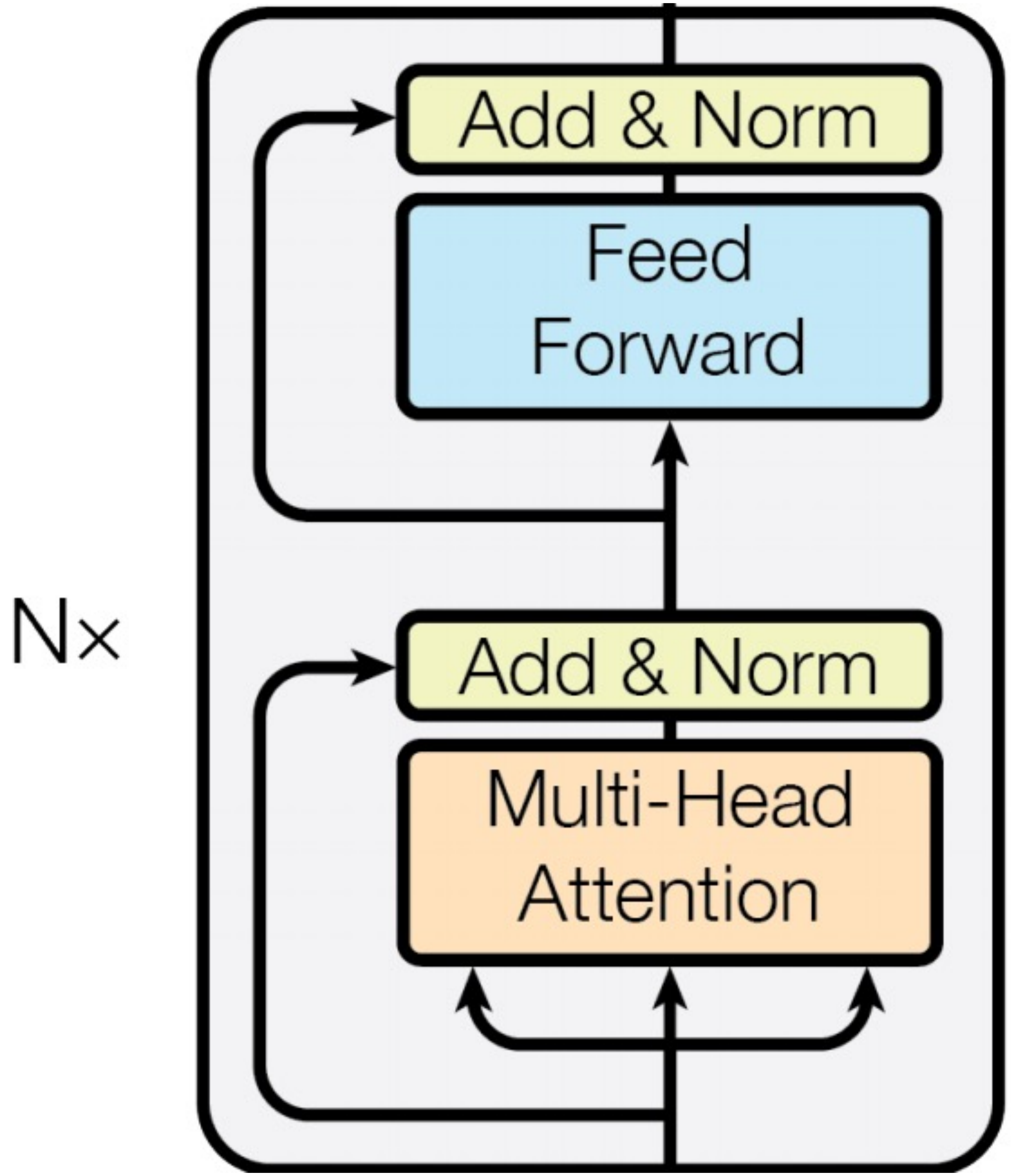
position

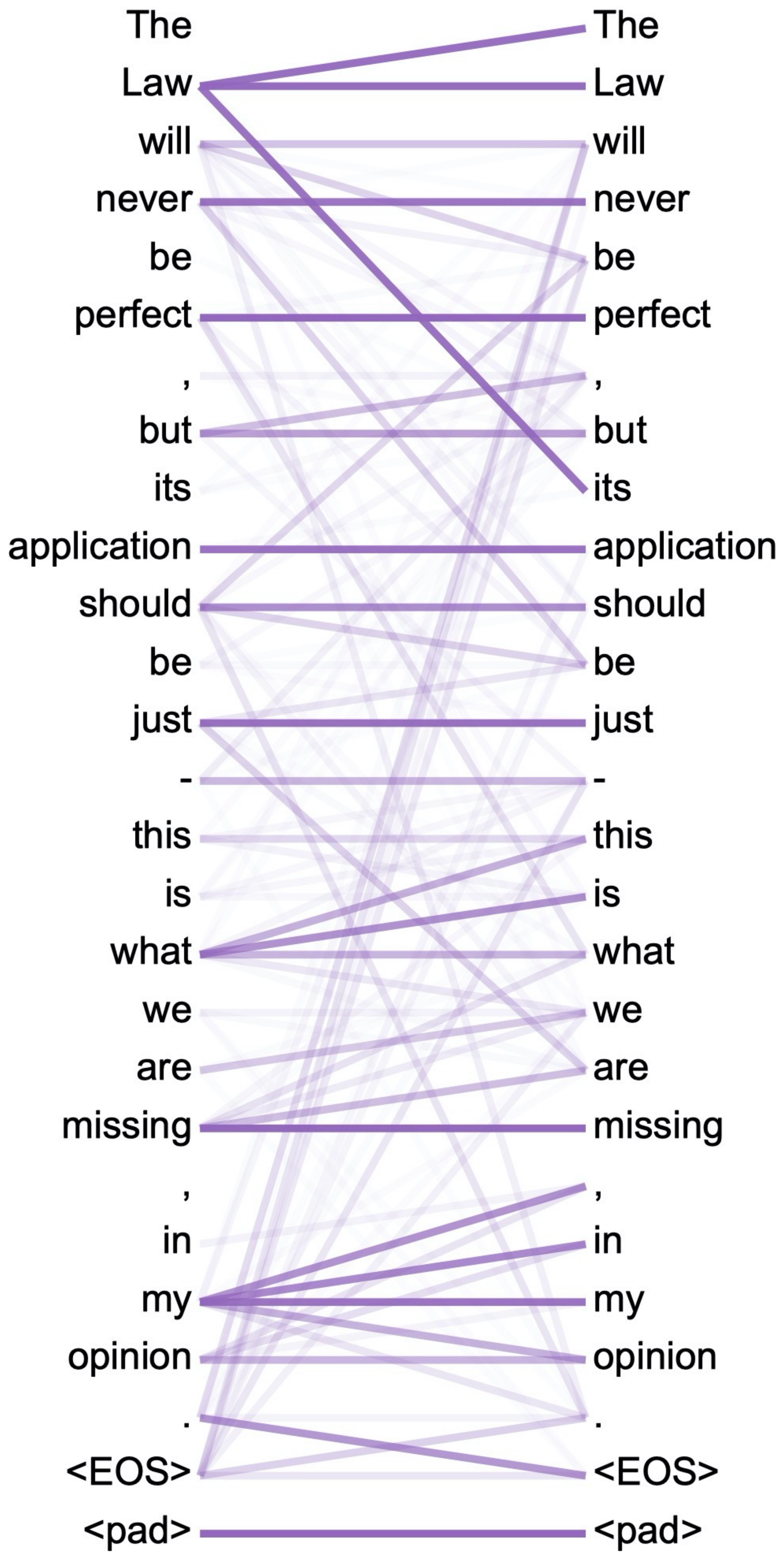


Embedding dimension

Transformer blocks

- A **Transformer** is a sequence of transformer blocks
  - Vaswani et al.: N=12 blocks, embedding dimension = 512, 6 attention heads
  - **Add & Norm:** residual connection followed by [layer normalization]()
  - **Feedforward:** two linear layers with ReLUs in between, applied independently to each vector
- Attention is the only interaction between inputs!

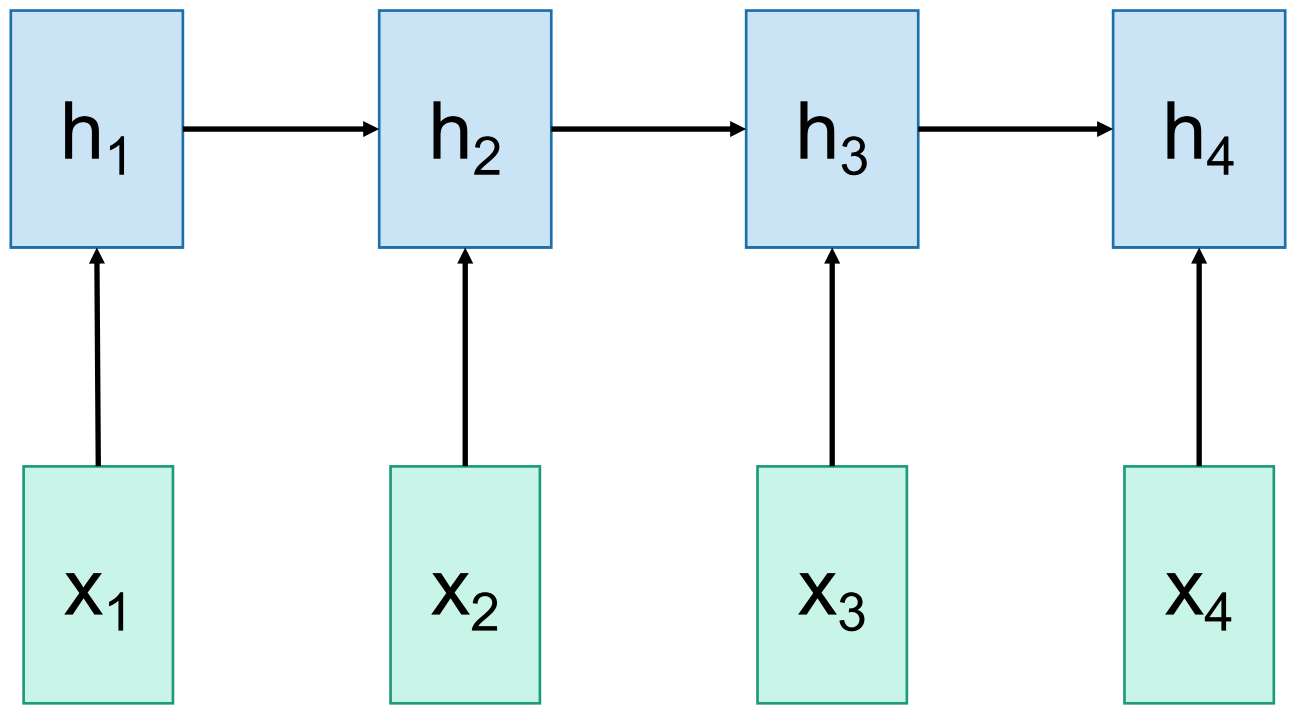



The Law will never be perfect , but its application should be just - this is what we are missing , in my opinion . <EOS> <pad>
The Law will never be perfect , but its application should be just - this is what we are missing , in my opinion . <EOS> <pad>

Different ways of processing sequences

RNN	1D convolutional network	Transformer

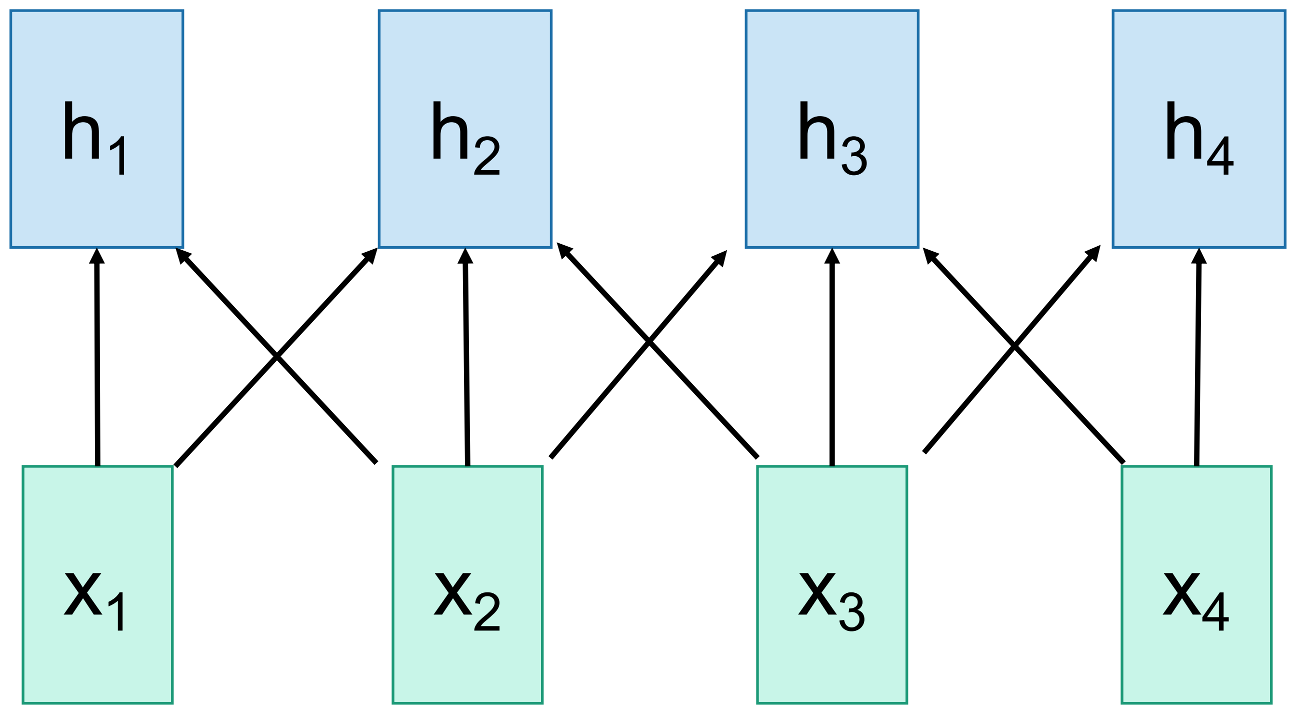
RNN



Works on **ordered sequences**

- Pros: Good at long sequences: the last hidden vector encapsulates the whole sequence
- Cons: Not parallelizable: need to compute hidden states sequentially

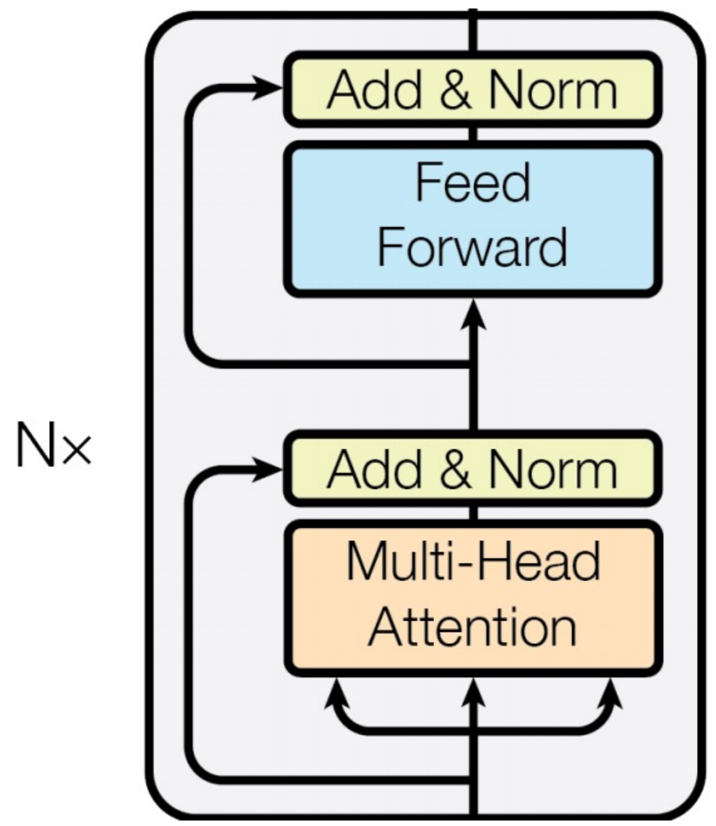
1D convolutional network



Works on **multidimensional grids**

- Con: Bad at long sequences: Need to stack many conv layers for outputs to "see" the whole sequence
- Pro: Highly parallel: Each output can be computed in parallel

Transformer

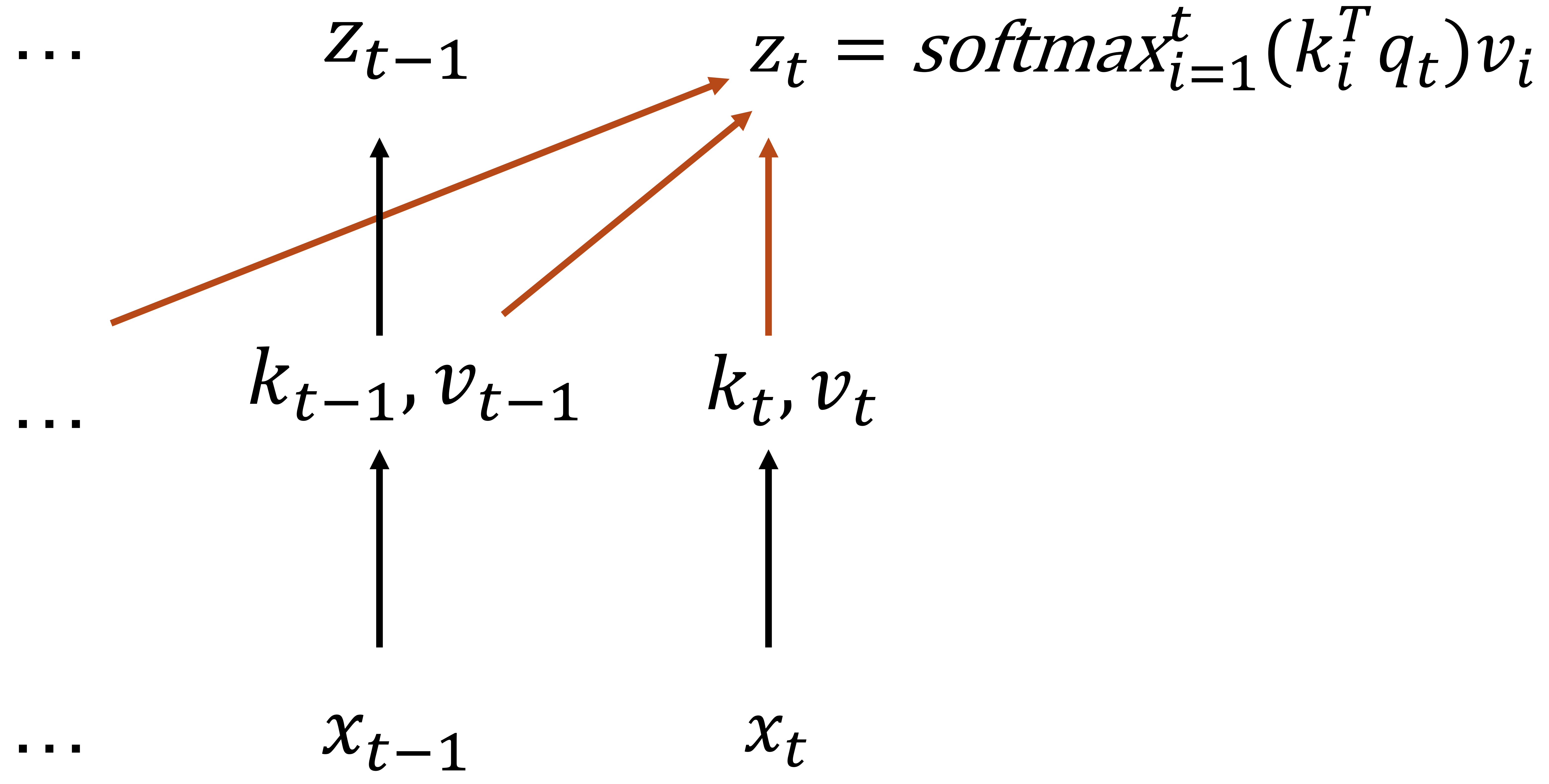


- Works on **sets of vectors**
- Pro: Good at long sequences: after one self-attention layer, each output "sees" all inputs!
- Pro: Highly parallel: Each output can be computed in parallel
- Con: Very memory-intensive

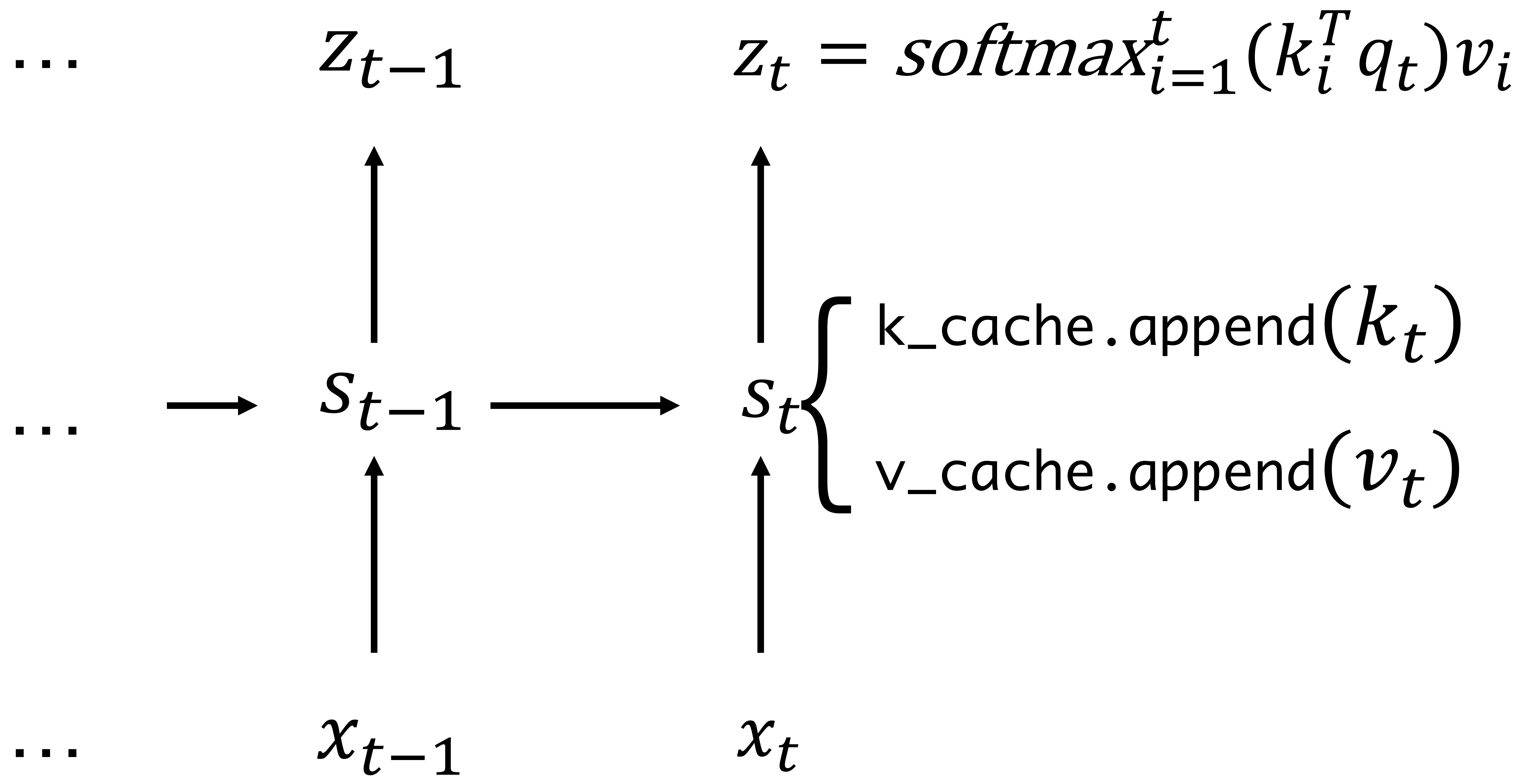
RNN

$$
\begin{tikzcd}
\cdots & z_{t-1} & z_t = \texttt{output}(s_t, x_t) \\
\cdots \arrow[r] & s_{t-1} \arrow[u] \arrow[r] & s_t = \texttt{update}(s_{t-1}, x_t) \arrow[u] \\
\cdots & x_{t-1} \arrow[u] & x_t \arrow[u]
\end{tikzcd}
$$

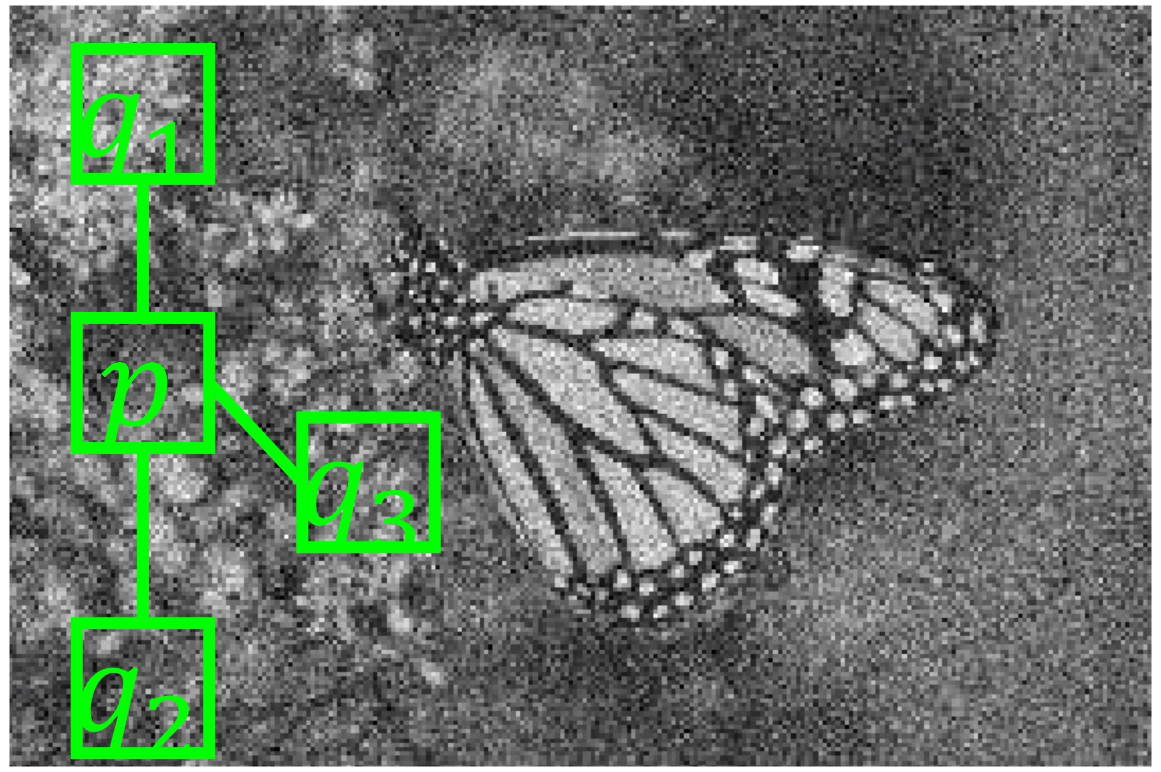
Transformer



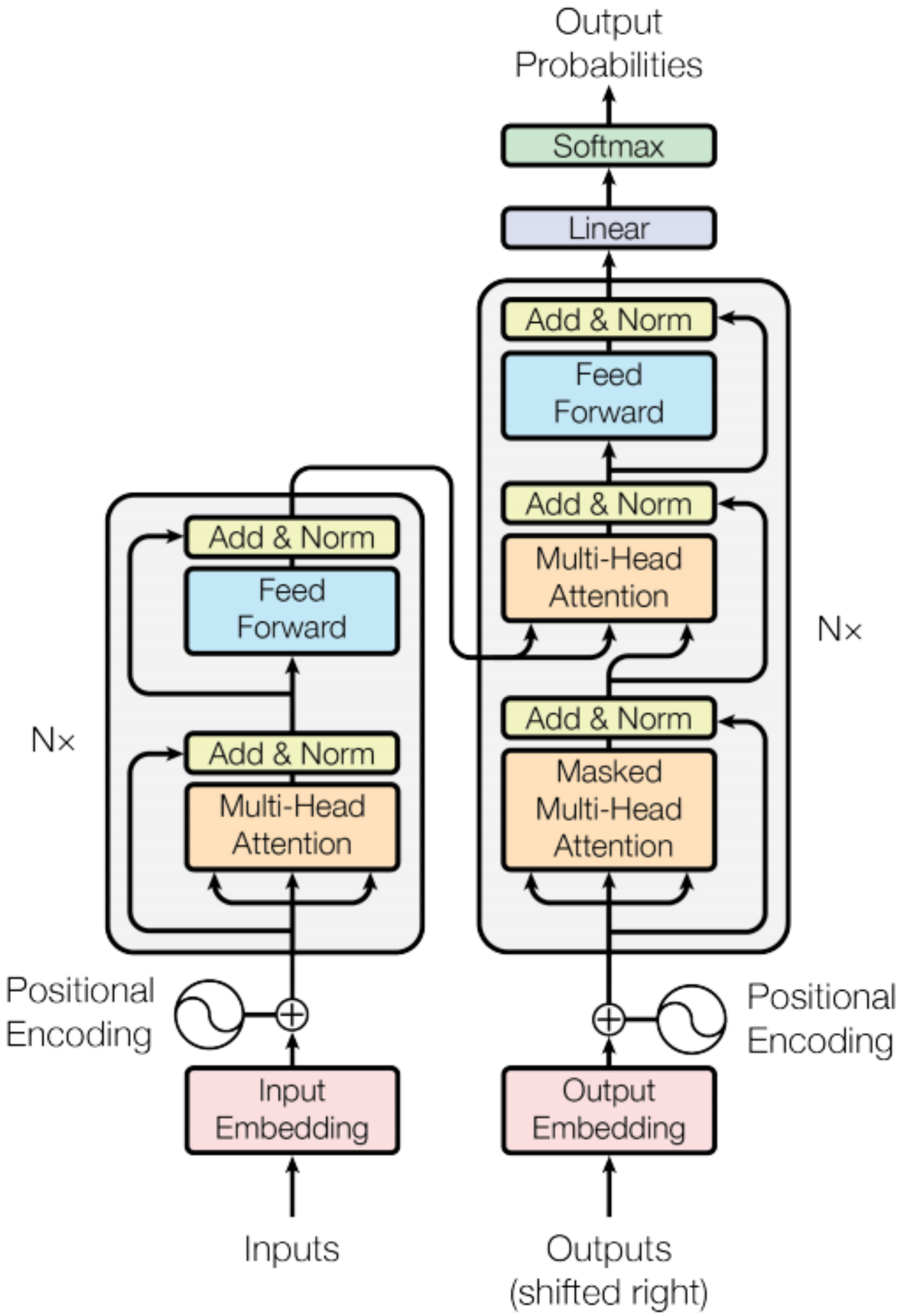
Transformer



It is all Non-local Means



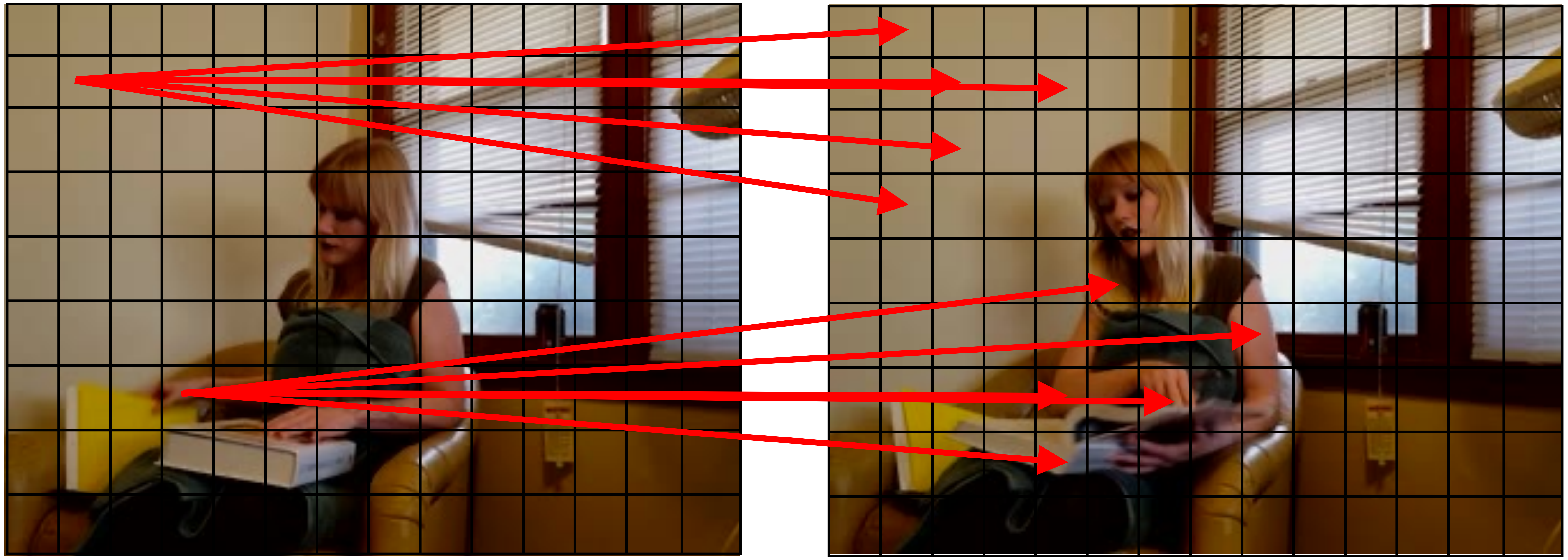
Buades et al., 2005.



Graph Neural Networks and its connection to Self-Attention

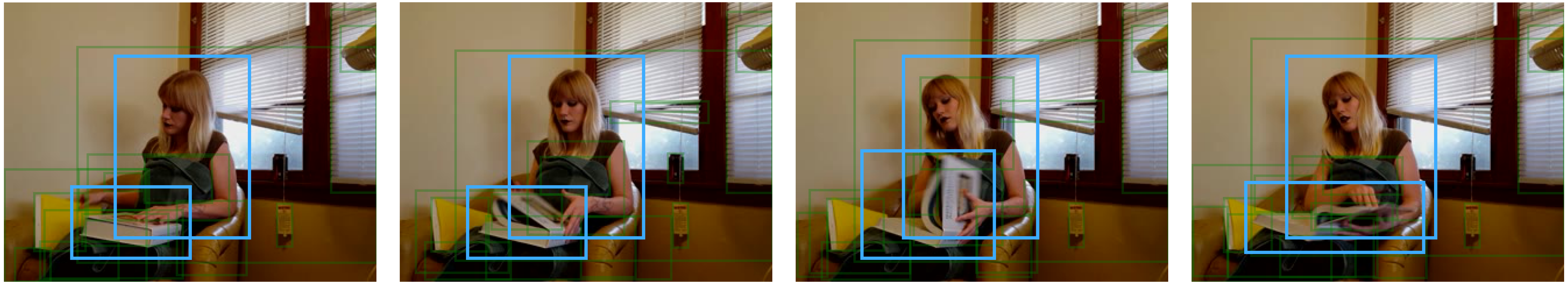
Opening A Book

Opening A Book



The Non-local / Self-Attention Block

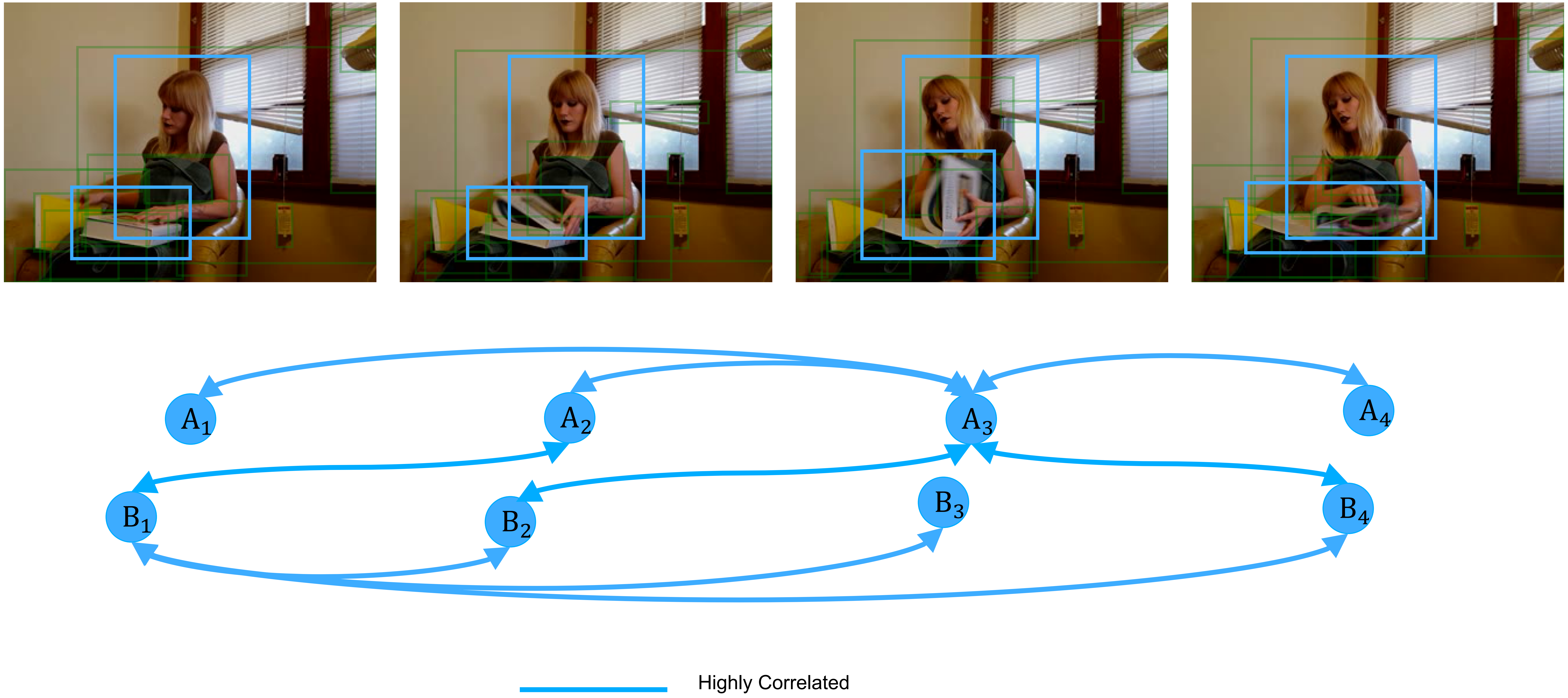
Opening A Book



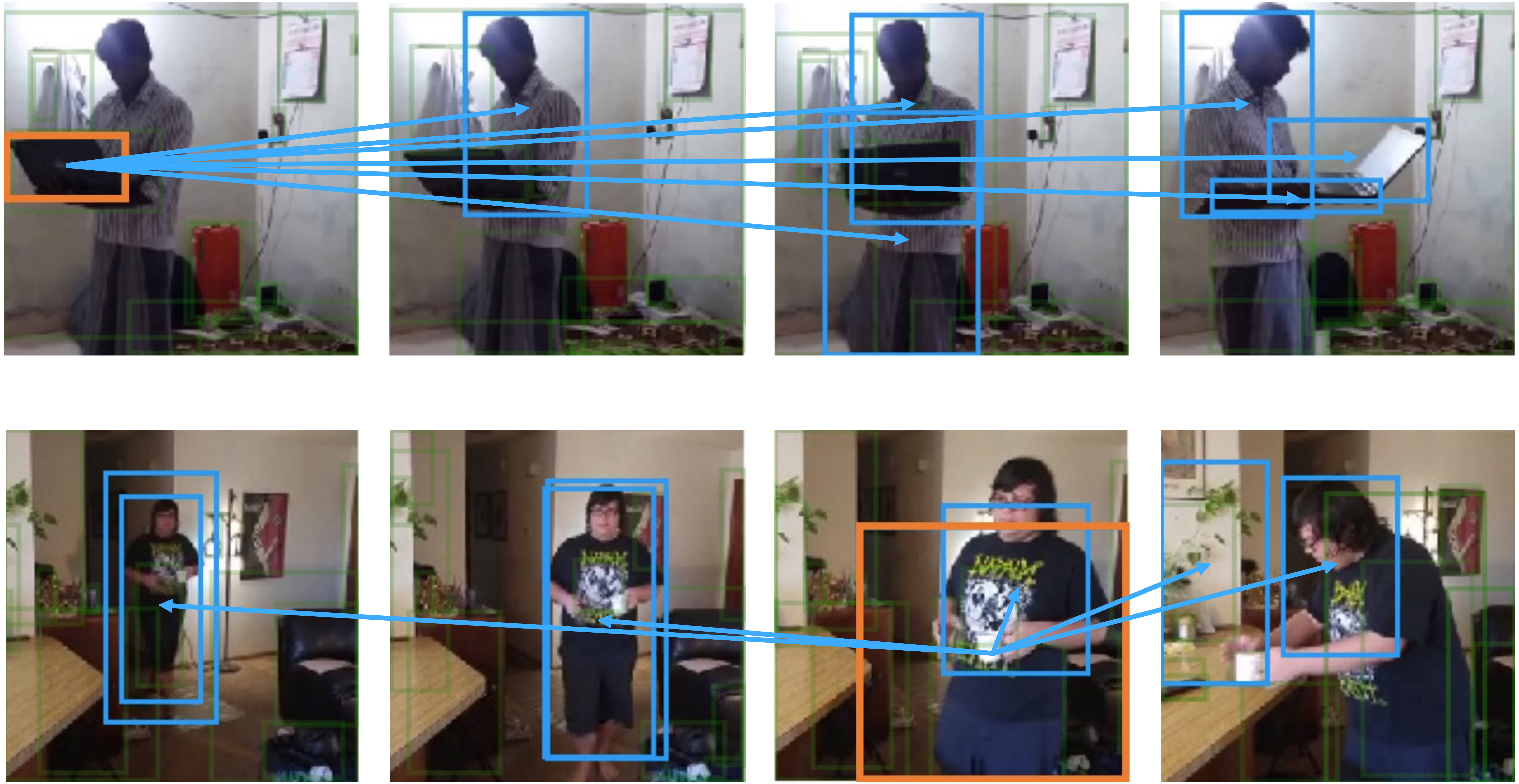
Object states changes over time

Human-object, object-object interactions

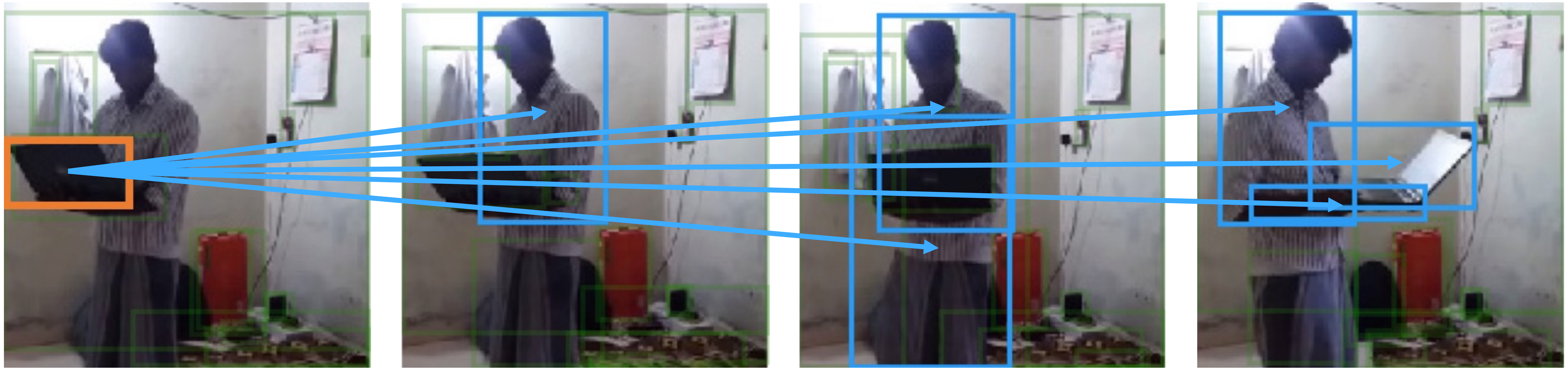
Opening A Book



Relations between Regions



Relations between Regions

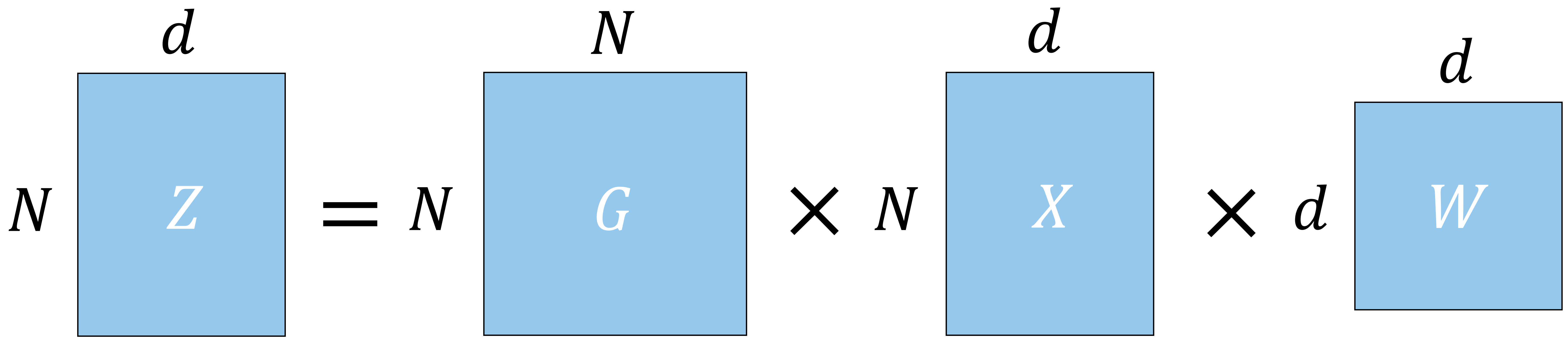


$$f(x_i, x_j) = \phi(x_i)^T \phi'(x_j)$$

$$G_{ij} = \frac{\exp f(x_i, x_j)}{\sum_{\forall j} \exp f(x_i, x_j)}$$

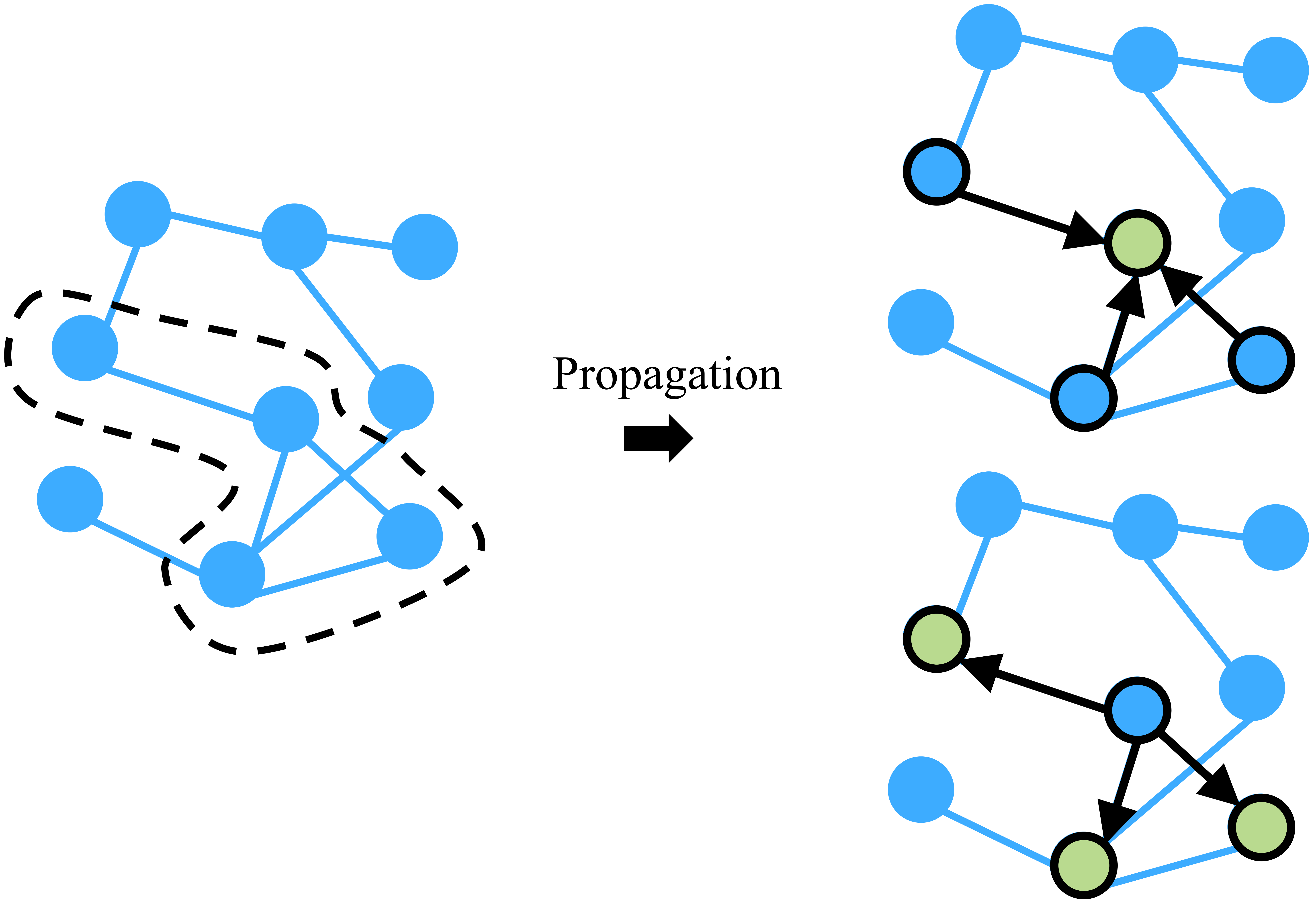
Graph Convolutional Network

$$Z = GXW$$



Kipf et al., 2017

Graph Convolutional Network



Connecting Non-local Means and GCN

The Non-local Operator:

$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j)\ g(x_j)$$

$$= \sum_{\forall j} \frac{f(x_i, x_j)}{\sum_{\forall j} f(x_i, x_j)} g(x_j)$$

$$= \sum_{\forall j} G_{ij}\ g(x_j)$$

$$z_i = y_i W + x_i$$

$$= \sum_{\forall j} G_{ij}\ g(x_j)\, W + x_i$$

$$\boxed{Z = G\ g(X)\ W} + X$$

The Graph Convolution