Inductive biases, graph neural networks, attention and relational inference

Seongok Ryu

ACE-Team, KAIST Chemistry

Abstract of this survey

- Deep neural networks have shown powerful performance on many tasks, such as vision recognition, natural language processing and others.
- The major cornerstone operations of deep neural networks are fully-connected, convolution and recurrence.
- Such operations can be considered as involving different relational inductive biases: weak, locality and sequenciality.
- Graph neural networks, one of the most impactful neural network in 2018, can involve manually defined inductive biases represented by an adjacency matrix.
- Attention mechanisms, which are widely used at NLP and other areas, can be interpreted as procedures to capture relation between elements. In addition, we can more flexibly represent such relations by adopting the attention mechanisms as done in “Attention is all you need”.
- As done in a lot of literatures, the relation between entities can inferred by mimicking attentions and inferring the relation corresponds to the edge state updating in graph neural networks, so-called “relational inference”.
- We present “Inductive biases, graph neural networks, attention mechanism and relational inference” in this survey.

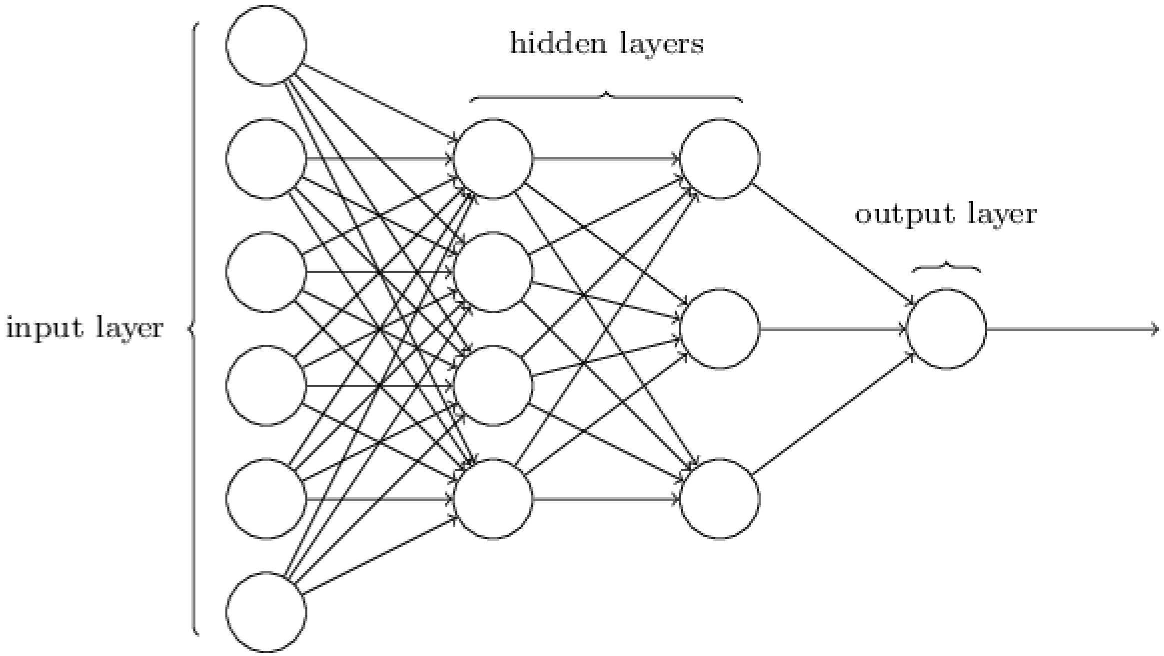
Table of contents

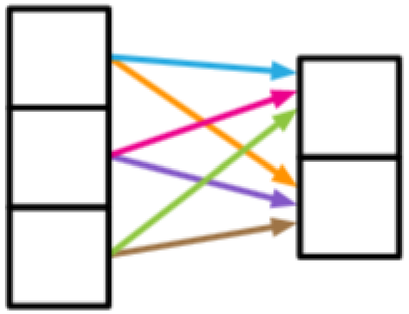
- Inductive biases in neural networks
- Graph neural networks
- Attention mechanism
- Relational inference

Inductive biases in neural network

Weight sharing in neural network

Fully-connected neural network (sometimes referred as multi-layer perceptron)



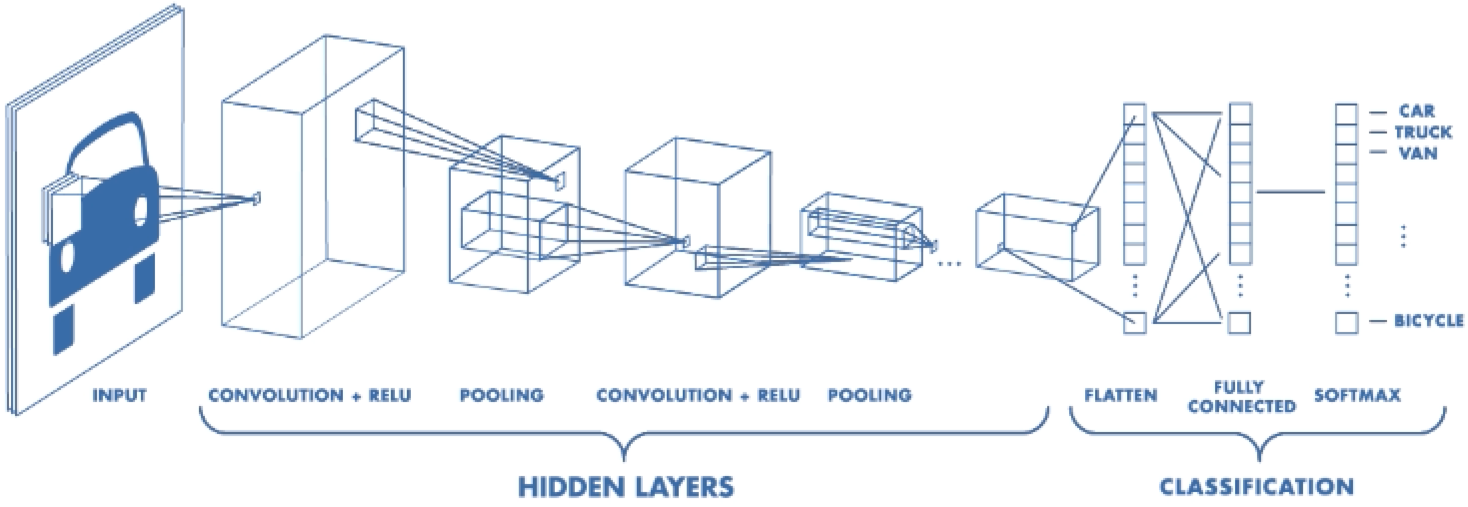


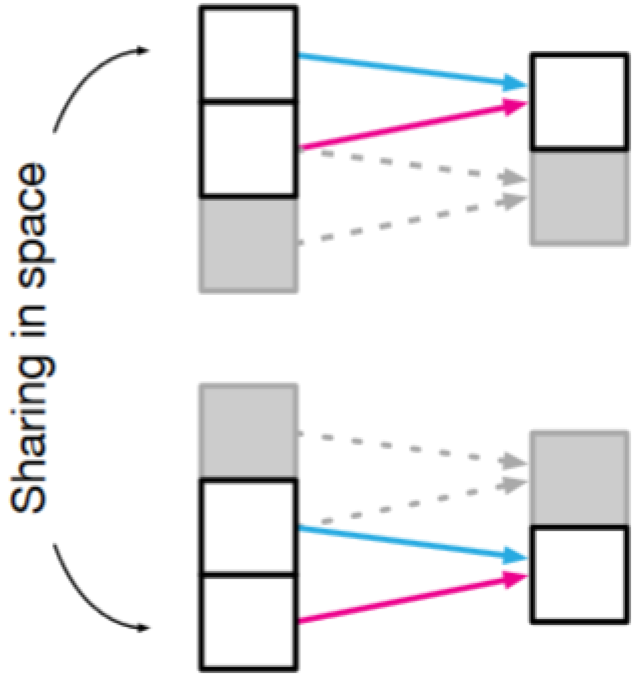
No weight sharing

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Weight sharing in neural network

Convolutional neural network

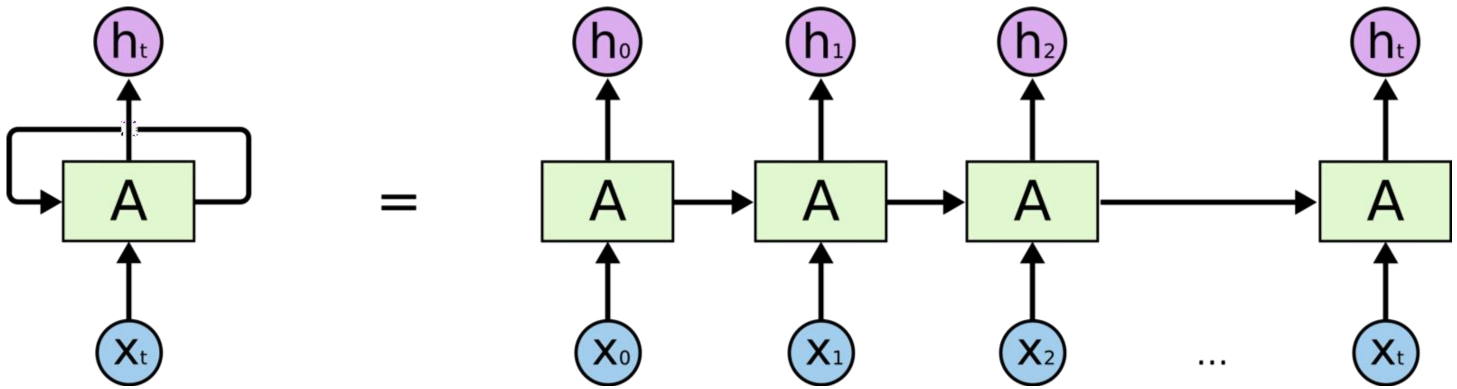


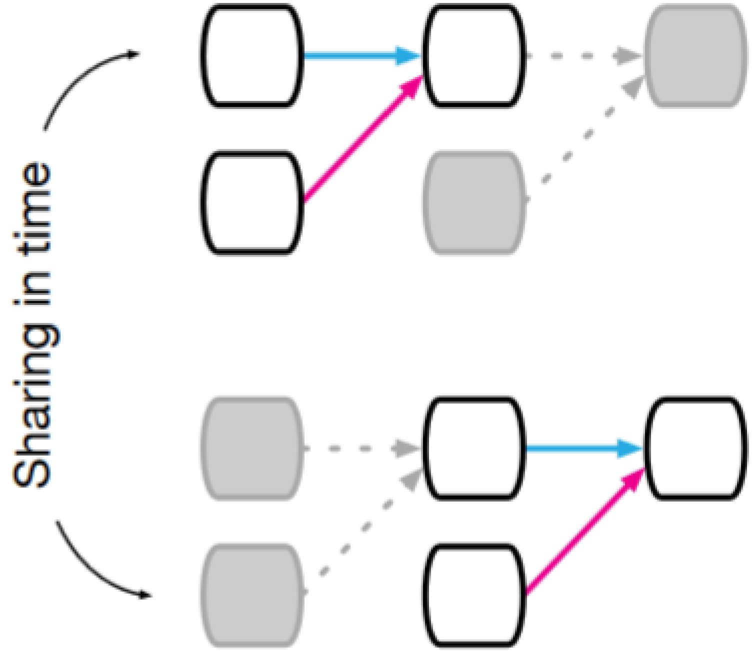


Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Weight sharing in neural network

Recurrent neural network

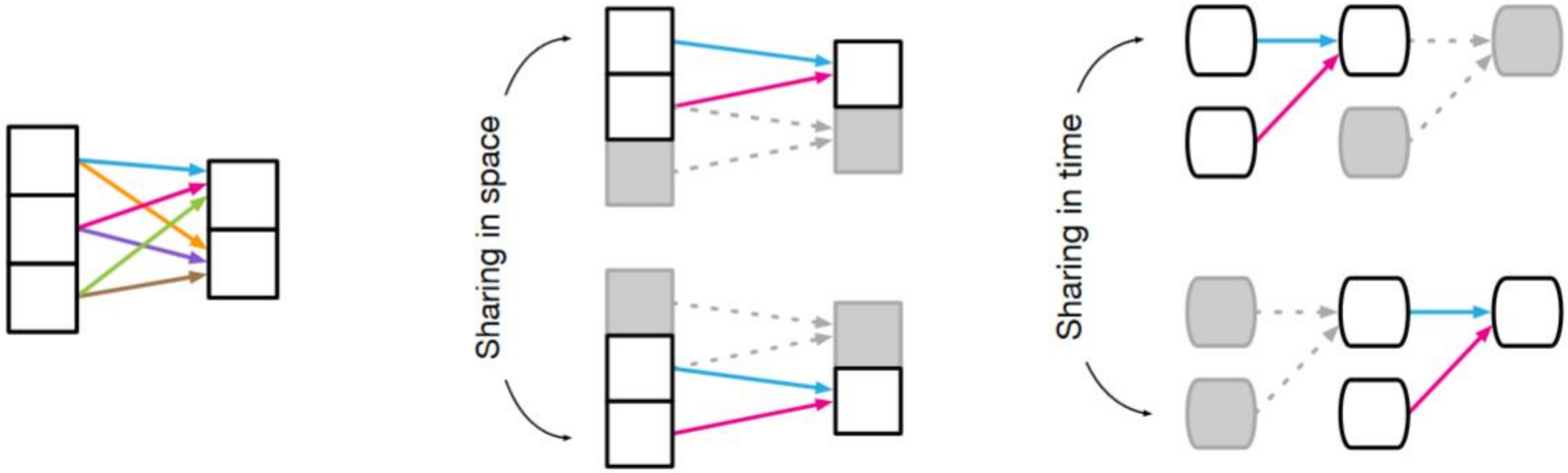




Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Inductive biases

Inductive bias is another name of weight sharing

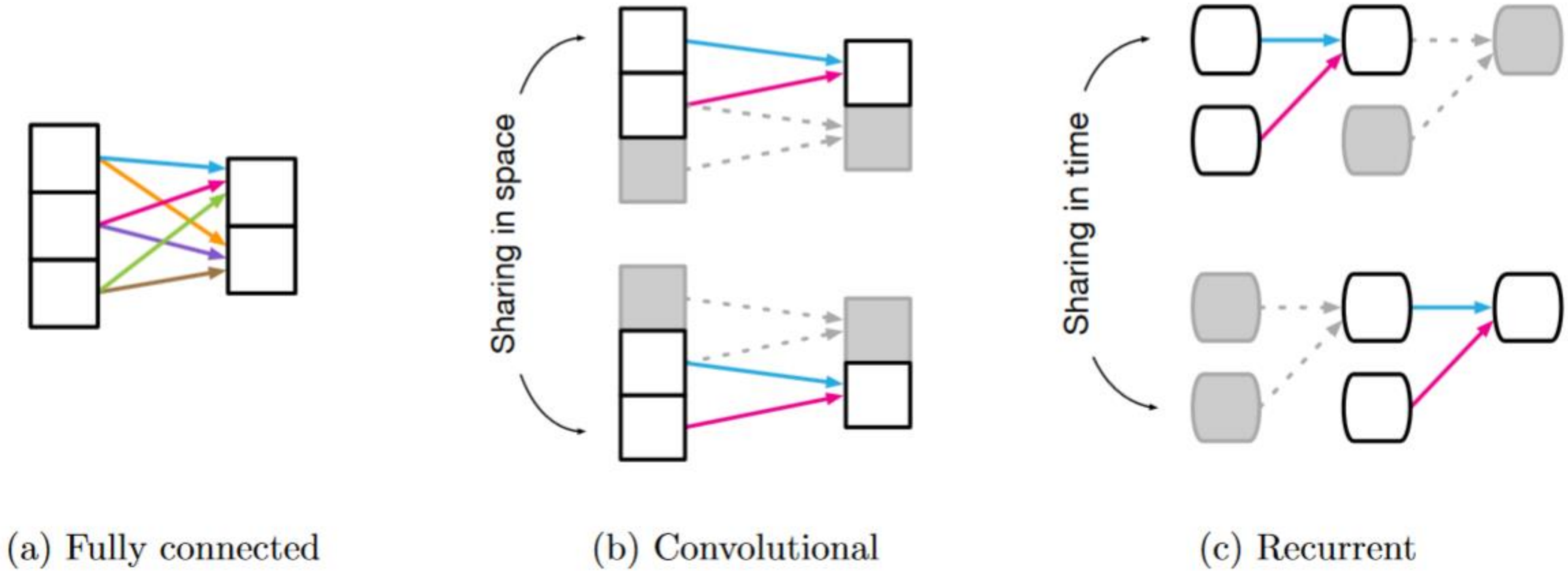


(a) Fully connected (b) Convolutional (c) Recurrent

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Inductive biases

Inductive bias is another name of weight sharing

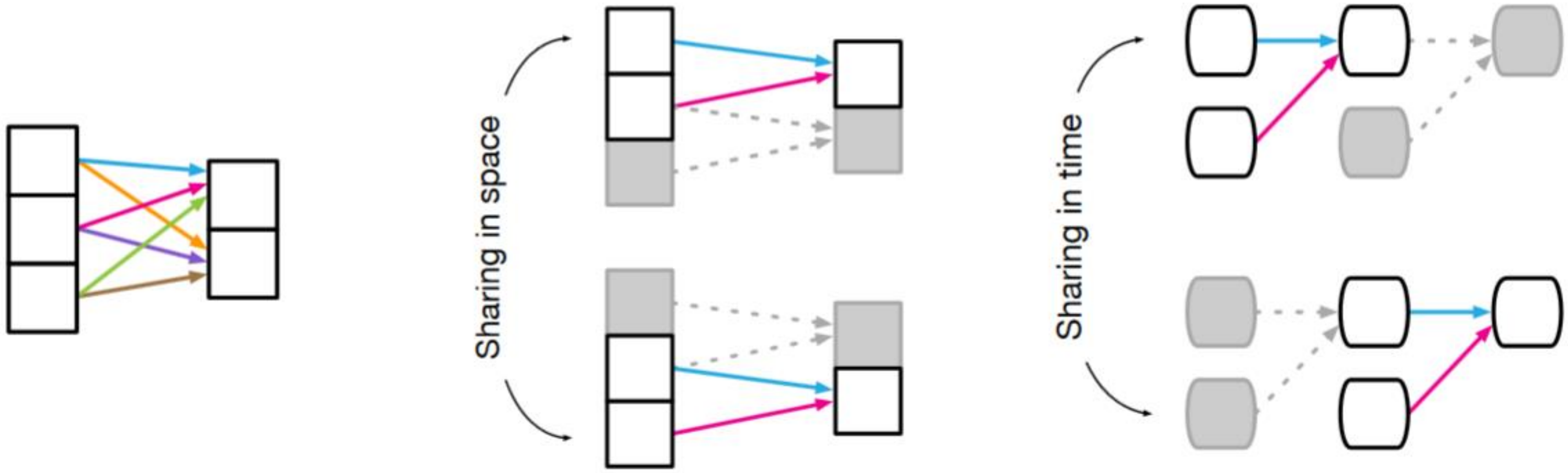


Q) What is the inductive bias for each operation?

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Inductive biases

Inductive bias is another name of weight sharing



(a) Fully connected (b) Convolutional (c) Recurrent

Q) What is the inductive bias for each operation?
Q) Before the question, what is the meaning of inductive bias?

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Inductive biases

Interpretations of the inductive bias

- ✓ An inductive bias allows a learning algorithm to prioritize one solution (or interpretation) over another, independent of the observed data.

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Inductive biases

Interpretations of the inductive bias

- ✓ An inductive bias allows a learning algorithm to prioritize one solution (or interpretation) over another, independent of the observed data.
- ✓ In a Bayesian model, inductive biases are typically expressed through the choice and parameterization of the prior distribution.

$$p(\omega|X,Y) = \frac{p(Y|X,\omega)\cdot \boldsymbol{p(\omega)}}{p(Y|X)}$$

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Inductive biases

Interpretations of the inductive bias

- ✓ An inductive bias allows a learning algorithm to prioritize one solution (or interpretation) over another, independent of the observed data.
- ✓ In a Bayesian model, inductive biases are typically expressed through the choice and parameterization of the prior distribution.

$$p(\omega|X,Y) = \frac{p(Y|X,\omega) \cdot \boldsymbol{p(\omega)}}{p(Y|X)}$$

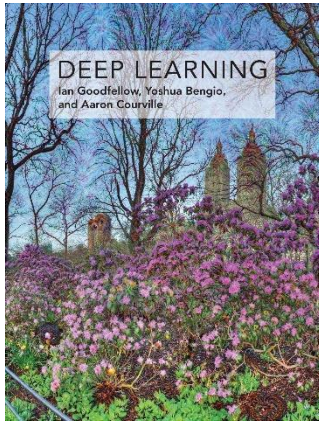
- ✓ In other contexts, an inductive bias might be a regularization term added to avoid overfitting, or it might be encoded in the architecture of the algorithm itself.

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Inductive biases

Interpretations of the inductive bias

“**Priors can be considered weak or strong depending on how concentrated the probability density in the prior is**. A weak prior is a prior distribution with high entropy, such as a Gaussian distribution with high variance. Such a prior allows the data to move the parameters more or less freely. A strong prior has very low entropy, such as a Guassian distribution with low variance. Such a prior plays a more active role in determining where the parameters end up.
An infinitely strong prior places zero probability on some parameters and says that these parameter values are completely forbidden, regardless of how much support the data gives to those values.”

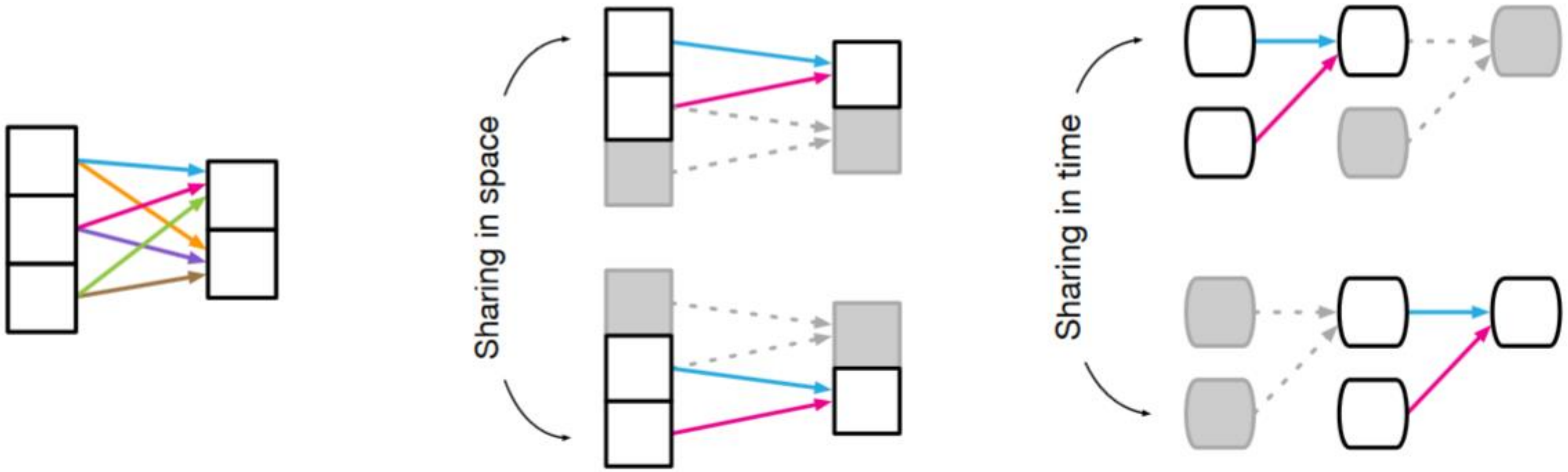


- In 9.4 Convolution and Pooling as an Infinitely Strong Prior

Goodfellow, Ian, et al. *Deep learning*. Vol. 1. Cambridge: MIT press, 2016.

Inductive biases

Inductive biases in neural networks

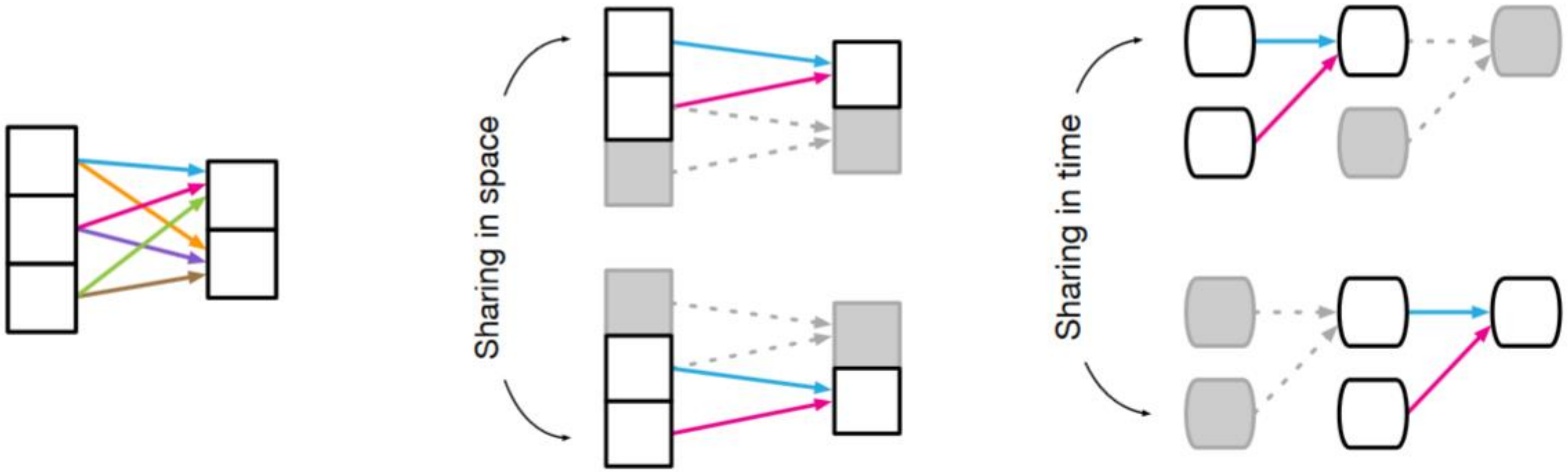


Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Inductive biases

Inductive biases in neural networks



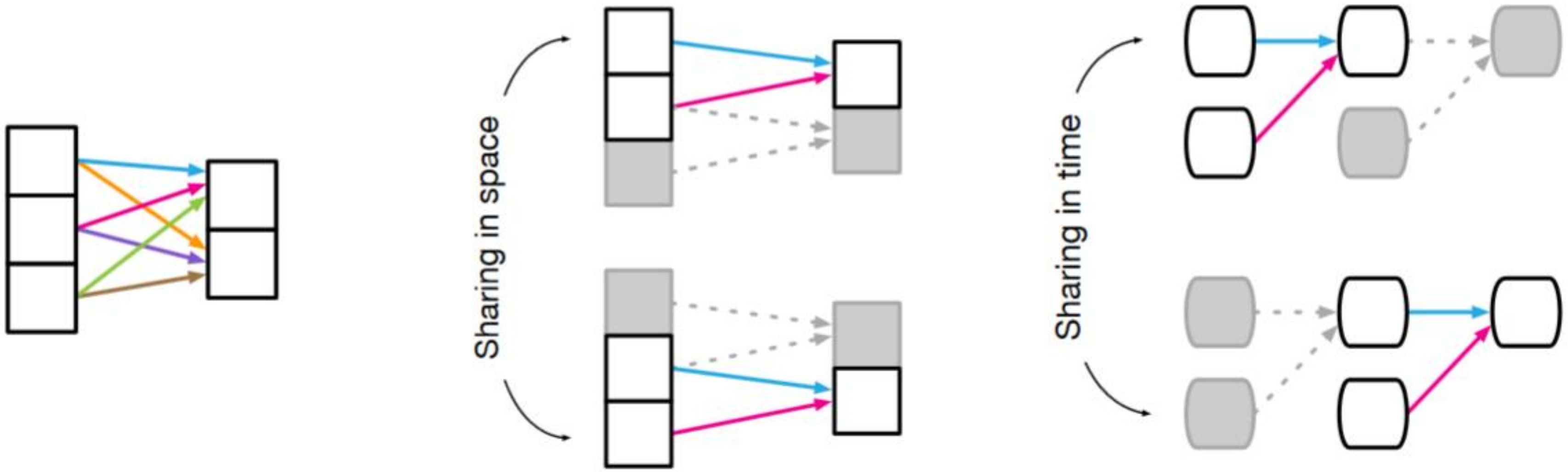
Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation

Q) What are the entities, relations, relational inductive bias and invariance for GNN?

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Inductive biases

Inductive biases in neural networks



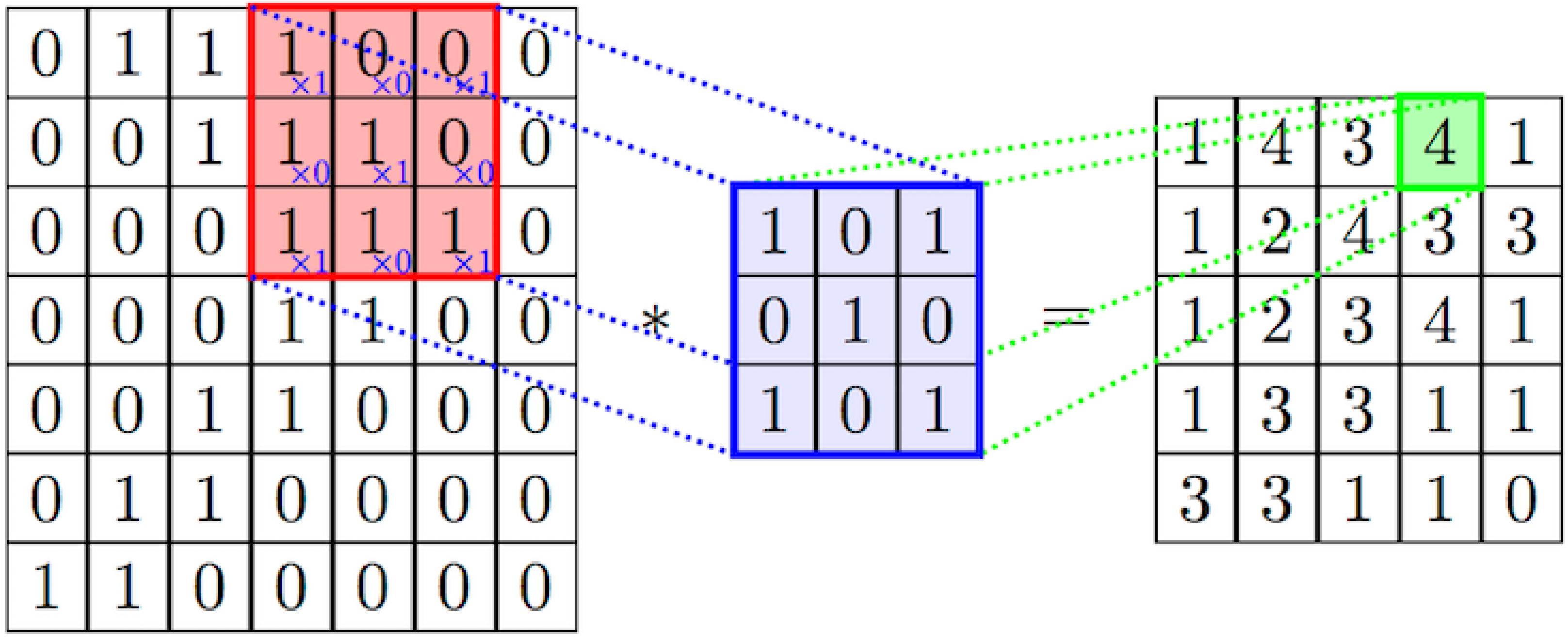
Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation
Graph network	Nodes	Edges	Arbitrary	Node, edge permutations

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Graph neural networks

Graph neural network

Convolutional neural network

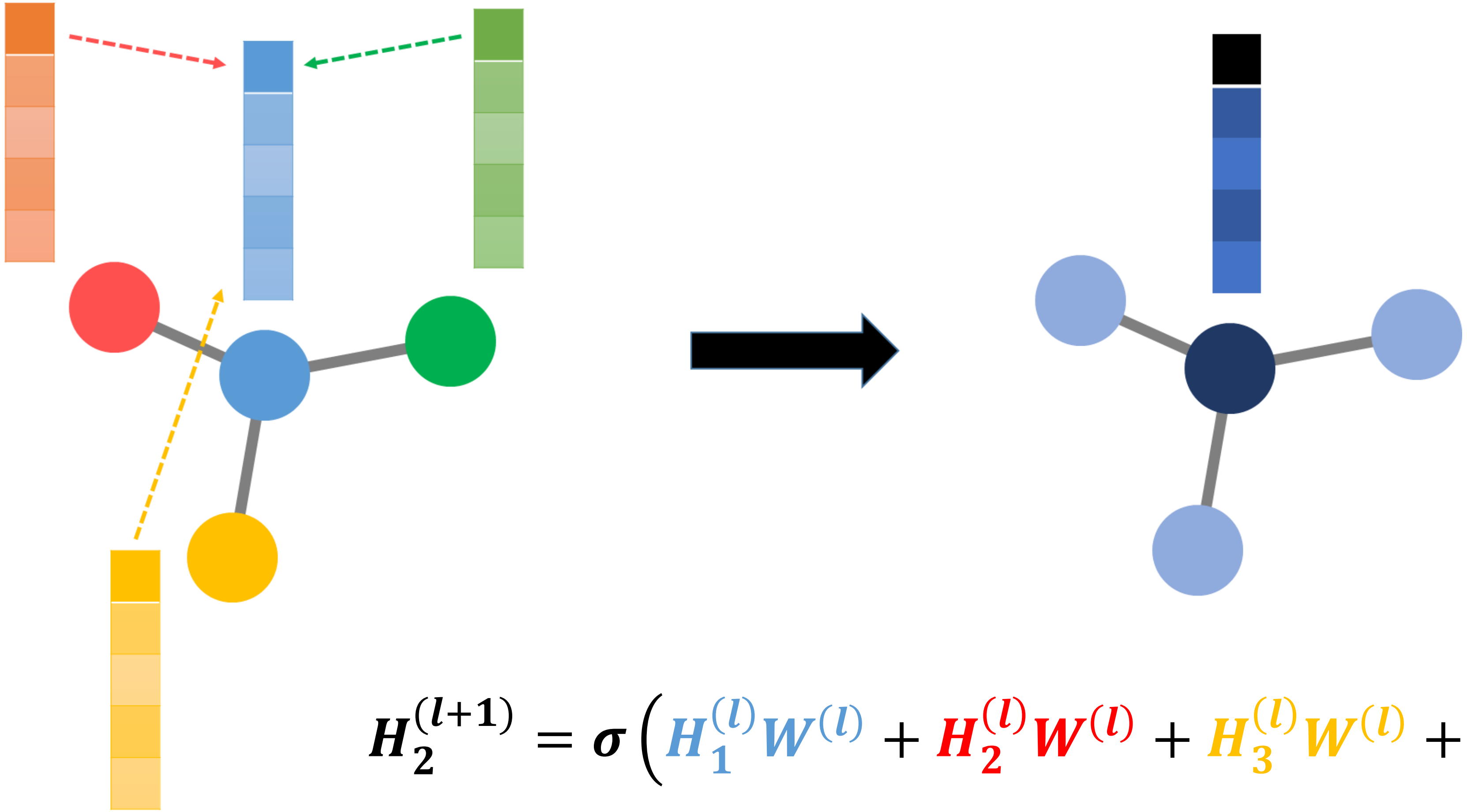


$$\boldsymbol{X}_{\boldsymbol{i}}^{(\boldsymbol{l}+\mathbf{1})} = \boldsymbol{\sigma}\left(\sum_{\boldsymbol{j}\in[\boldsymbol{i}-\boldsymbol{k},\boldsymbol{i}+\boldsymbol{k}]} \boldsymbol{W}_{\boldsymbol{j}}^{(\boldsymbol{l})} \boldsymbol{X}_{\boldsymbol{j}}^{(\boldsymbol{l})} + \boldsymbol{b}^{(\boldsymbol{l})}\right)$$

Learnable parameters are shared

Graph neural network

Graph convolutional network

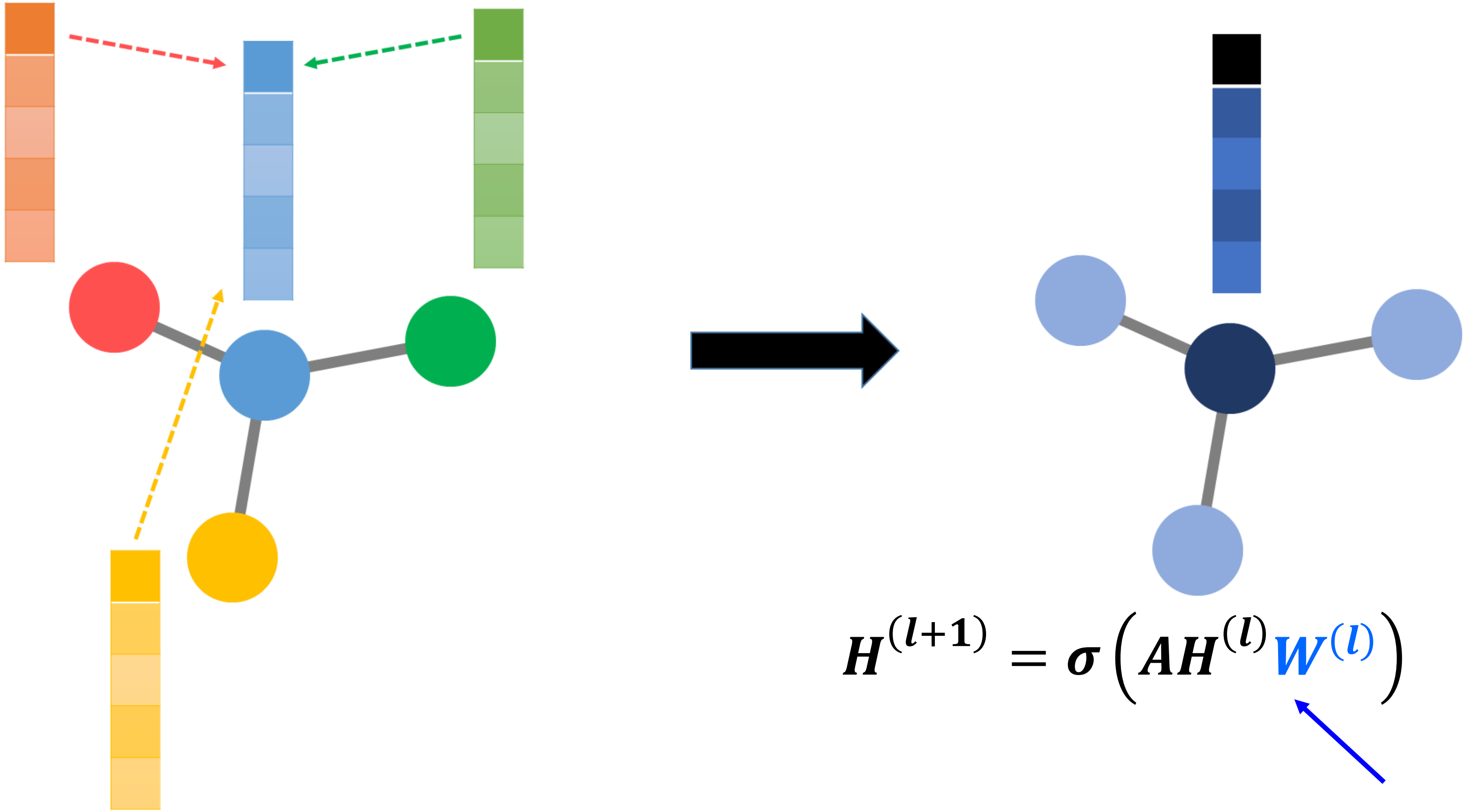


$$H_2^{(l+1)} = \sigma\left(H_1^{(l)}W^{(l)} + H_2^{(l)}W^{(l)} + H_3^{(l)}W^{(l)} + H_4^{(l)}W^{(l)}\right)$$

$$\Rightarrow \quad H_i^{(l+1)} = \sigma\left(\sum_{j\in N(i)} H_j^{(l)}W^{(l)}\right)$$

Graph neural network

Graph convolutional network



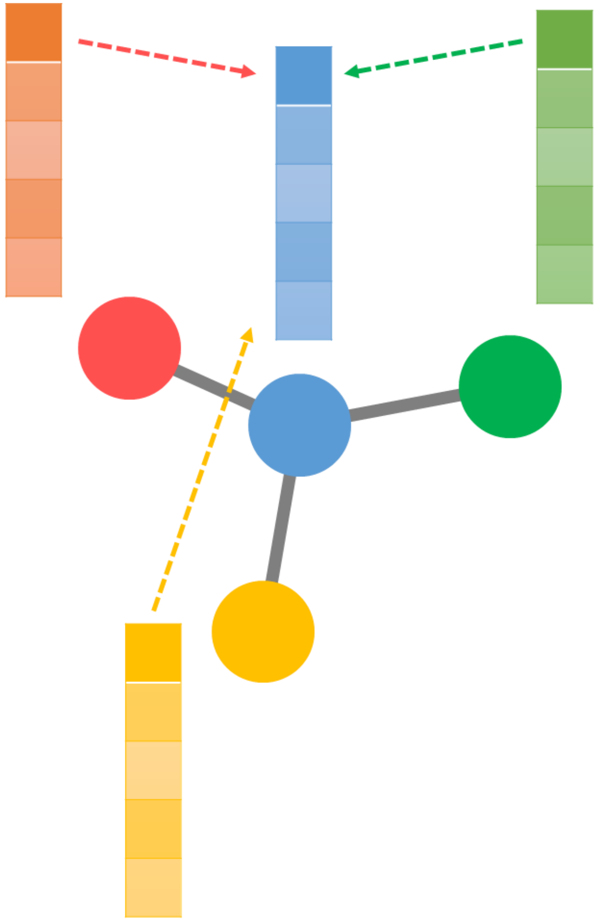
$$H^{(l+1)} = \sigma\left(AH^{(l)}W^{(l)}\right)$$

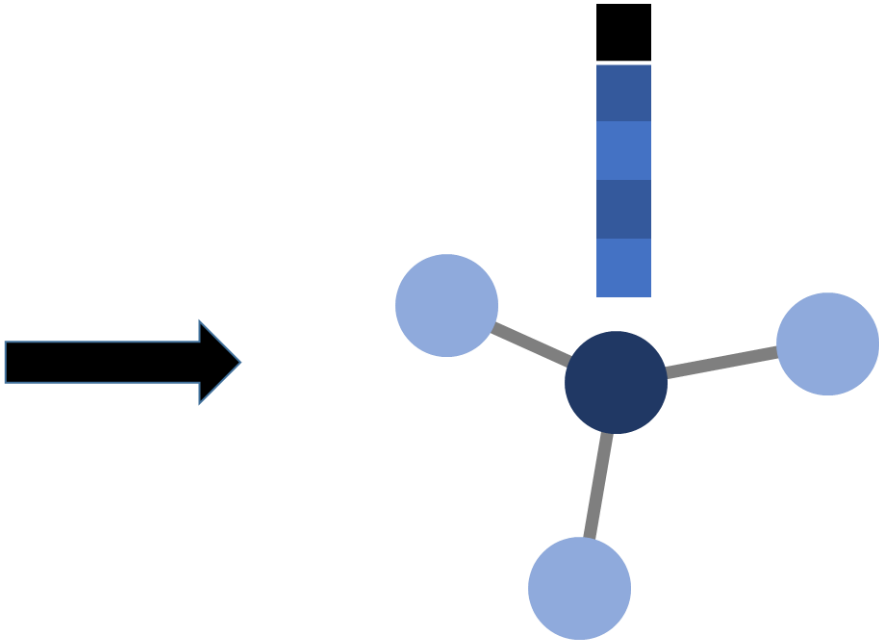
Learnable parameter is shared

Question) What is the inductive bias for GCN?

Graph neural network

Graph convolutional network





$$H^{(l+1)} = \sigma\left(AH^{(l)}W^{(l)}\right)$$

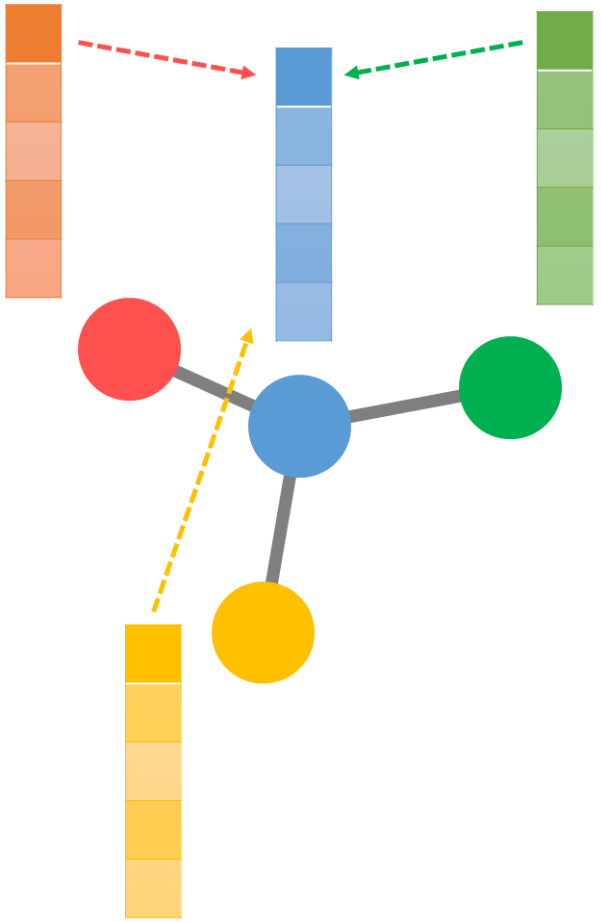
Learnable parameter is shared

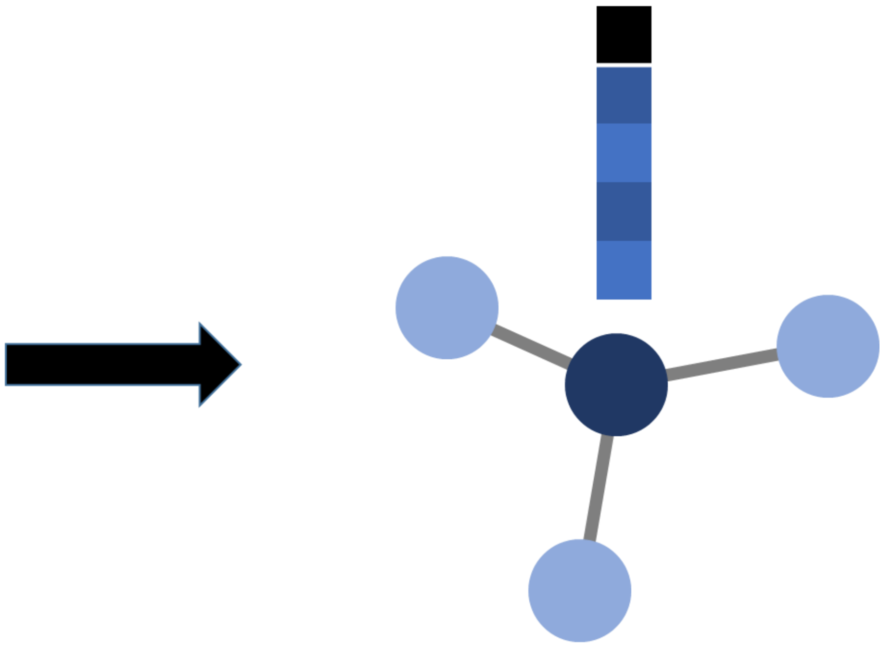
Question) What is the inductive bias for GCN?

Answer) Connectivity between nodes – the adjacency matrix

Graph neural network

Graph convolutional network





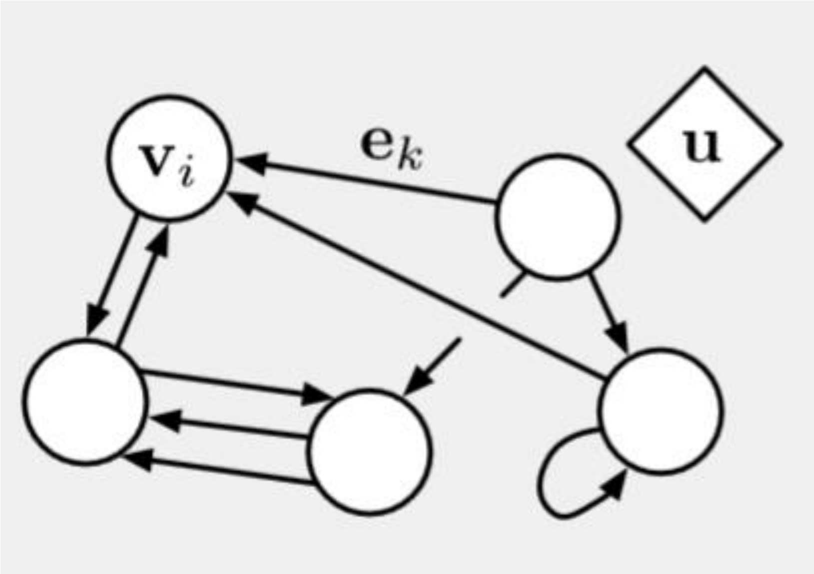
$$H^{(l+1)} = \sigma\left(AH^{(l)}W^{(l)}\right)$$

Learnable parameter is shared

Sharing weights for all nodes in graph,
but nodes are differently updated by reflecting individual node features, $H_j^{(l)}$

Graph neural network

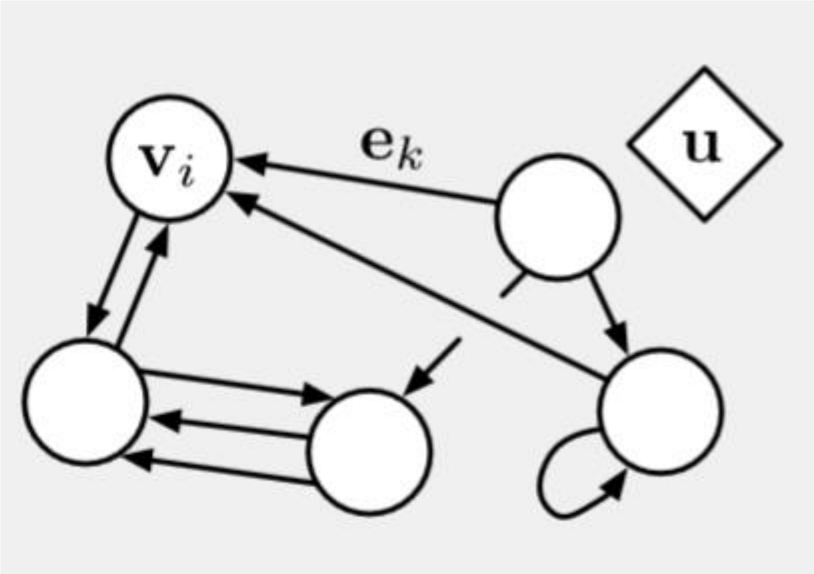
Graph neural networks



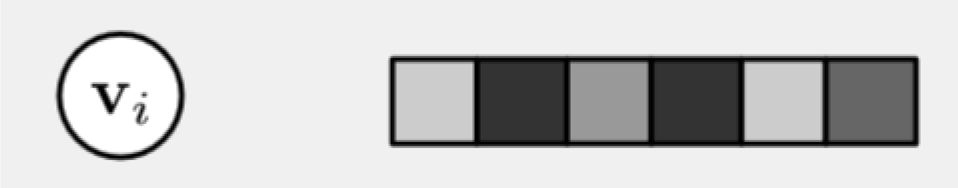
Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Graph neural network

Graph neural networks



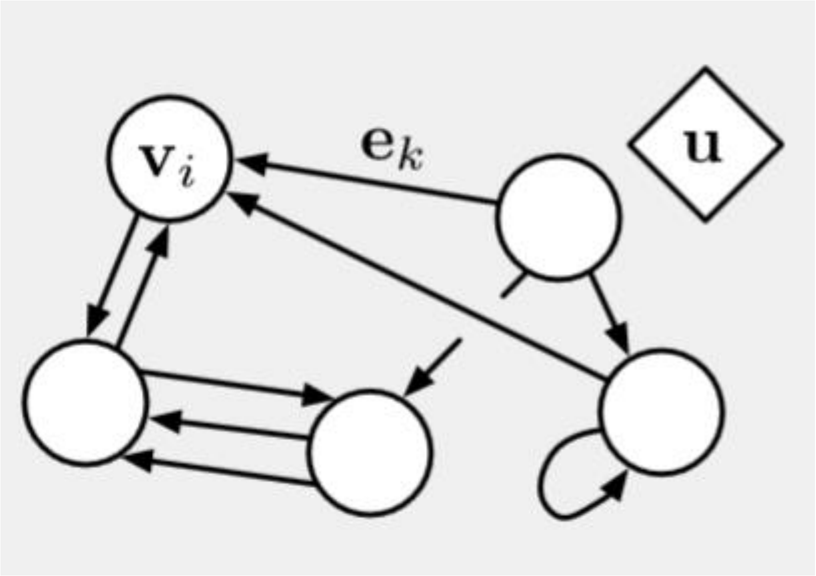
Node's attribute



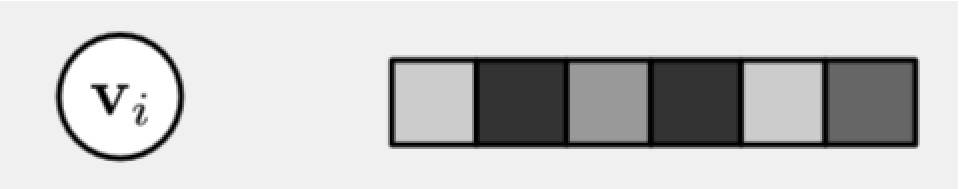
Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Graph neural network

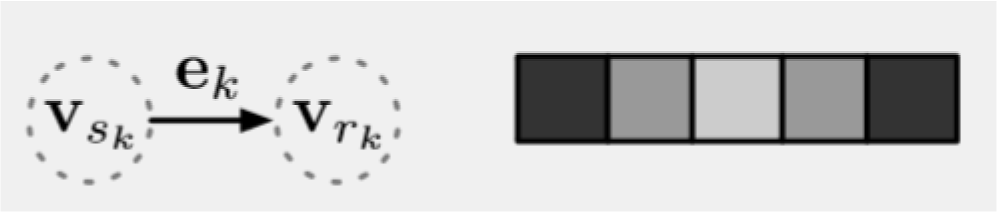
Graph neural networks



Node's attribute



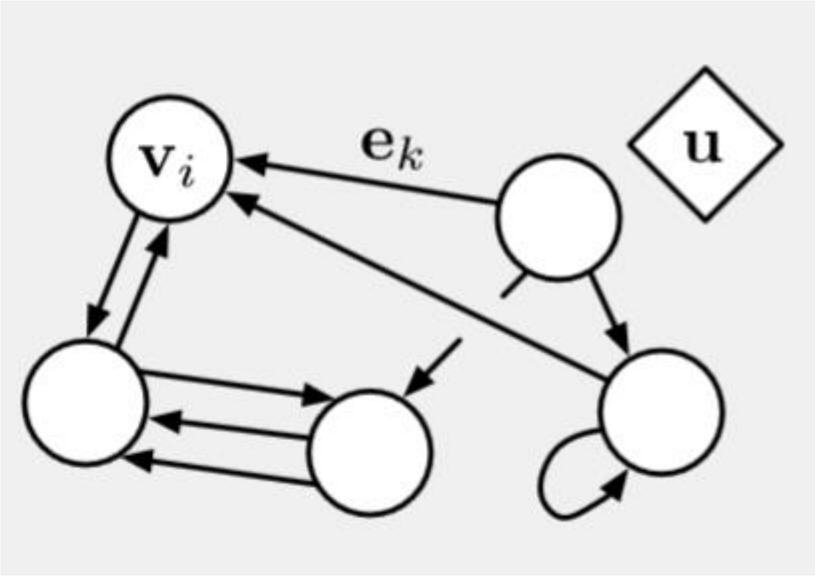
Edge's attribute



Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

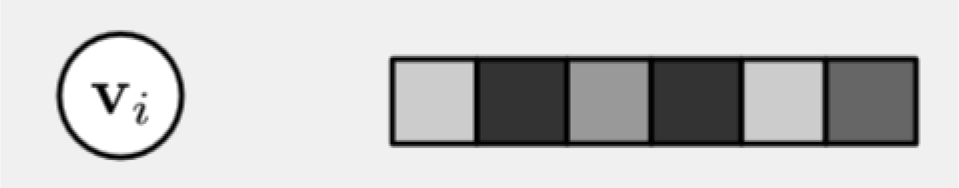
Graph neural network

Graph neural networks

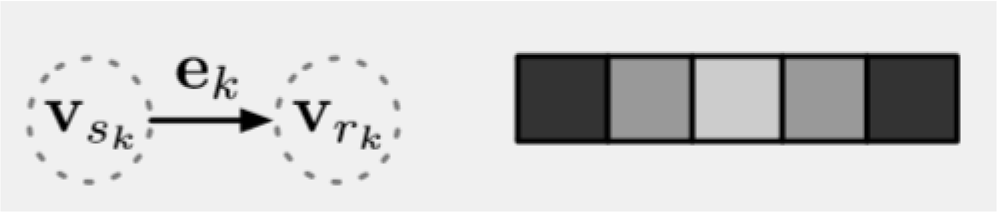


- ✓ Directed : one-way edges, from a "sender" node to a "receiver" node.
- ✓ Attribute : properties that can be encoded as a vector, set, or even another graph
- ✓ Attributed : edges and vertices have attributes associated with them

Node's attribute



Edge's attribute

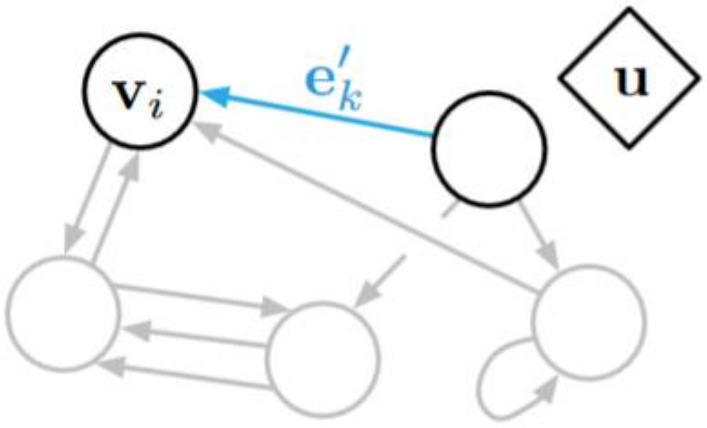


Global attribute

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Graph neural network

GNN blocks



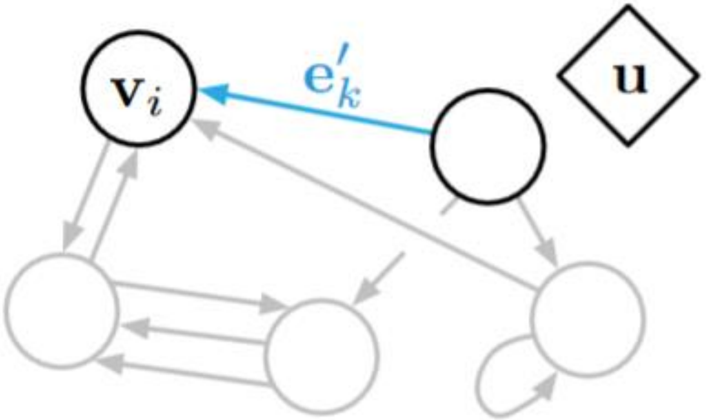
(a) Edge update

$$\mathbf{e}'_k = \mathrm{NN}\left(\mathbf{v}_{s_k}, \mathbf{v}_{r_k}, \mathbf{e}_k, \mathbf{u}\right)$$

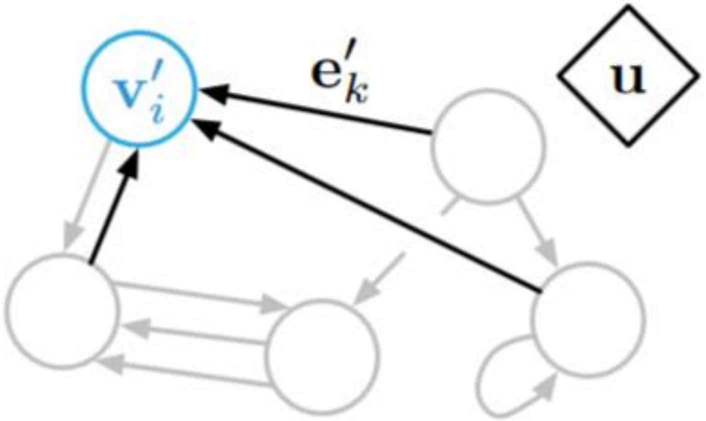
Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Graph neural network

GNN blocks



(a) Edge update



(b) Node update

$$\mathbf{e}'_k = \mathrm{NN}\left(\mathbf{v}_{s_k}, \mathbf{v}_{r_k}, \mathbf{e}_k, \mathbf{u}\right)$$

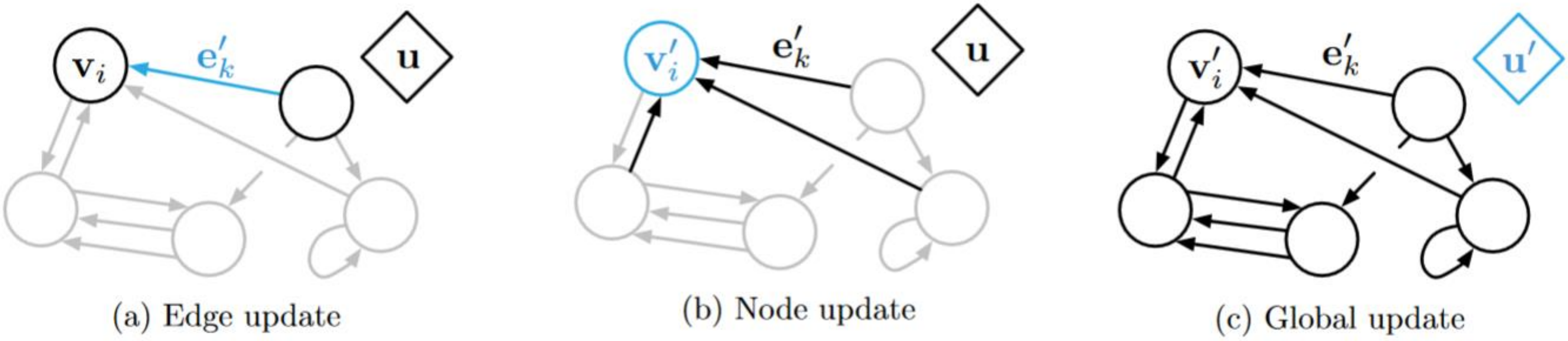
$$\bar{\mathbf{e}}'_i = \sum_{k:r_k=i} \mathbf{e}'_k$$

$$\mathbf{v}'_i = \mathrm{NN}(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u})$$

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Graph neural network

GNN blocks



$$\mathbf{e}'_k = \mathrm{NN}\left(\mathbf{v}_{s_k}, \mathbf{v}_{r_k}, \mathbf{e}_k, \mathbf{u}\right)$$

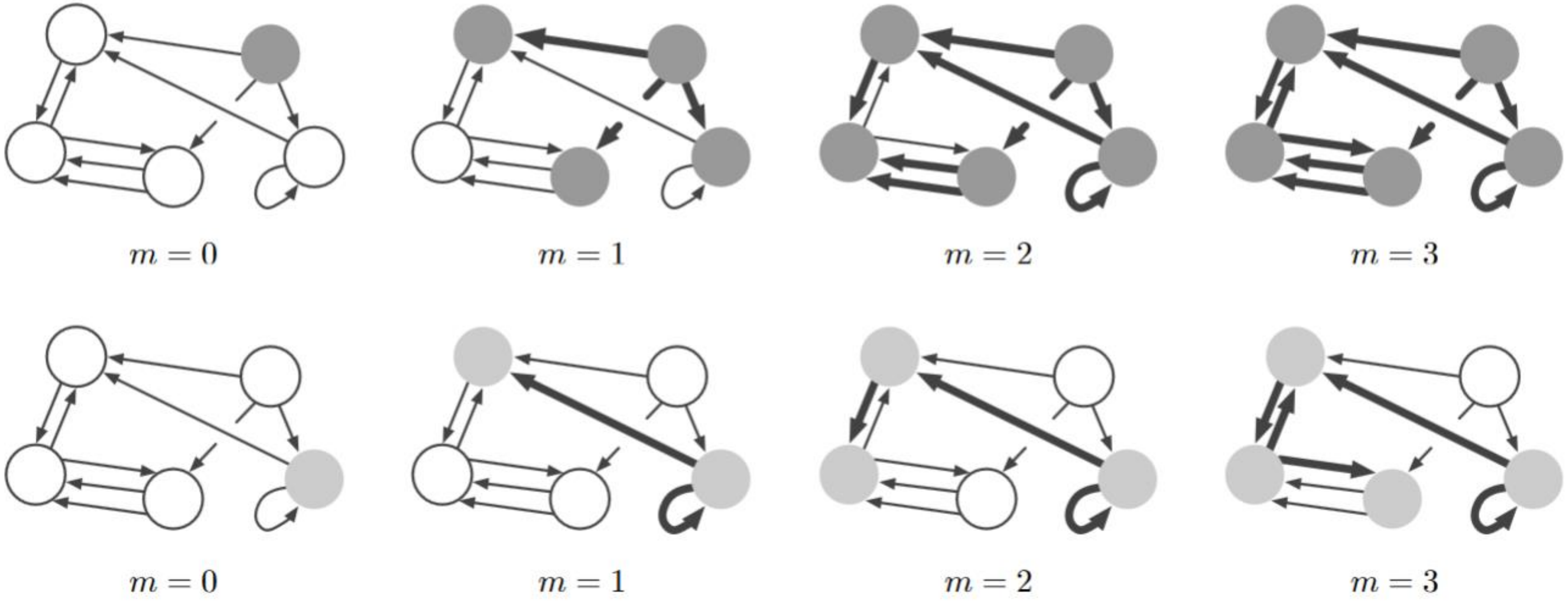
$$\bar{\mathbf{e}}'_i = \sum_{k:r_k=i} \mathbf{e}'_k$$

$$\mathbf{v}'_i = \mathrm{NN}(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u})$$

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Graph neural network

Case) Message passing neural network



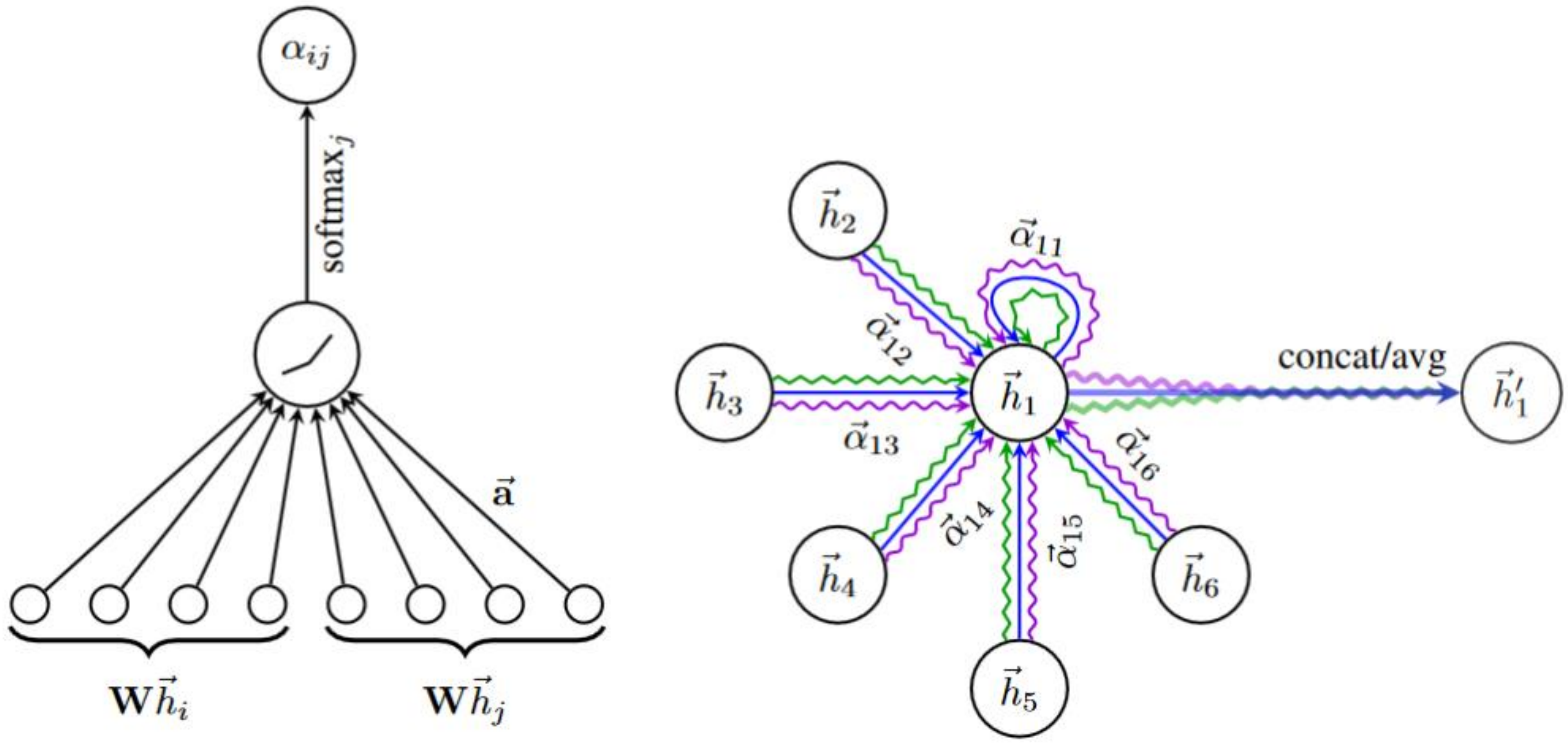
$$\mathbf{v}_i' = \mathrm{GRU}(\mathbf{v}_i, \bar{\mathbf{e}}_i{}') \qquad \bar{e}_i' = \sum_i \mathbf{v}_{s_k}$$

* Note that during the full message passing procedure, this propagation of information happens simultaneously for all nodes and edges in the graph.

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Graph neural network

Case) Graph attention network



$$\mathbf{e}'_k = \mathrm{NN}(\mathbf{v}_{r_k}, \mathbf{v}_{s_k})$$

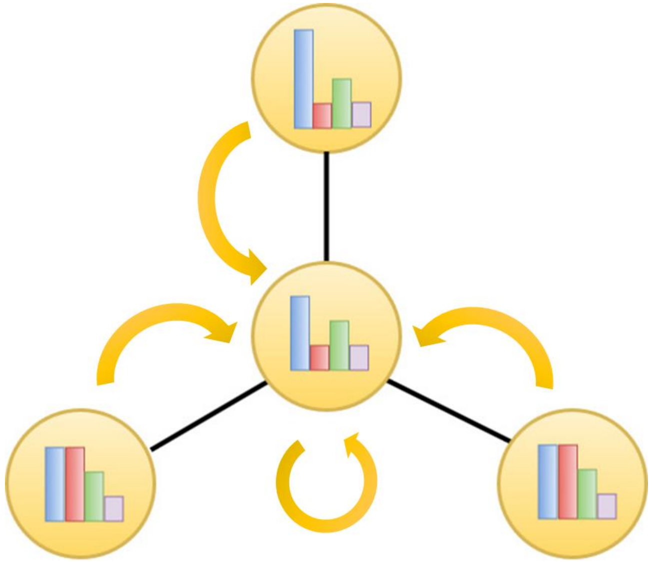
$$\mathbf{v}'_i = \sum_{k:r_k=i} \mathrm{NN}(\mathbf{e}'_k, \mathbf{v}_{s_k})$$

Velickovic, Petar, et al. "Graph attention networks." *arXiv preprint arXiv:1710.10903* 1.2 (2017).

Graph neural network

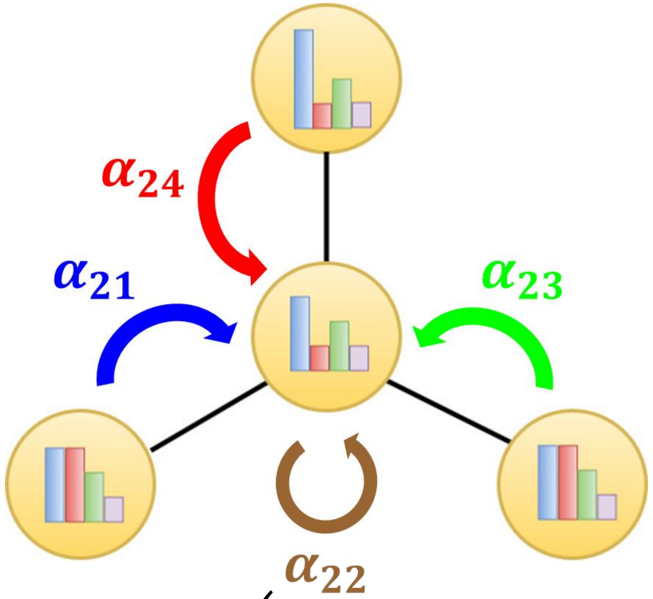
Comparison between GCN and GAT

Vanilla GCN updates information of neighbor atoms **with same importance**.



$$H^{(l+1)} = \sigma\left(\sum_{j\in N(i)} H_j^{(l)} W^{(l)}\right)$$

Attention mechanism enables GCN to update nodes **with different importance.**



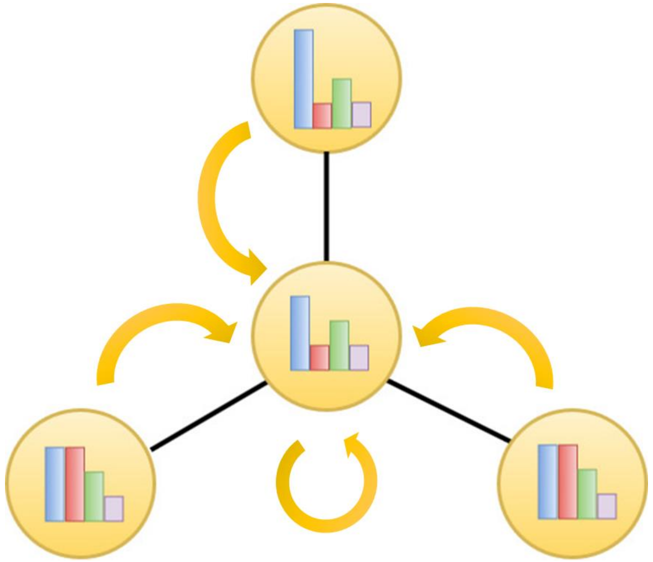
$$H^{(l+1)} = \sigma\left(\sum_{j\in N(i)} \alpha_{ij} H_j^{(l)} W^{(l)}\right)$$

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

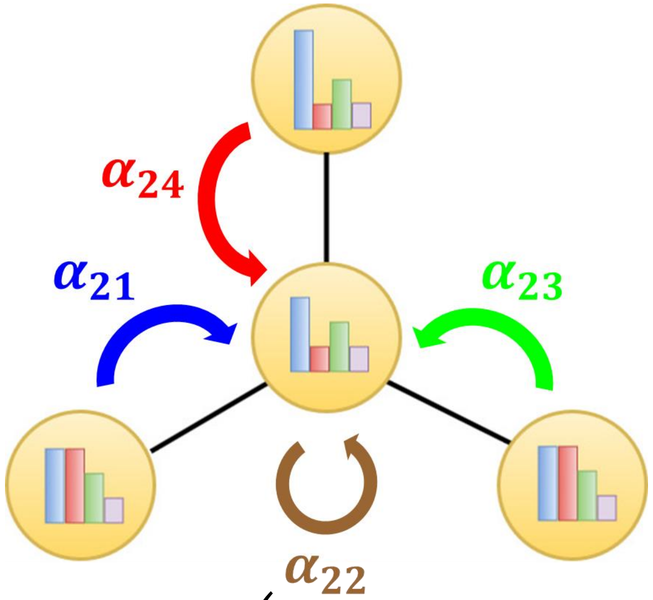
Graph neural network

Comparison between GCN and GAT

The **attention** is nothing but **edge attribute** which find a relation between node attributes



$$H^{(l+1)} = \sigma\left(\sum_{j \in N(i)} H_j^{(l)} W^{(l)}\right)$$

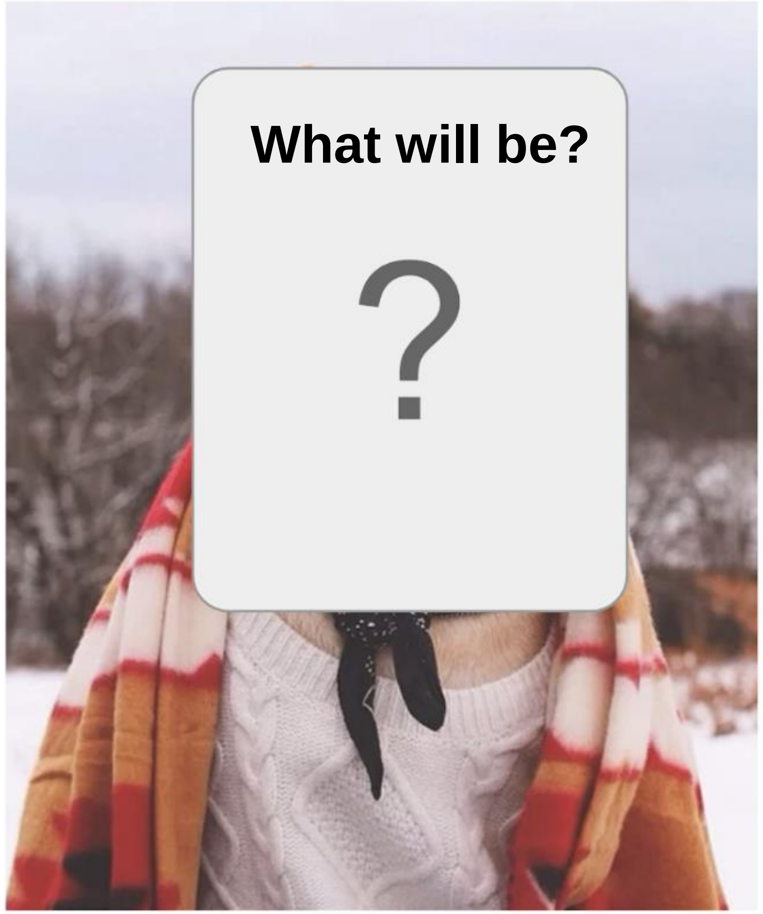


$$H^{(l+1)} = \sigma\left(\sum_{j \in N(i)} \alpha_{ij} H_j^{(l)} W^{(l)}\right)$$

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Attention mechanism

Attention mechanism



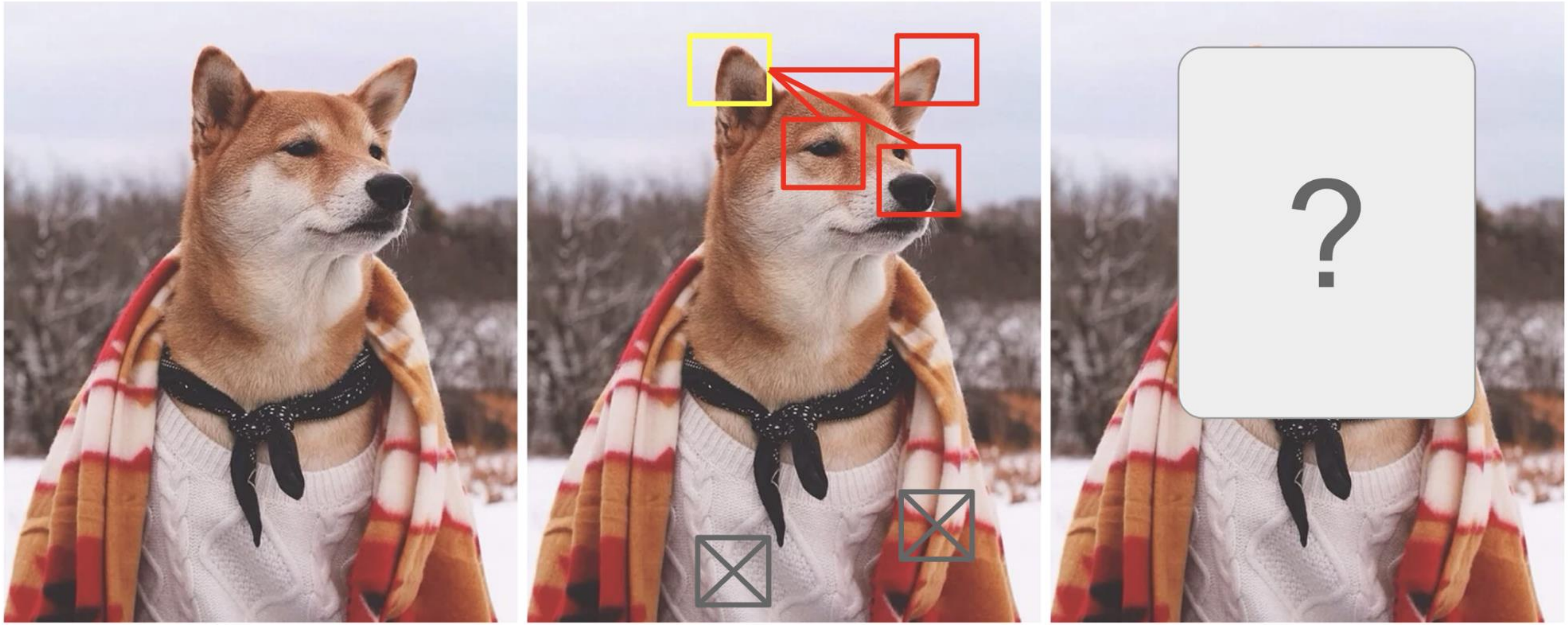
https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

Attention mechanism

https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

Attention mechanism

We deduce something by paying attention to something that is relatively more important.



https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

Attention mechanism

While attention is typically thought of as an orienting mechanism for perception, its "spotlight" can also be focused internally, toward the contents of memory. This idea, a recent focus in neuroscience studies, has also inspired work in AI. In some architectures, **attentional mechanisms have been used to select information to be read out from the internal memory of the network.** This has helped provide recent successes in machine translation and led to important advances on memory and reasoning tasks. These architectures offer a novel implementation of content-addressable retrieval, **which was itself a concept originally introduced to AI from neuroscience.**

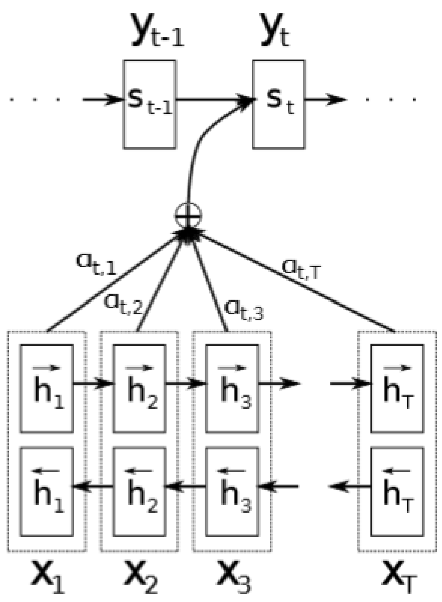
Neuroscience-Inspired Artificial Intelligence

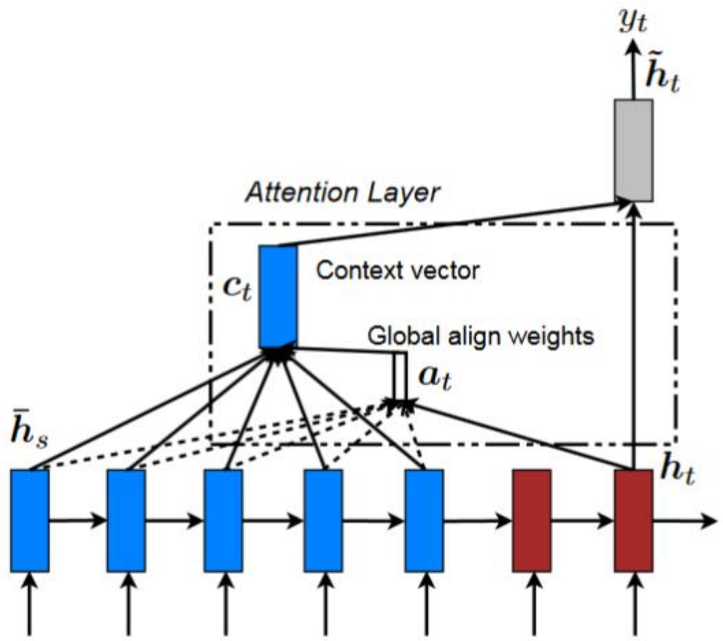
Demis Hassabis,[1,2,*] Dharshan Kumaran,[1,3] Christopher Summerfield,[1,4] and Matthew Botvinick[1,2]
[1]DeepMind, 5 New Street Square, London, UK
[2]Gatsby Computational Neuroscience Unit, 25 Howland Street, London, UK
[3]Institute of Cognitive Neuroscience, University College London, 17 Queen Square, London, UK
[4]Department of Experimental Psychology, University of Oxford, Oxford, UK

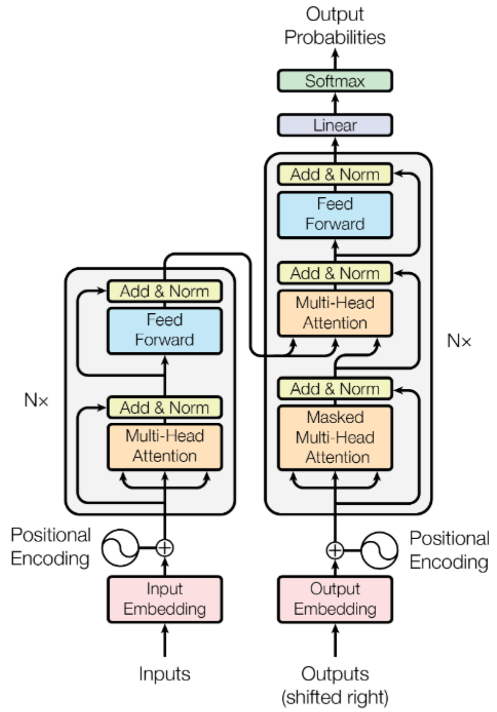
Attention mechanism

Representative papers about the attention mechanism

- ✓ Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).
- ✓ Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).
- ✓ Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.







Seq2Seq

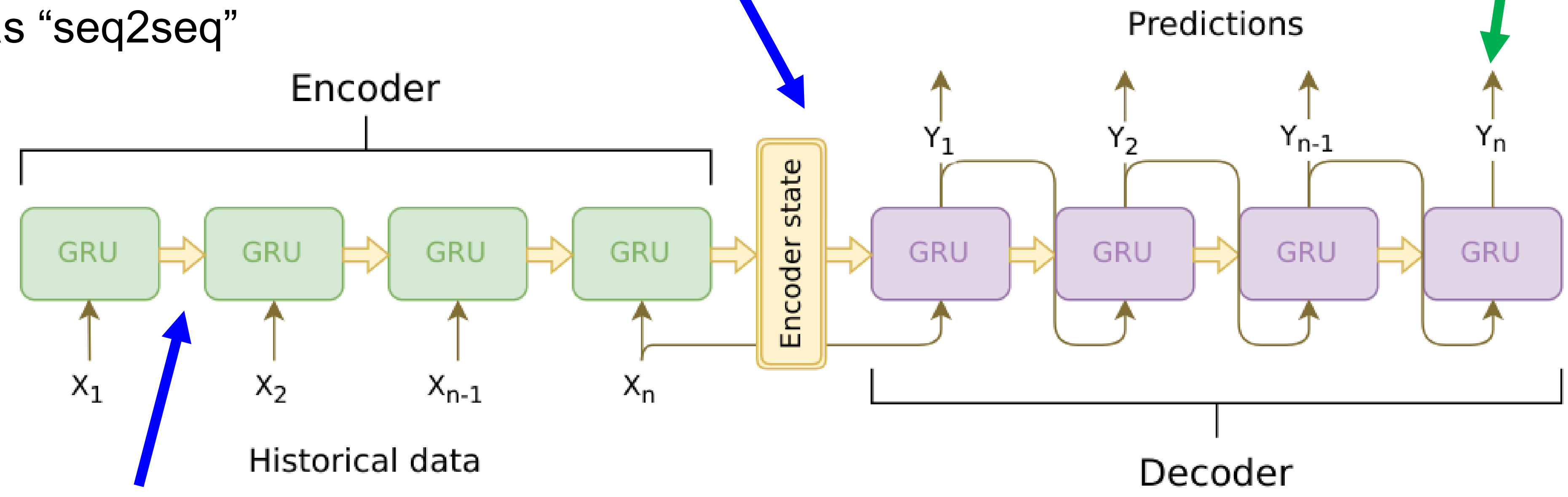
RNN encoder-decoder for neural machine translation

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(\mathbf{y}_t | \{\mathbf{y}_1, \dots, \mathbf{y}_{t-1}\}, \mathbf{c})$$

$$p(\mathbf{y}_t | \{\mathbf{y}_1, \dots, \mathbf{y}_{t-1}\}, \mathbf{c}) = g(\mathbf{y}_{t-1}, \mathbf{s}_t, \mathbf{c})$$

$$\mathbf{c} = q(\{\mathbf{h}_1, \dots, \mathbf{h}_T\}) = \mathbf{h}_T$$

referred as "seq2seq"



$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1}) \in \mathbb{R}^n$: hidden state at time t

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. 2014.

Seq2Seq

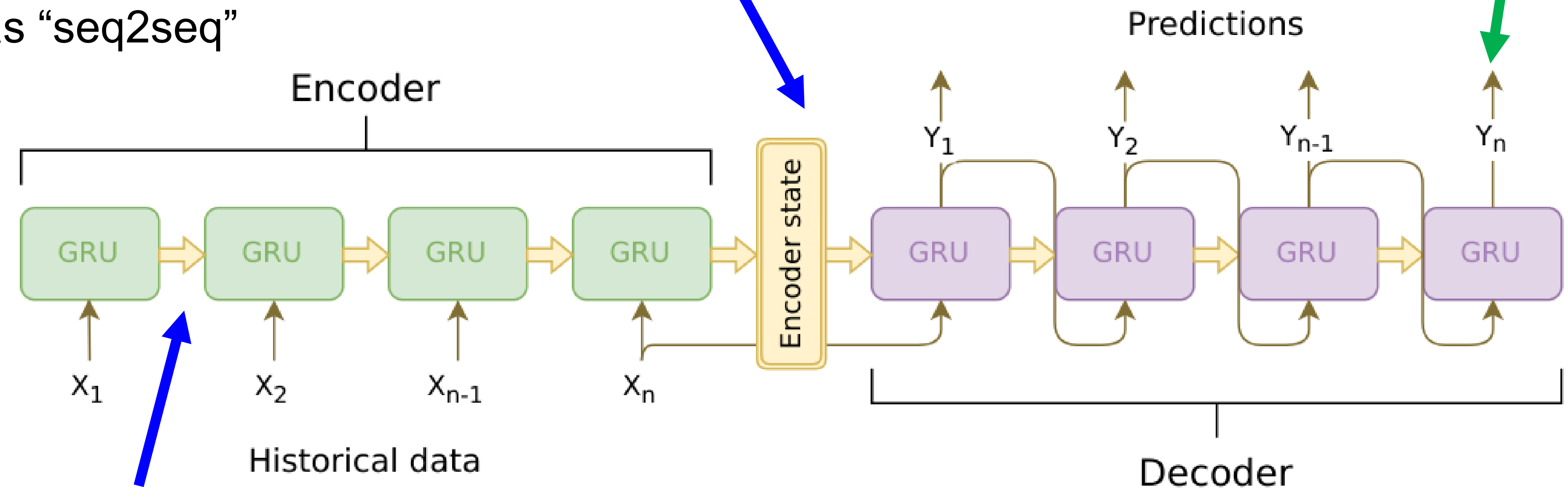
RNN encoder-decoder for neural machine translation

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(\mathbf{y}_t | \{\mathbf{y}_1, \dots, \mathbf{y}_{t-1}\}, \mathbf{c})$$

$$p(\mathbf{y}_t | \{\mathbf{y}_1, \dots, \mathbf{y}_{t-1}\}, \mathbf{c}) = g(\mathbf{y}_{t-1}, \mathbf{s}_t, \mathbf{c})$$

$$\mathbf{c} = q(\{\mathbf{h}_1, \dots, \mathbf{h}_T\}) = \mathbf{h}_T$$

referred as “seq2seq”



$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1}) \in \mathbb{R}^n$: hidden state at time t

Question) What’s wrong with the seq2seq model?

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. 2014.

Seq2Seq

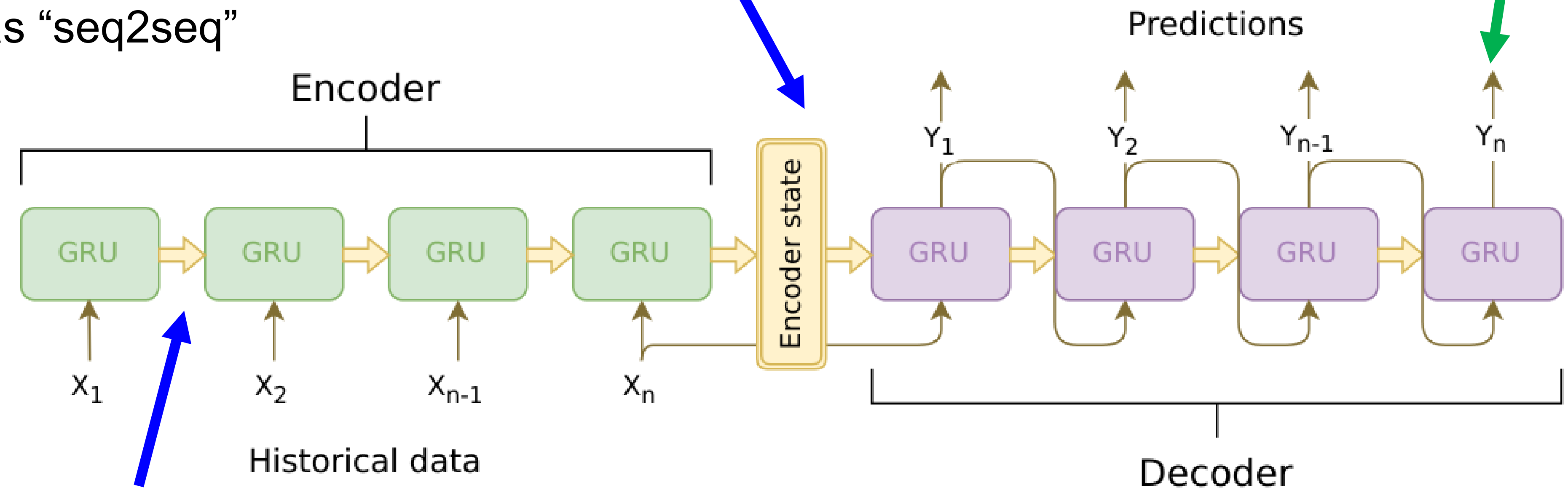
RNN encoder-decoder for neural machine translation

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(\mathbf{y}_t | \{\mathbf{y}_1, \dots, \mathbf{y}_{t-1}\}, \mathbf{c})$$

$$p(\mathbf{y}_t | \{\mathbf{y}_1, \dots, \mathbf{y}_{t-1}\}, \mathbf{c}) = g(\mathbf{y}_{t-1}, \mathbf{s}_t, \mathbf{c})$$

$$\mathbf{c} = q(\{\mathbf{h}_1, \dots, \mathbf{h}_T\}) = \mathbf{h}_T$$

referred as “seq2seq”



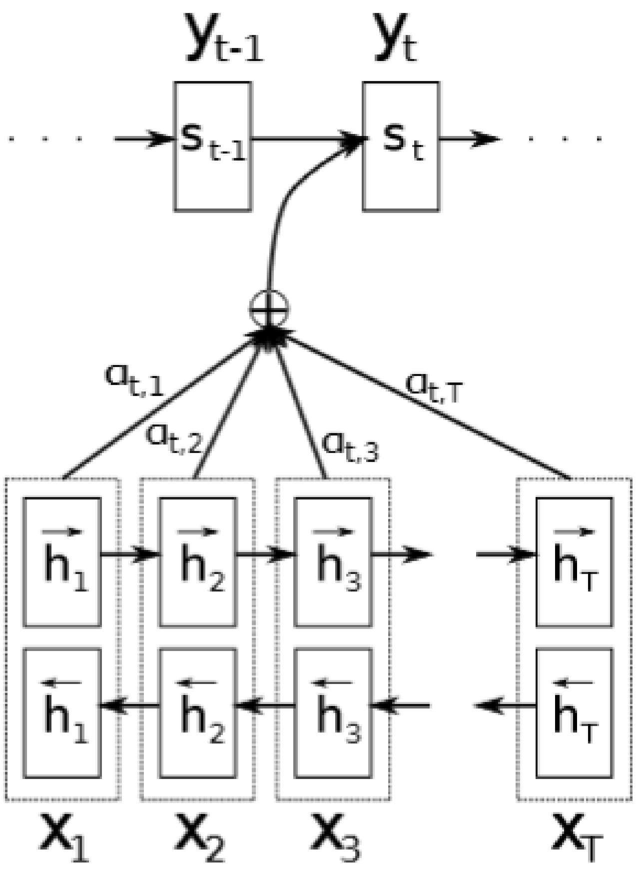
$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1}) \in \mathbb{R}^n$: hidden state at time t

In capability of remembering long sentences : Often it has forgotten the first part once it completes processing the whole input. **The attention mechanism was born to resolve this problem**.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. 2014.

Learning to align and translate

RNN encoder-decoder with an alignment model



In Bahdanau et.al., conditional probability of $\mathbf{y}$ to be predicted is

$$p(\mathbf{y}_t | \{\mathbf{y}_t, \ldots, \mathbf{y}_{t-1}\}, \mathbf{x}) = g(\mathbf{y}_{t-1}, \mathbf{s}_t, \mathbf{c}_t)$$

where $\mathbf{s}_t$ is an RNN hidden state for time t, computed by

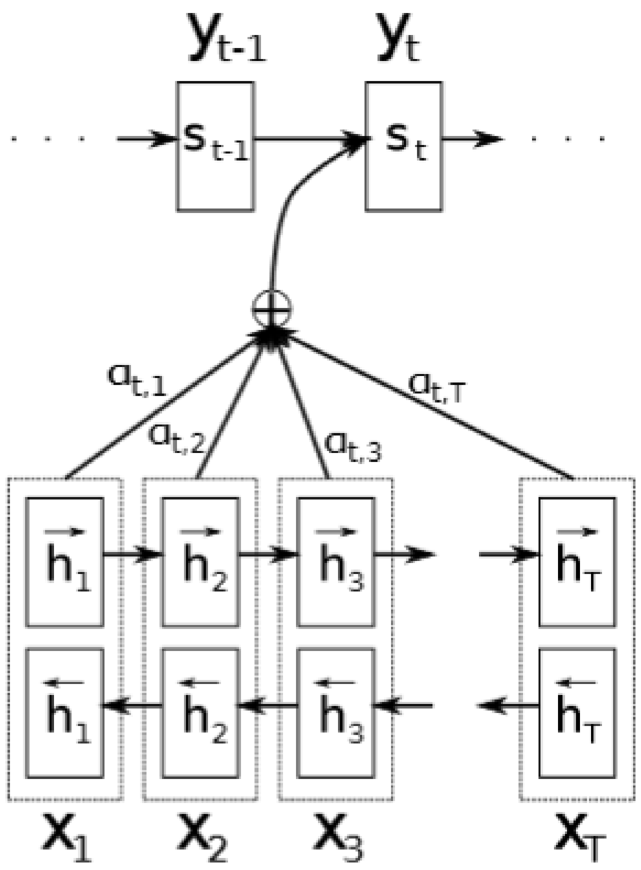
$$\mathbf{s}_t = f(\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t)$$

Question) What is the difference to the seq2seq model?

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).

Learning to align and translate

RNN encoder-decoder with an alignment model



In Bahdanau et.al., conditional probability of $\mathbf{y}$ to be predicted is

$$p(\mathbf{y}_t|\{\mathbf{y}_t, \dots, \mathbf{y}_{t-1}\}, \mathbf{x}) = g(\mathbf{y}_{t-1}, \mathbf{s}_t, \mathbf{c}_t)$$

where $\mathbf{s}_t$ is an RNN hidden state for time t, computed by

$$\mathbf{s}_t = f(\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t)$$

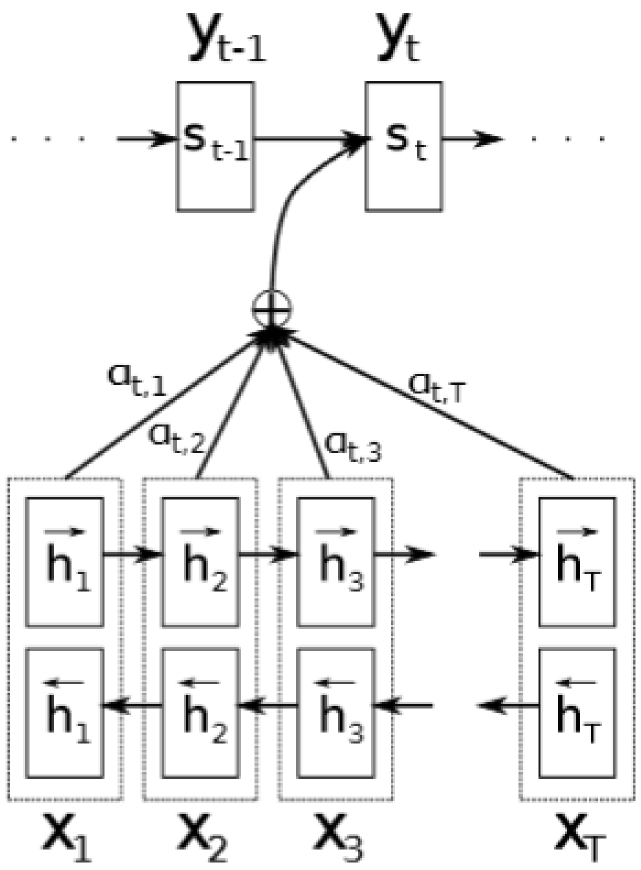
Question) What is the difference to the seq2seq model?

Answer) Here the probability is conditioned on a distinct context vector $\mathbf{c}_t$ for each target word $\mathbf{y}_t$.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).

Learning to align and translate

RNN encoder-decoder with an alignment model



In Bahdanau et.al., conditional probability of $\mathbf{y}$ to be predicted is

$$p(\mathbf{y}_t | \{\mathbf{y}_t, \dots, \mathbf{y}_{t-1}\}, \mathbf{x}) = g(\mathbf{y}_{t-1}, \mathbf{s}_t, \mathbf{c}_t)$$

where $\mathbf{s}_t$ is an RNN hidden state for time t, computed by

$$\mathbf{s}_t = f(\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t)$$

$$\mathbf{c}_t = \sum_{j=1}^{T} \alpha_{tj} \mathbf{h}_j \qquad \alpha_{tj} = \frac{\exp(e_{tj})}{\sum_{k=1}^{T} \exp(e_{tk})}$$

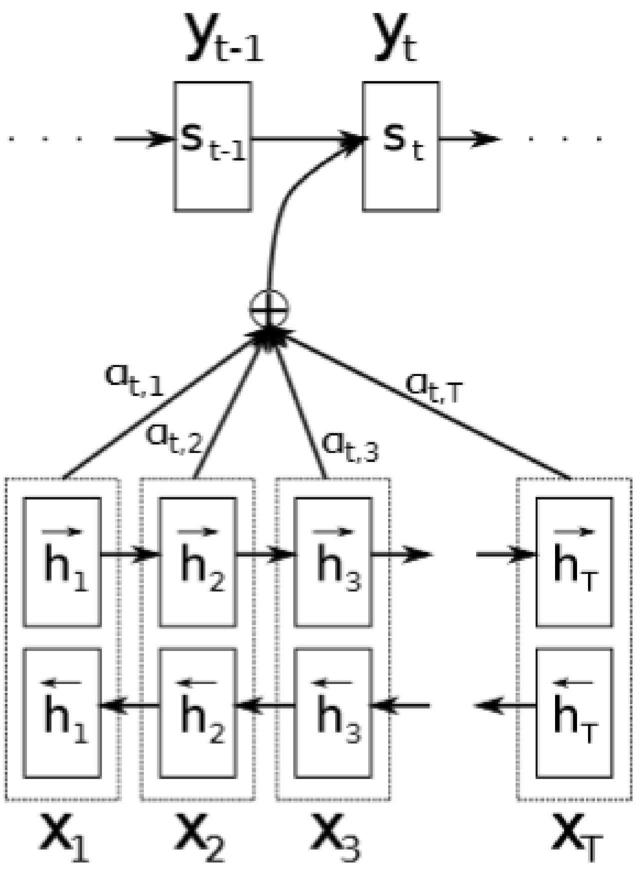
$$e_{tj} = a(\mathbf{s}_{t-1}, \mathbf{h}_j) = \mathbf{v}_a^T \tanh(\mathbf{W}_a \mathbf{s}_{t-1} + \mathbf{U}_a \mathbf{h}_j)$$

Question) What lessons can we get from this model?

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).

Learning to align and translate

RNN encoder-decoder with an alignment model



In Bahdanau et.al., conditional probability of $\mathbf{y}$ to be predicted is

$$p(\mathbf{y}_t | \{\mathbf{y}_t, \dots, \mathbf{y}_{t-1}\}, \mathbf{x}) = g(\mathbf{y}_{t-1}, \mathbf{s}_t, \mathbf{c}_t)$$

where $\mathbf{s}_t$ is an RNN hidden state for time t, computed by

$$\mathbf{s}_t = f(\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t)$$

$$\mathbf{c}_t = \sum_{j=1}^{T} \alpha_{tj} \mathbf{h}_j \qquad \alpha_{tj} = \frac{\exp(e_{tj})}{\sum_{k=1}^{T} \exp(e_{tk})}$$

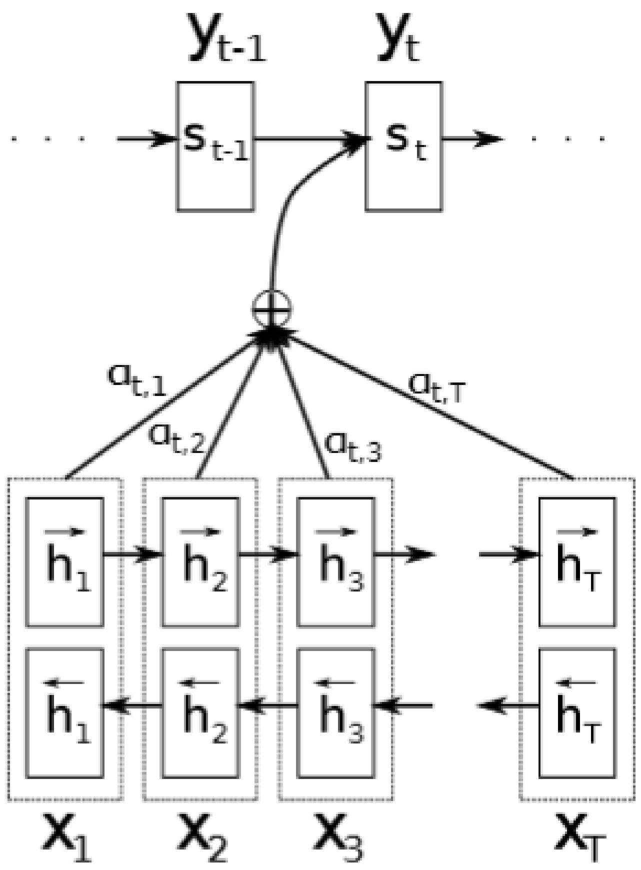
$$e_{tj} = a(\mathbf{s}_{t-1}, \mathbf{h}_j) = \mathbf{v}_a^T \tanh(\mathbf{W}_a \mathbf{s}_{t-1} + \mathbf{U}_a \mathbf{h}_j)$$

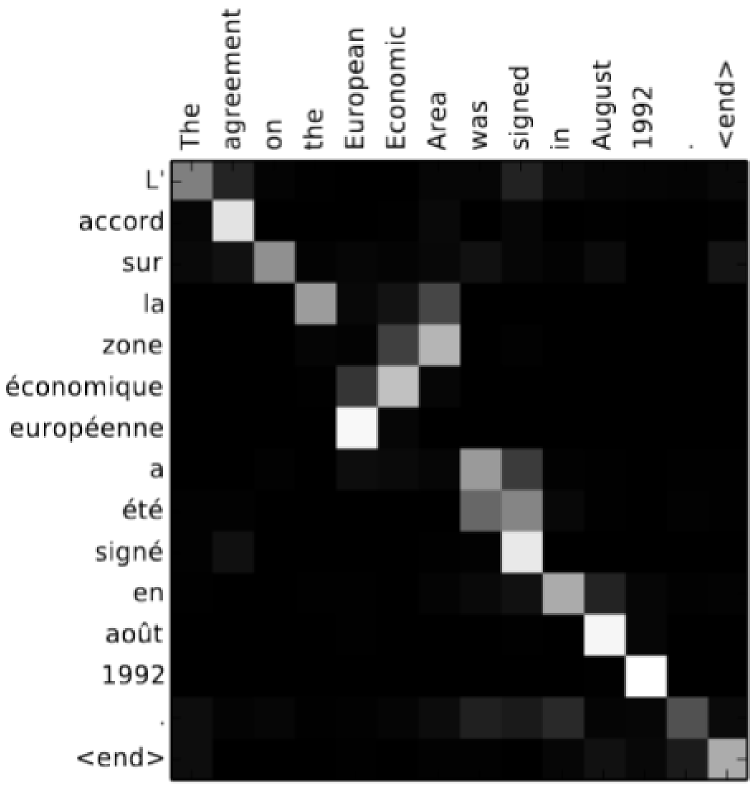
This is an ***alignment model*** which scores **how well the inputs around position *j* and the output at position *i* match**.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).

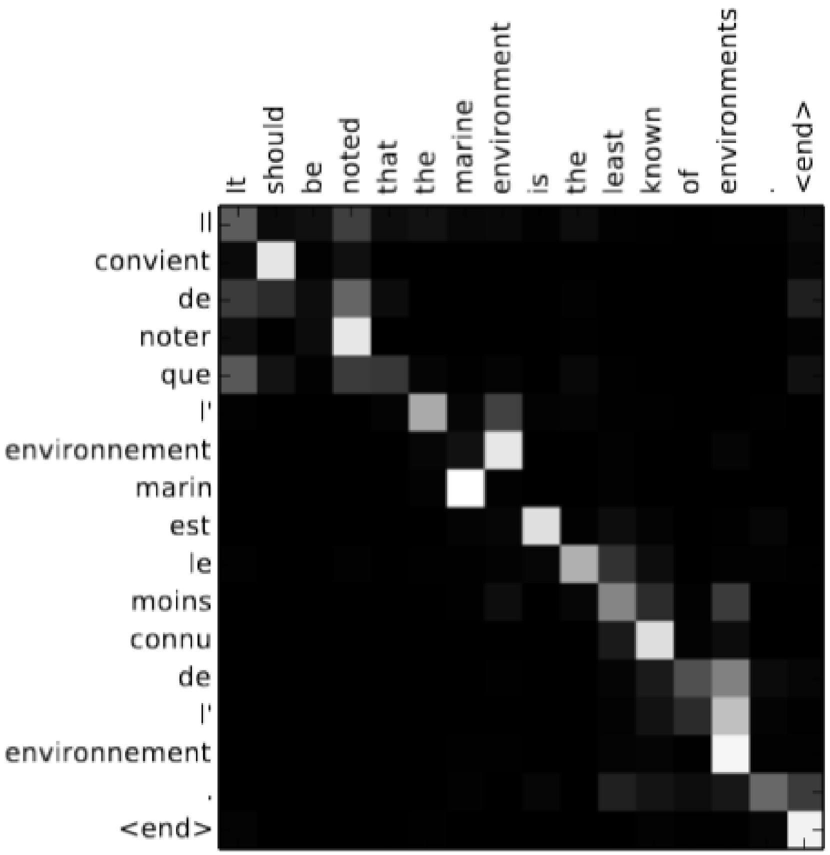
Learning to align and translate

RNN encoder-decoder with an alignment model





(a)

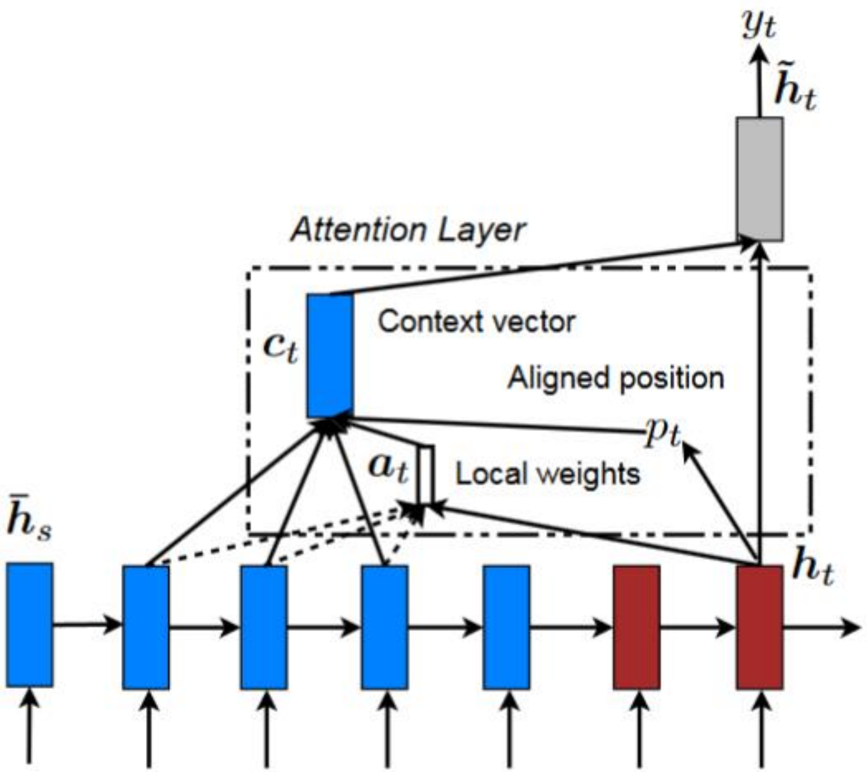


(b)

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).

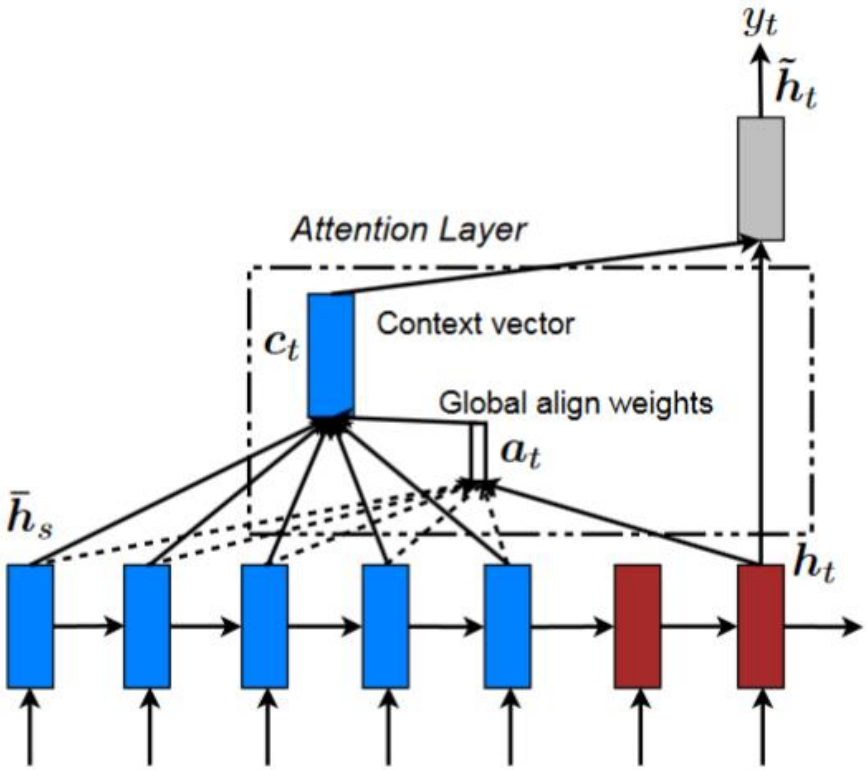
Local and Global attention

Local attention



Only looks at a subset of source words at a time

Global attention

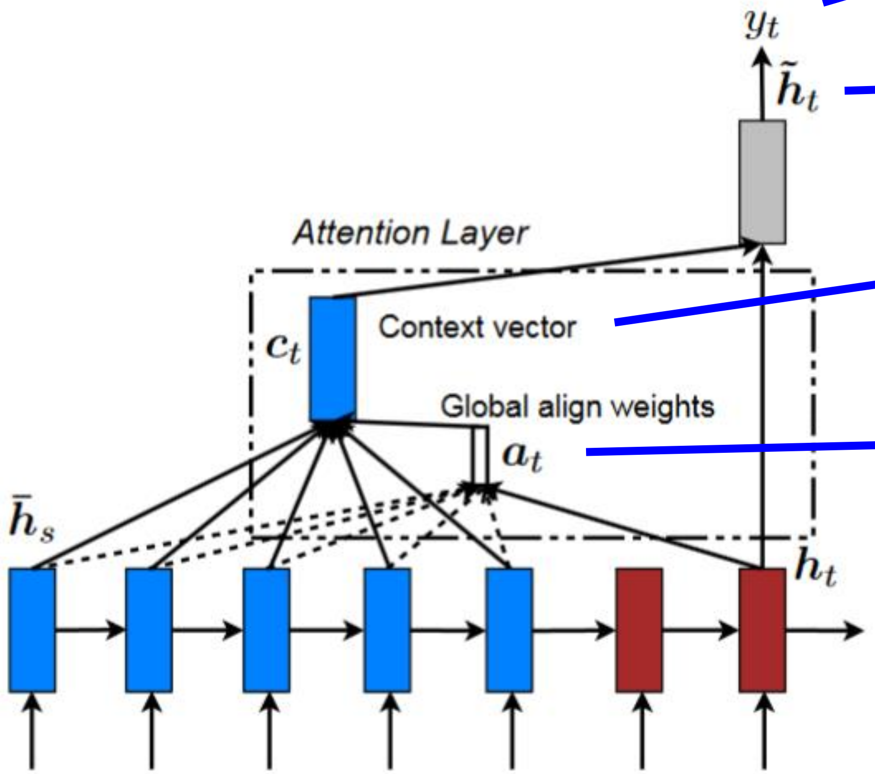


Always attends to all source words

Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).

Local and Global attention

Global attention



$$p(\mathbf{y}_t|\mathbf{y}_{<t},\mathbf{x}) = \mathrm{softmax}\left(\mathbf{W}_s\tilde{\mathbf{h}}_t\right)$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_c[\mathbf{c}_t;\mathbf{h}_t])$$

$$\mathbf{c}_t = \sum_{s'} a_t(s')\bar{\mathbf{h}}_{s'}$$

$$a_t(s) = \mathrm{align}\left(\mathbf{h}_t,\bar{\mathbf{h}}_s\right) = \frac{\exp\left(\mathrm{score}\left(\mathbf{h}_t,\bar{\mathbf{h}}_s\right)\right)}{\sum_{s'}\exp\left(\mathrm{score}\left(\mathbf{h}_t,\bar{\mathbf{h}}_{s'}\right)\right)}$$

Always attends to all source words

Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).

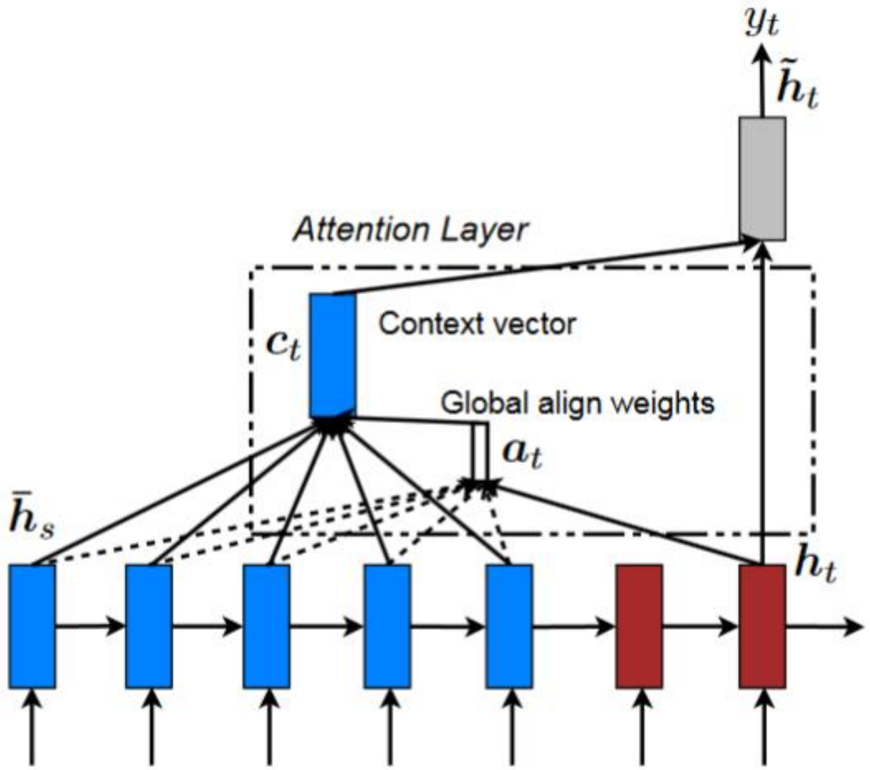
Local and Global attention

Name	Alignment score function	Citation
Additive	$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{v}_a^T \tanh(\mathbf{W}_a[\mathbf{s}_t; \mathbf{h}_i])$	Bahdanau 2015
Location-Base	$\alpha_{t,i} = \text{softmax}(\mathbf{W}_a \mathbf{s}_t)$ Note : This simplifies the softmax alignment max to only depend on the target position.	Luong 2015
General	$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{s}_t^T \mathbf{W}_a \mathbf{h}_i$	Luong 2015
Dot-Product	$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{s}_t^T \mathbf{h}_i$	Luong 2015
Scaled Dot-Product	$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \frac{\mathbf{s}_t^T \mathbf{h}_i}{\sqrt{n}}$ where n is the dimension of source hidden state	Vaswani 2017
Self-Attention	Relating different positions of the same input sequence. Theoretically the self-attention can adopt any score functions above, just replace	Cheng 2016
Global/Soft	Attending to the entire input state space	Xu 2015
Local/Hard	Attending to the part of input state space; i.e. a patch of the input image.	Xu 2015 Luong 2015

https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

Local and Global attention

Global attention



Always attends to all source words

The global attention has a drawback that

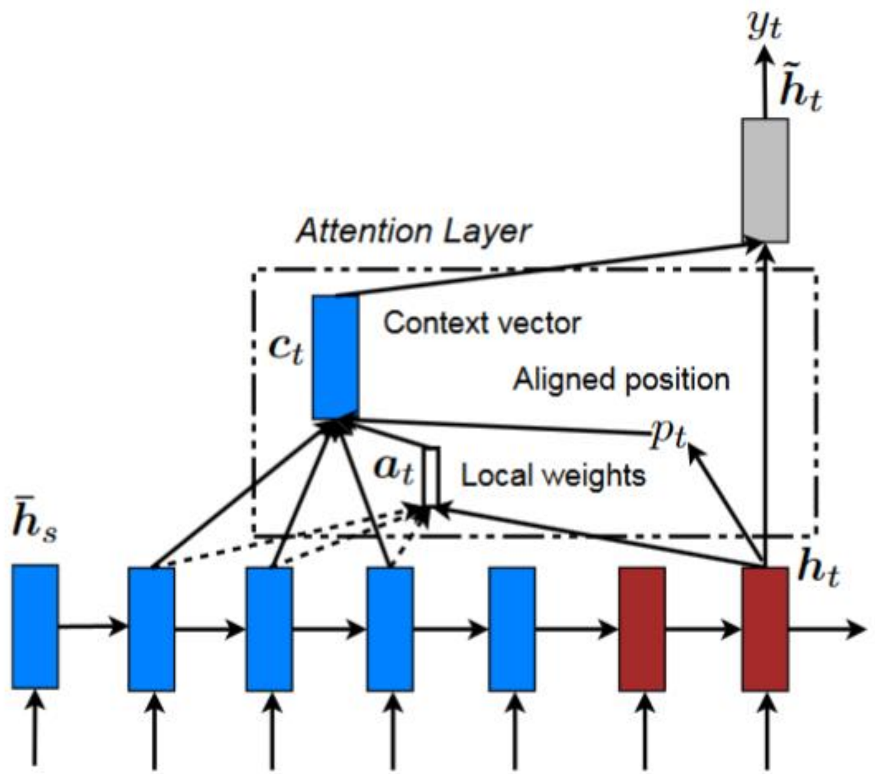
✓ It has to attend to all words on the source side for each target word,

✓ Which is expensive and can potentially render it impractical to translate longer sequences, e.g., paragraphs or documents.

Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).

Local and Global attention

Local attention



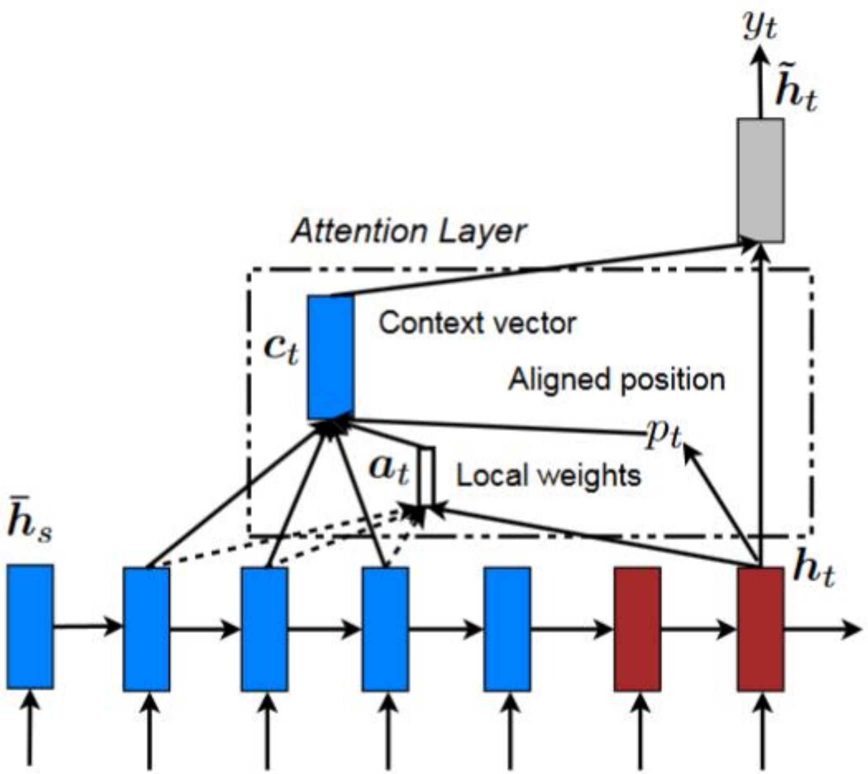
- ✓ Local attention mechanism that chooses to **focus only on a small subset of the source positions** per target word.
- ✓ This selectively **focuses on a small window of context** and is differentiable.
- ✓ The model first generates **an aligned position $\boldsymbol{p_t}$** for each target word at time t. The context vector c_t is then derived as a weighted average over the set of source hidden states within the window $[p_t - D : p_t + D]$.

Only looks at a subset of source words at a time

Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).

Local and Global attention

Local attention



Only looks at a subset of source words at a time

Monotonic alignment (**local-m**)
: set $p_t = t$ assuming that source and target sequences are roughly monotonically aligned. The alignment vector is

$$a_t(s) = \text{align}(\mathbf{h}_t, \bar{\mathbf{h}}_s) = \frac{\exp\left(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_s)\right)}{\sum_{s'} \exp\left(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_{s'})\right)}$$

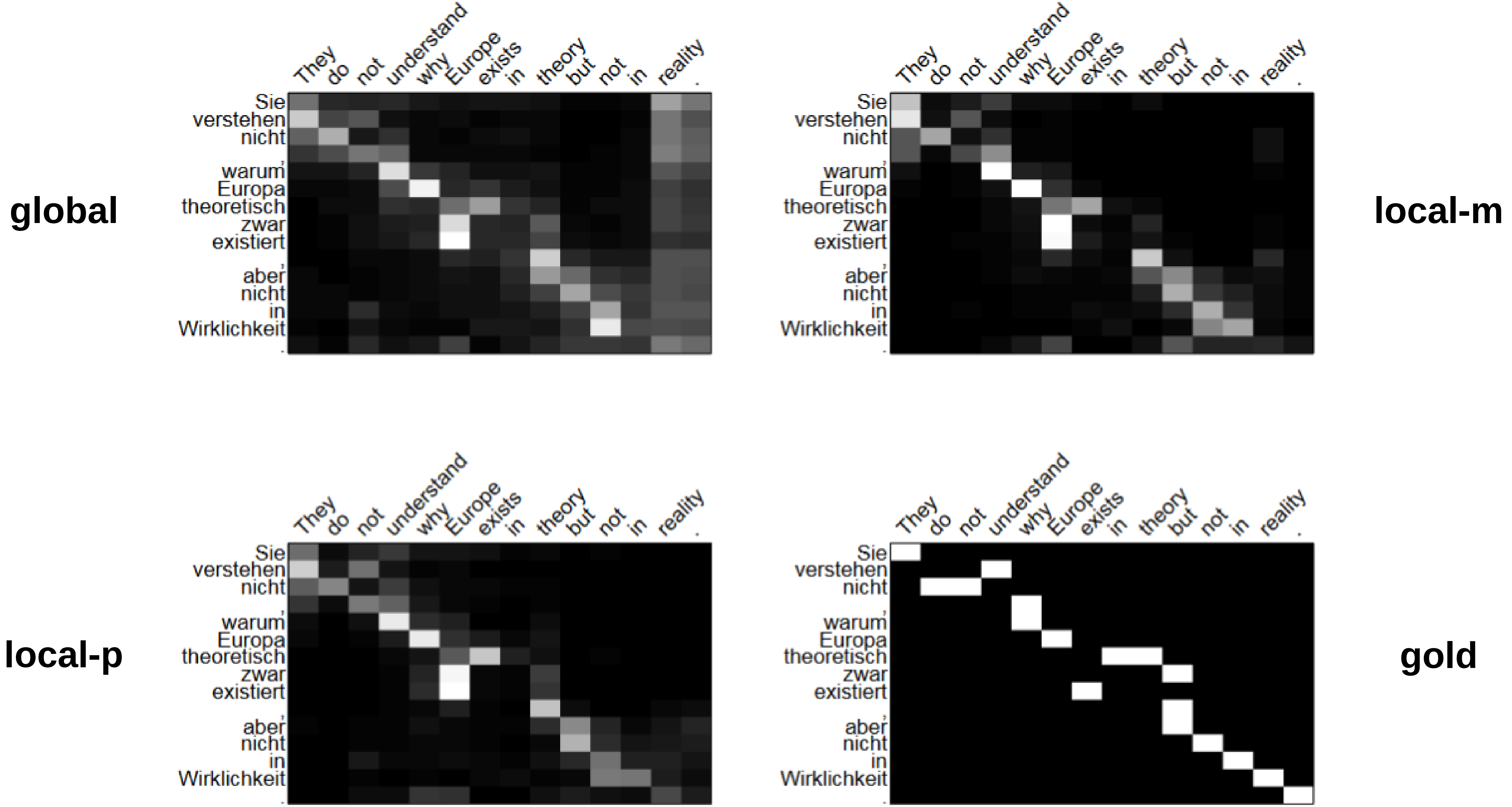
Predictive alignment (**local-p**)
: instead of assuming monotonic alignments, the model predicts an aligned position as follows:

$$p_t = S \cdot \text{sigmoid}\left(\mathbf{v}_p^T \tanh(\mathbf{W}_p \mathbf{h}_t)\right)$$

$$a_t(s) = \text{align}(\mathbf{h}_t, \bar{\mathbf{h}}_s) \exp\left(-\frac{(s - p_t)^2}{2\sigma^2}\right)$$

Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).

Local and Global attention



Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).

Transformer

Attention Is All You Need

The most impactful and interesting paper in 2017

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

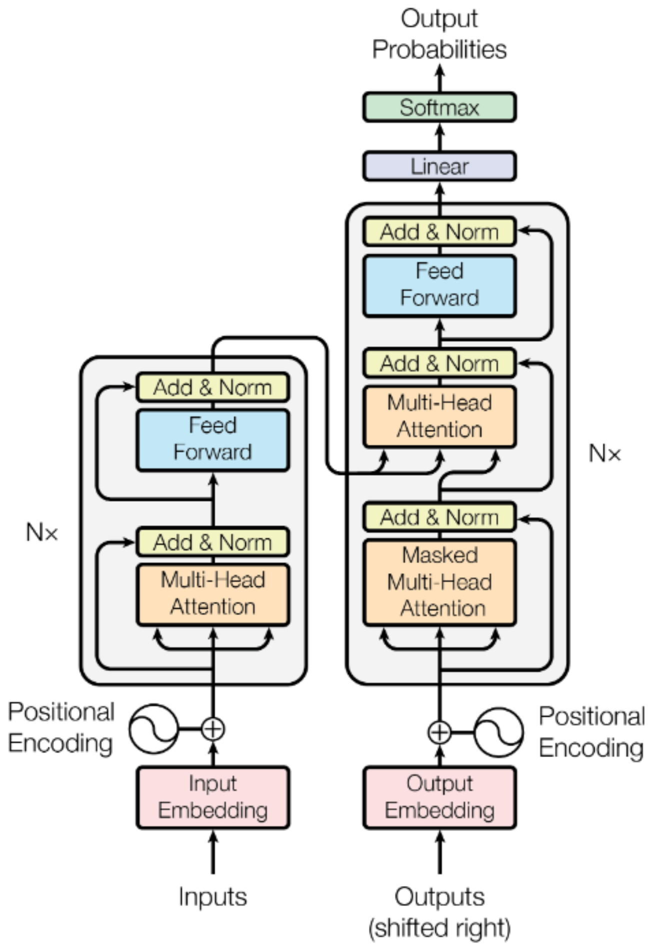
Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

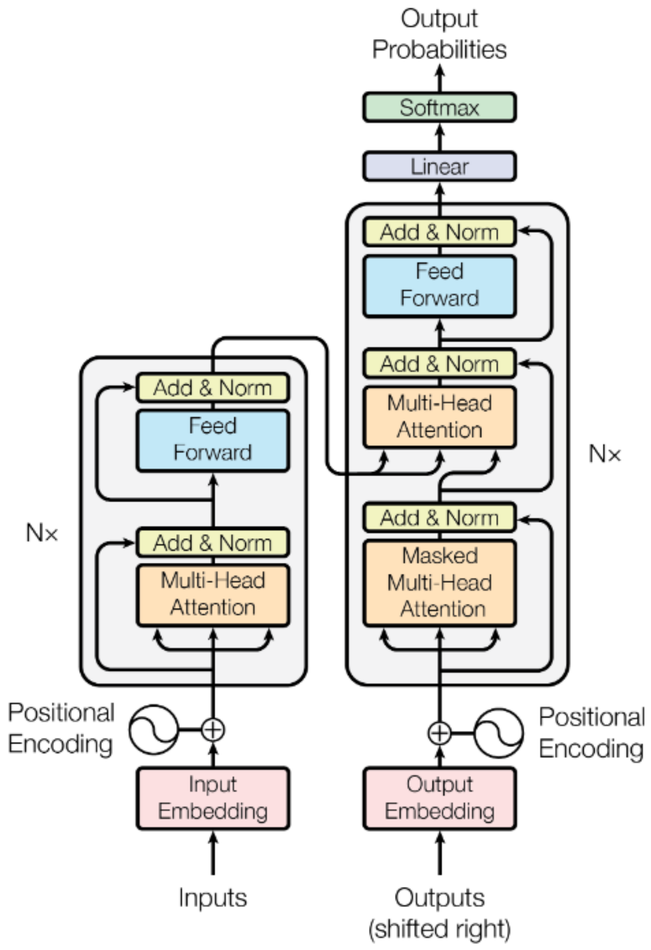


Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

Before showing details of the transformer

Q1) Does the transformer use recurrent network units?
A1) No.



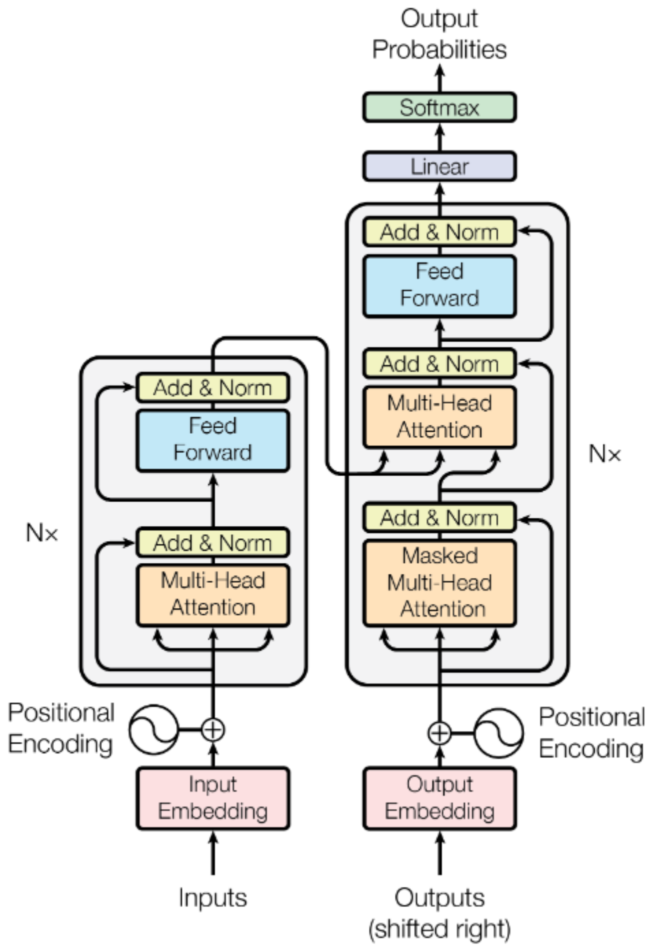
Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

Before showing details of the transformer

Q1) Does the transformer use recurrent network units?
A1) No.

Q2) What operations are used in the transformer?
A3) Only using MLP.



Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

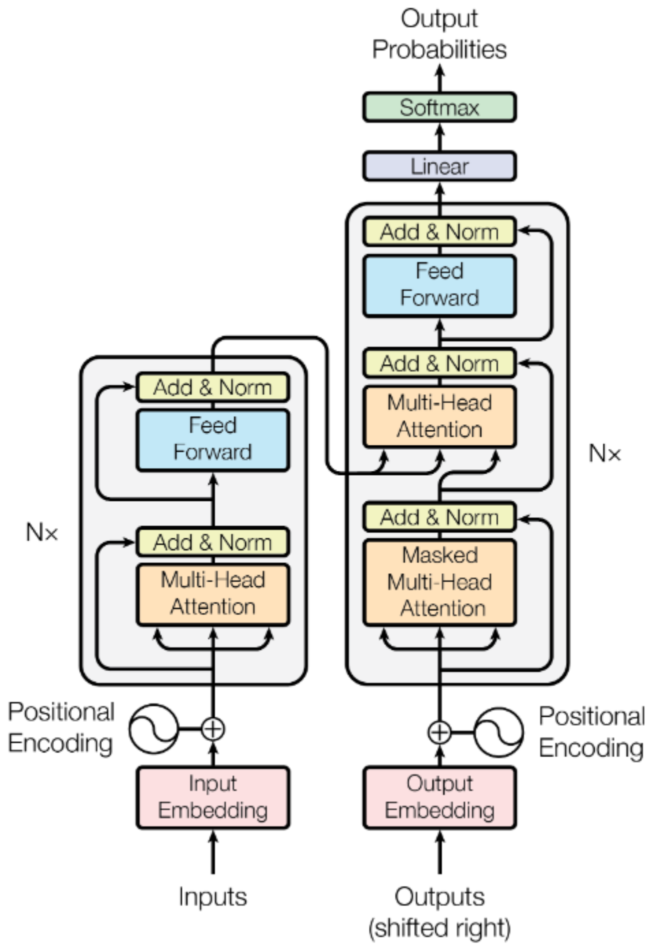
Transformer

Before showing details of the transformer

Q1) Does the transformer use recurrent network units?
A1) No.

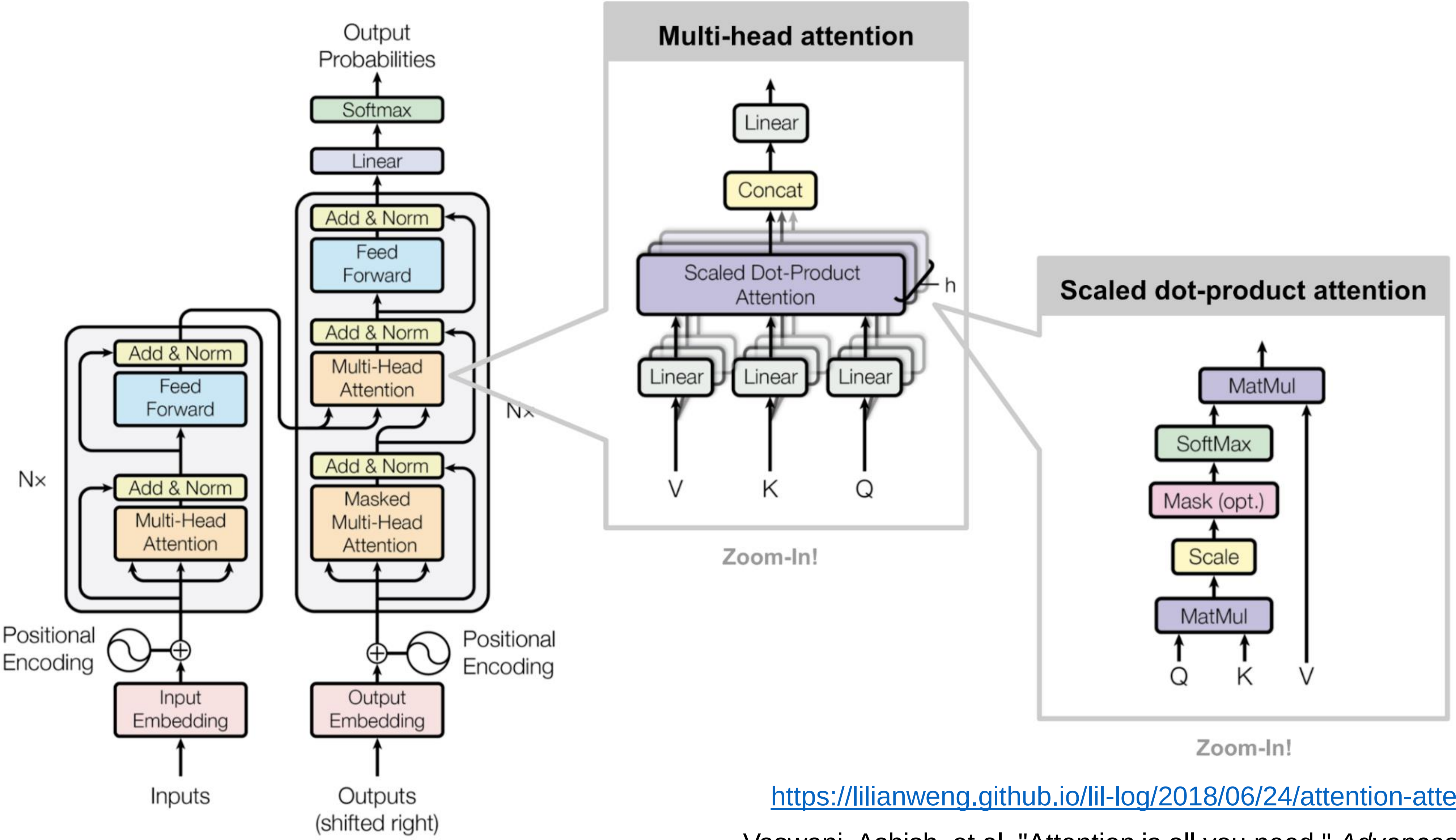
Q2) What operations are used in the transformer?
A3) Only using MLP.

Q3) Is it possible?
A3) Yes, using MLP with attention is enough.
“Attention is all you need!”



Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

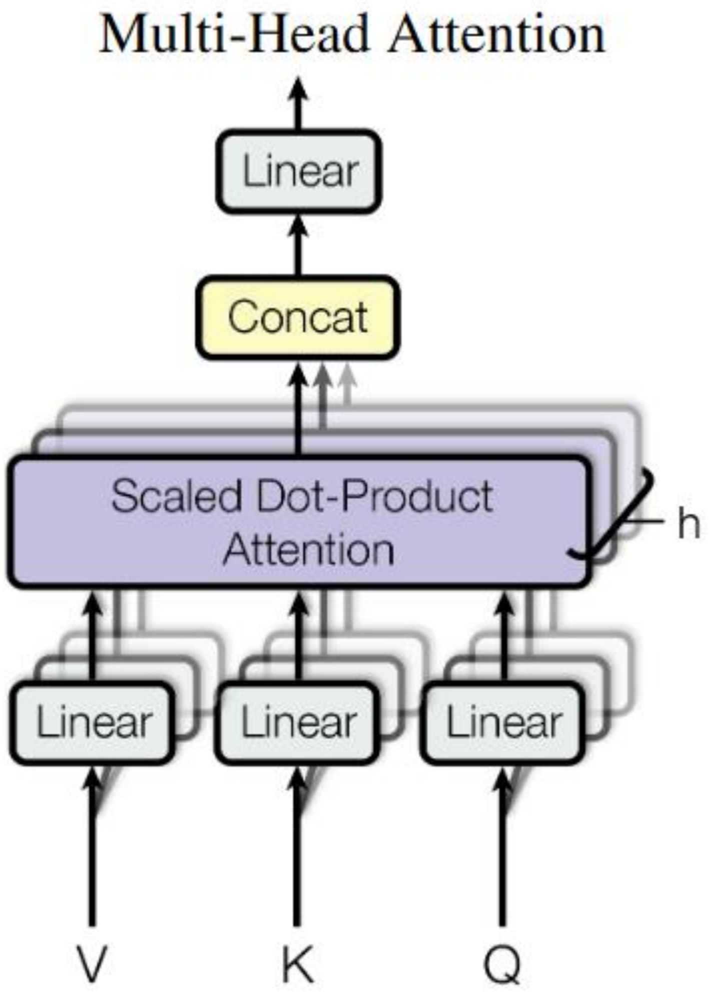


https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

Self-attention



An attention function can be described as mapping a query and a set of key-value pairs to an output

- ✓ Key-Value : encoder hidden states
- ✓ Query : the previous output in the decoder

$$\mathrm{Multihead}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_\mathrm{h})\mathbf{W}^O$$

$$\mathrm{head}_\mathrm{i} = \mathrm{Attention}\left(\mathbf{QW}_i^Q, \mathbf{KW}_i^K, \mathbf{VW}_i^V\right)$$

$\mathbf{W}_i^Q \in \mathbb{R}^{d_{model} \times d_k}$ $\mathbf{W}_i^K \in \mathbb{R}^{d_{model} \times d_k}$

$\mathbf{W}_i^V \in \mathbb{R}^{d_{model} \times d_v}$

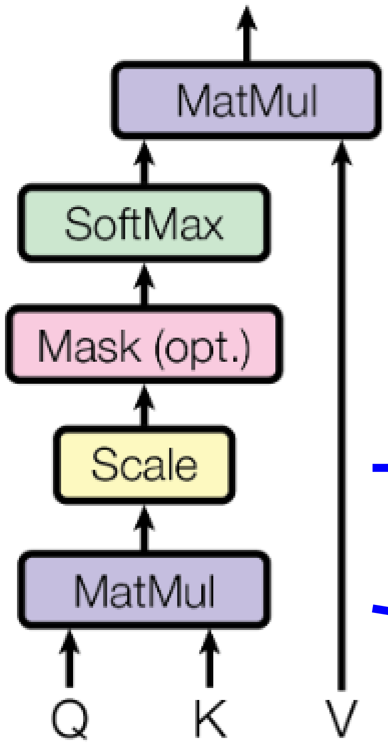
$\mathbf{W}^O \in \mathbb{R}^{hd_v \times d_{model}}$

Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

Self-attention

Scaled Dot-Product Attention



$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \in \mathbb{R}^{d_{model} \times d_v}$$

$$\text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)$$

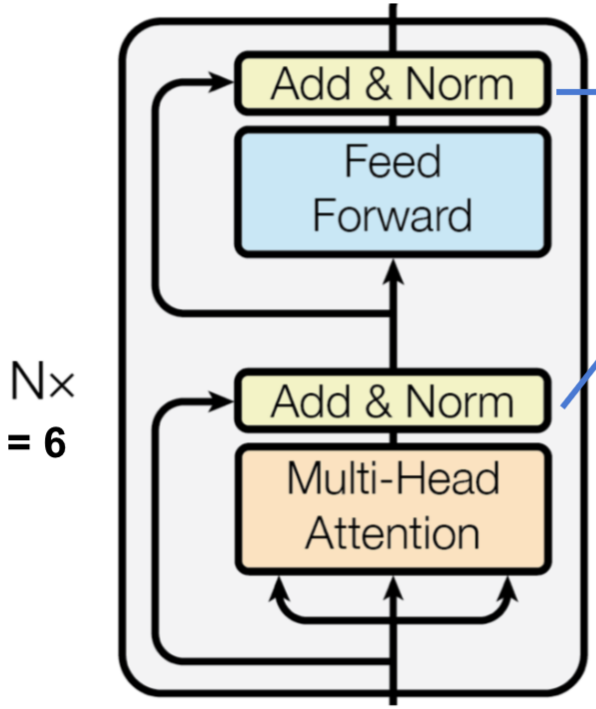
$\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}$: assume that the components of q and k are independent random variables with mean 0 and variance 1. Then their dot product, $q \cdot k = \sum_{i=1}^{d_k} q_i k_i$, has mean 0 and variance d_k.

$\mathbf{Q}\mathbf{K}^T \in \mathbb{R}^{d_{model} \times d_{model}}$: dot-product attention is much faster and more space-efficient in practice.

Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

Encoder



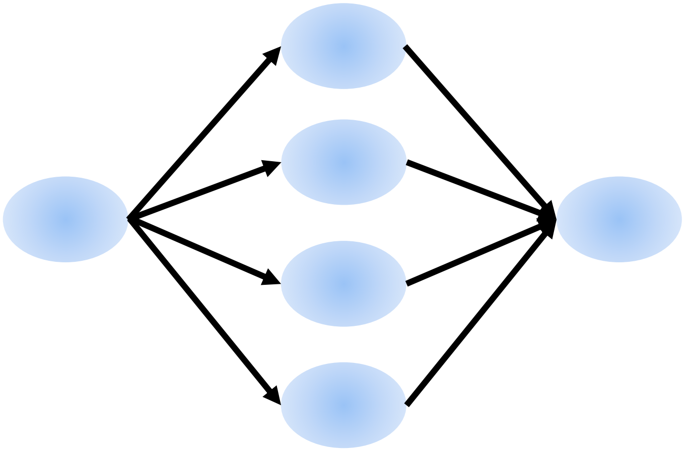
LayerNorm(x + SubLayer(x))

$$\mathrm{FFN}(\mathbf{x}) = \mathrm{ReLU}(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2$$

$$\mathbf{x} \in \mathbb{R}^{d_{model} \times d_v}$$

$$\mathbf{W}_1 \in \mathbb{R}^{d_v \times m * d_v}$$

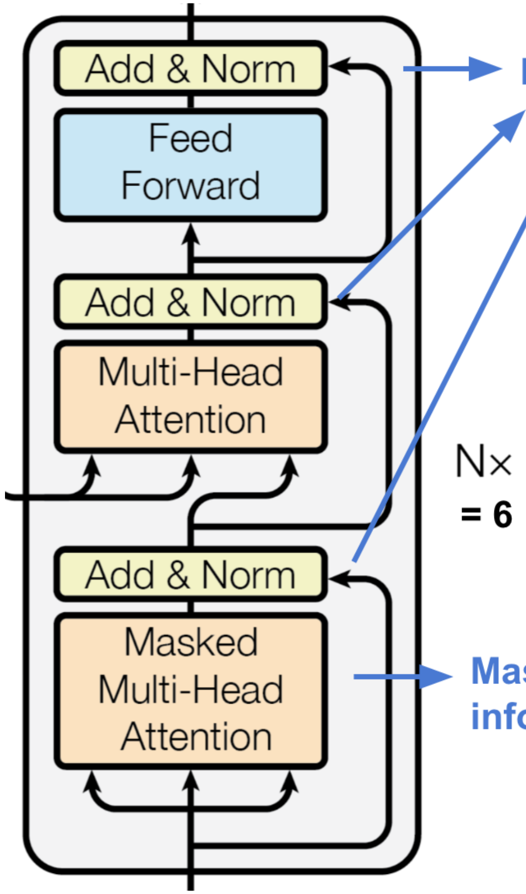
$$\mathbf{W}_2 \in \mathbb{R}^{m * d_v \times d_v}$$



Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

Decoder

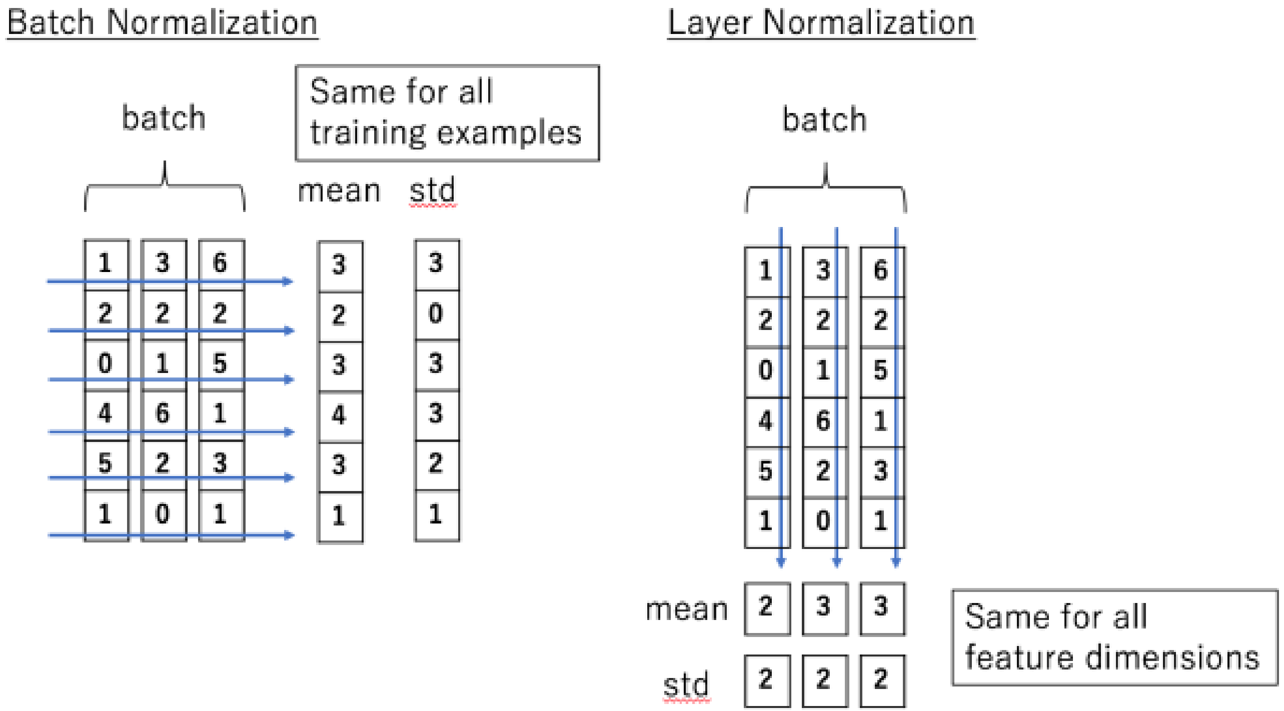


LayerNorm(x + SubLayer(x))

Masked: not to use the information in the future.

Layer normalization

✓ Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." *arXiv preprint arXiv:1607.06450* (2016).



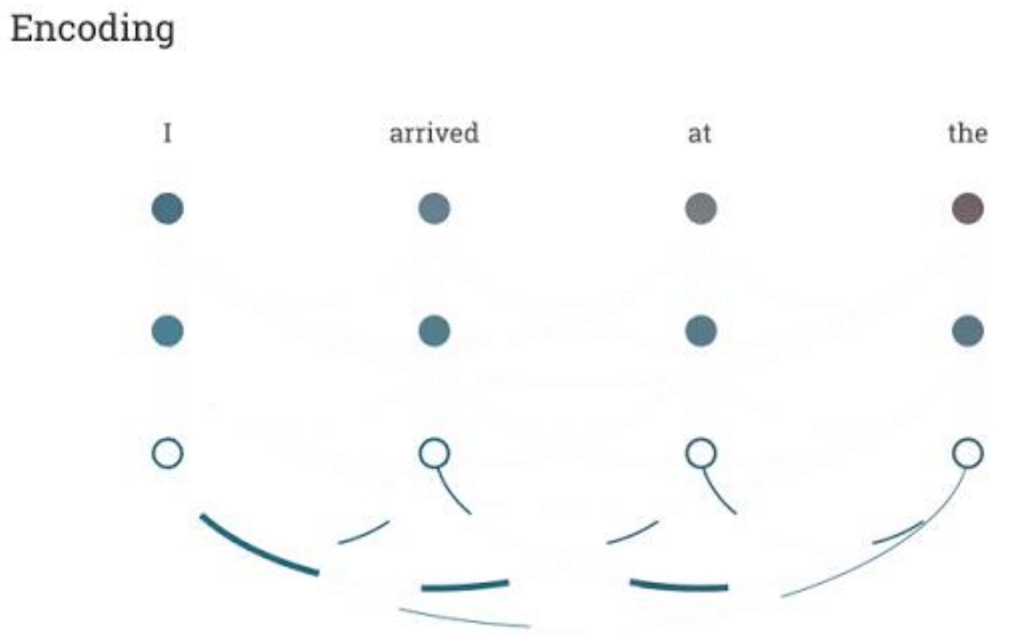
Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

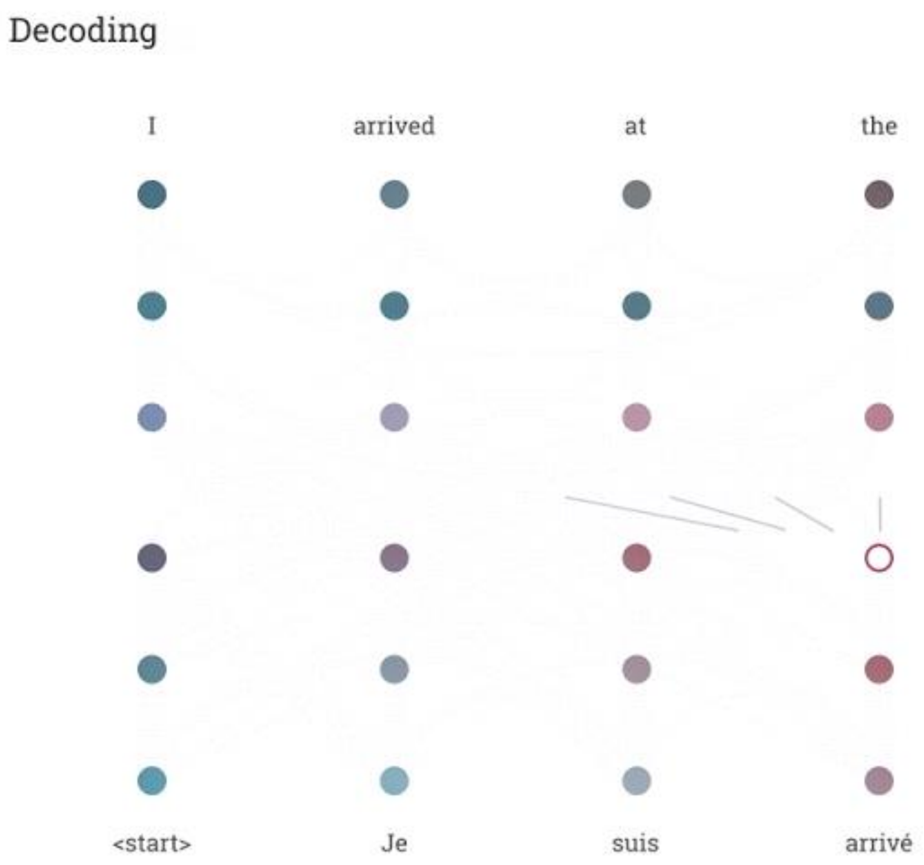
Transformer

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

Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer





We can think that the **relational inductive biases in the transformer** is **different** to the **that of standard RNN encoder-decoder model**.

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

Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

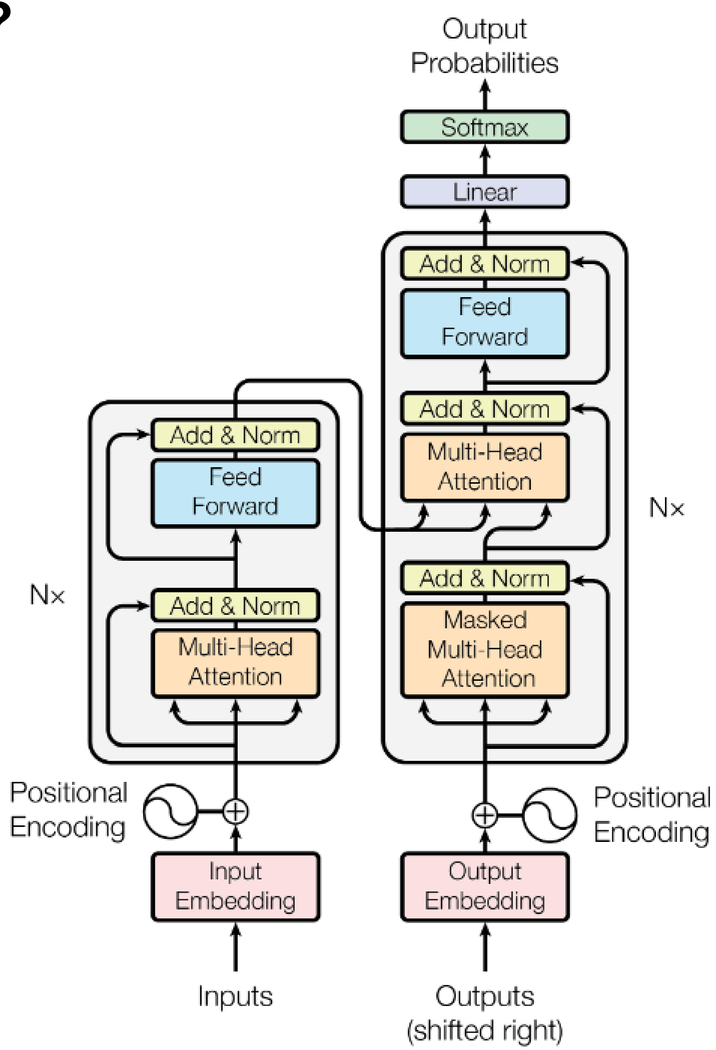
Question) Is not the position of words in the transformer important?

Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

Question) Is not the position of words in the transformer important?

Answer) "Since our model contains no recurrence and no convolution, **in order for the model to make use of the order of the sequence, we must inject some information about the relative or absolute position of the tokens in the sequence**. To this end, we add **"positional embeddings"** to the input embeddings at the bottoms of the encoder and decoder stacks."



Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Transformer

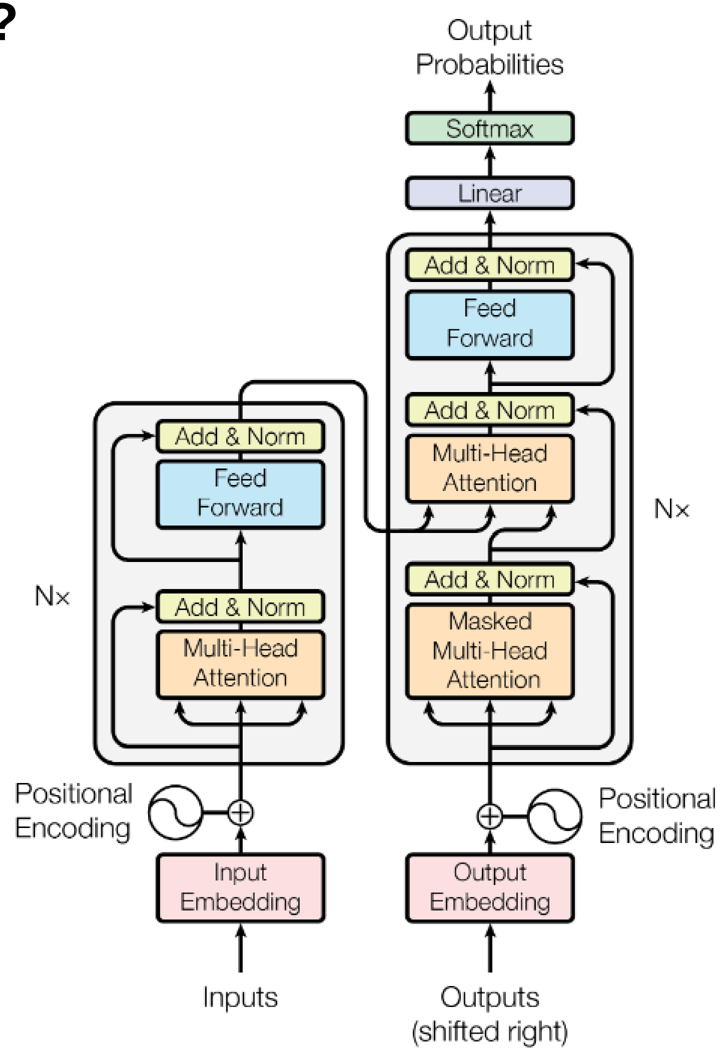
Question) Is not the position of words in the transformer important?

Answer) "Since our model contains no recurrence and no convolution, **in order for the model to make use of the order of the sequence, we must inject some information about the relative or absolute position of the tokens in the sequence**. To this end, we add **"positional embeddings"** to the input embeddings at the bottoms of the encoder and decoder stacks."

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

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

where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to sinusoid. The wavelengths form a geometric progression from 2π to $10000 \cdot 2\pi$.



Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

Self-attention with relative position

Self-attention (transformer)

$$z_i = \sum_{j=1}^{n} \alpha_{ij}\left(x_j W^V\right)$$

$$\alpha_{ij} = \frac{\exp e_{ij}}{\sum_{k=1}^{n} \exp e_{ik}}$$

$$e_{ij} = \frac{(x_i W^Q)\left(x_j W^K\right)^T}{\sqrt{d_z}}$$

Self-attention w/ relative position representation

$$z_i = \sum_{j=1}^{n} \alpha_{ij}\left(x_j W^V + a_{ij}^V\right)$$

$$\alpha_{ij} = \frac{\exp e_{ij}}{\sum_{k=1}^{n} \exp e_{ik}}$$

$$e_{ij} = \frac{(x_i W^Q)\left(x_j W^K + a_{ij}^K\right)^T}{\sqrt{d_z}}$$

Shaw, Peter, Jakob Uszkoreit, and Ashish Vaswani. "Self-Attention with Relative Position Representations." *arXiv preprint arXiv:1803.02155* (2018).

Self-attention with relative position

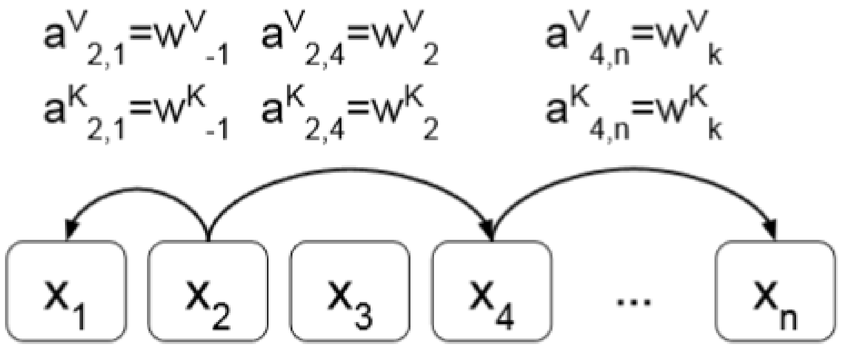


Figure 1: Example edges representing relative positions, or the distance between elements. We learn representations for each relative position within a clipping distance k. The figure assumes $2 <= k <= n - 4$. Note that not all edges are shown.

$$a_{ij}^{K} = w_{\text{clip}(j-i,k)}^{K} \qquad a_{ij}^{V} = w_{\text{clip}(j-i,k)}^{V}$$

$$\text{clip}(x, k) = \max(-k, \min(k, x))$$

Self-attention w/ relative position representation

$$z_i = \sum_{j=1}^{n} \alpha_{ij}\left(x_j W^V + a_{ij}^V\right)$$

$$\alpha_{ij} = \frac{\exp e_{ij}}{\sum_{k=1}^{n} \exp e_{ik}}$$

$$e_{ij} = \frac{(x_i W^Q)\left(x_j W^K + a_{ij}^K\right)^T}{\sqrt{d_z}}$$

Shaw, Peter, Jakob Uszkoreit, and Ashish Vaswani. "Self-Attention with Relative Position Representations." *arXiv preprint arXiv:1803.02155* (2018).

Self-attention with relative position

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**

Table 1: Experimental results for WMT 2014 English-to-German (EN-DE) and English-to-French (EN-FR) translation tasks, using newstest2014 test set.

k	EN-DE BLEU
0	12.5
1	25.5
2	25.8
4	25.9
16	25.8
64	25.9
256	25.8

Table 2: Experimental results for varying the clipping distance, k.

a_{ij}^V	a_{ij}^K	EN-DE BLEU
Yes	Yes	25.8
No	Yes	25.8
Yes	No	25.3
No	No	12.5

Table 3: Experimental results for ablating relative position representations a_{ij}^V and a_{ij}^K.

Shaw, Peter, Jakob Uszkoreit, and Ashish Vaswani. "Self-Attention with Relative Position Representations." *arXiv preprint arXiv:1803.02155* (2018).

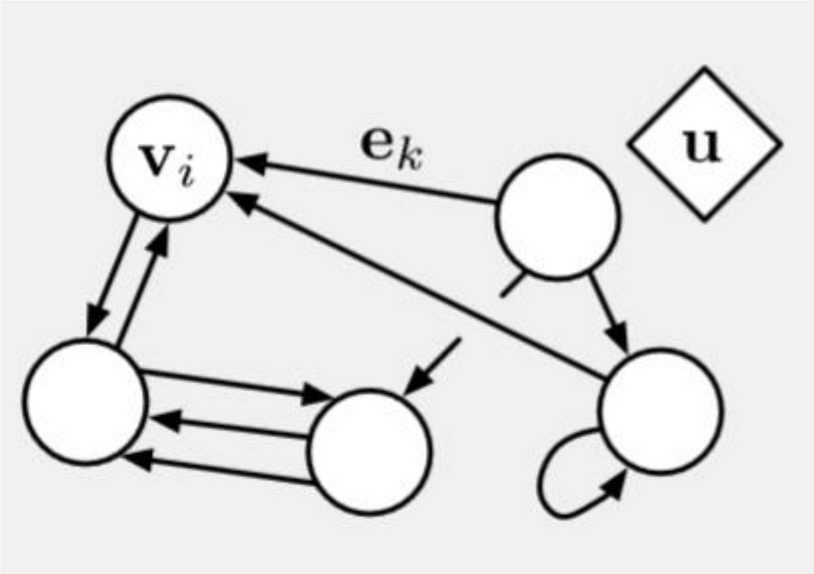
Summary

- ✓ Attention mechanisms have been used **to select information to be read out from the internal memory of the network**.
- ✓ In RNN encoder-decoder models, the attention mechanism is used to spotlight on more important words for predicting output words better.
- ✓ The Transformer does not employ any recurrent or convolution operations, but only using MLPs with the self-attentions. In other words, there are **weak inductive biases** in the Transformer.
- ✓ In order for the model to make use of the order of the sequence, the Transformer must inject some information about the relative or absolute position of the tokens in the sequence.
- ✓ The relative position representation is used and using it outperforms the Transformer which employs the position embedding at the bottom of encoder and decoder.
- ✓ In summary, **attention mechanisms find the relation between entities effectively**.

Relational inference

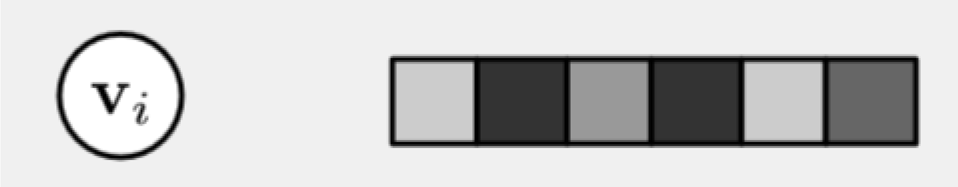
Relational inference

Recall that



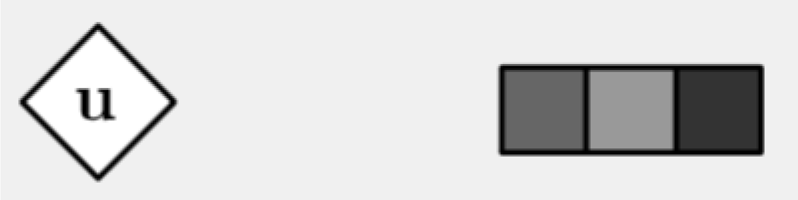
- ✓ Directed : one-way edges, from a “sender” node to a “receiver” node.
- ✓ Attribute : properties that can be encoded as a vector, set, or even another graph
- ✓ Attributed : edges and vertices have attributes associated with them

Node’s attribute



Edge’s attribute

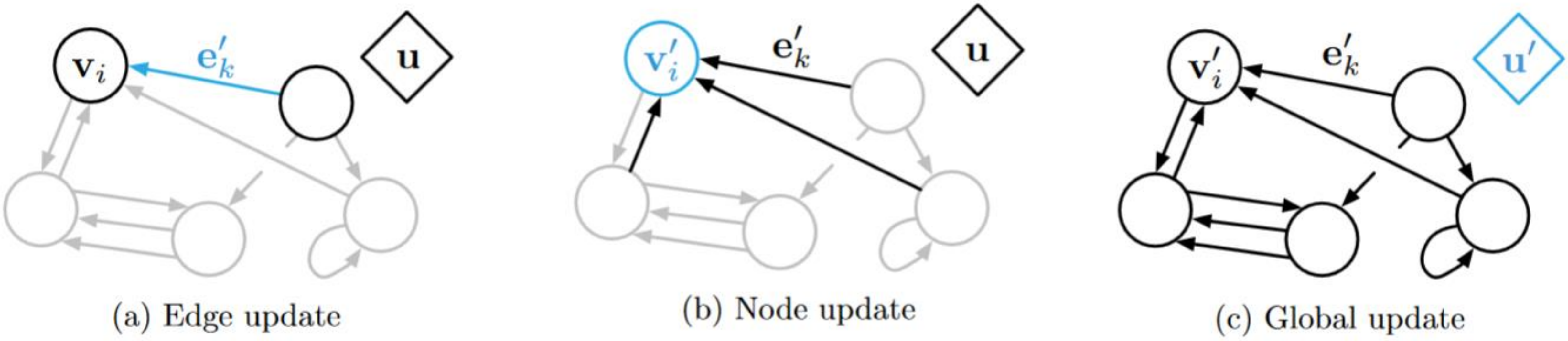
Global attribute



Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Relational inference

Recall that



$$\mathbf{e}'_k = \mathrm{NN}\left(\mathbf{v}_{s_k}, \mathbf{v}_{r_k}, \mathbf{e}_k, \mathbf{u}\right)$$

$$\bar{\mathbf{e}}'_i = \sum_{k:r_k=i} \mathbf{e}'_k$$

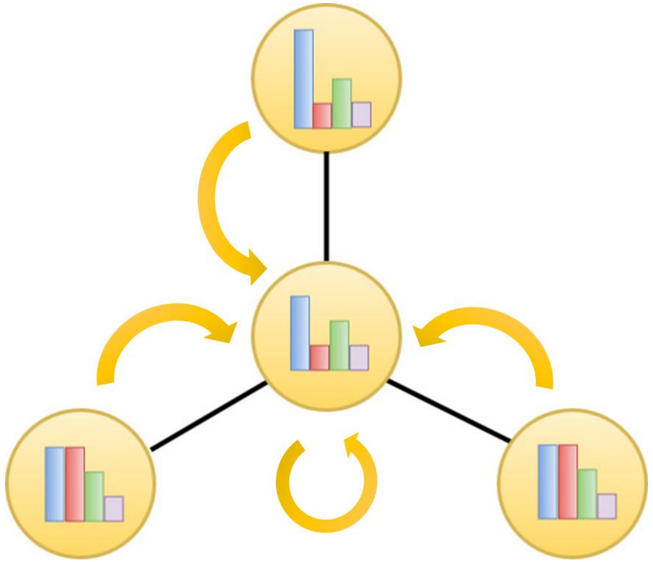
$$\mathbf{v}'_i = \mathrm{NN}(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u})$$

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

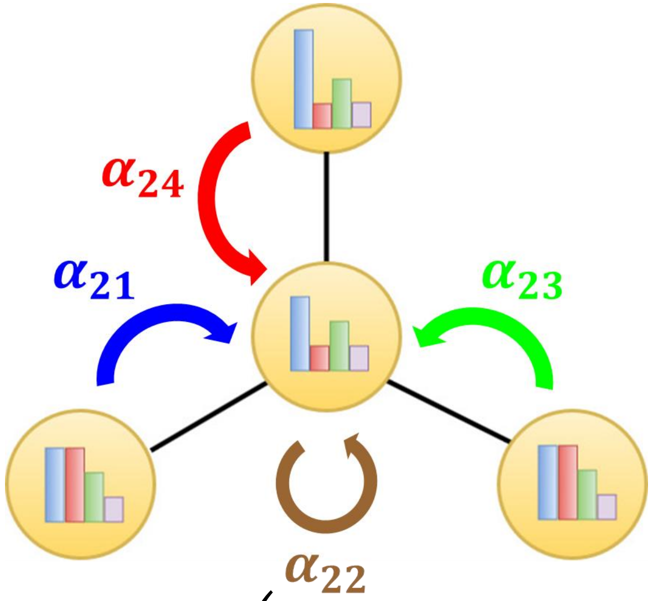
Relational inference

Recall that

The **attention** is nothing but **edge attribute** which find a relation between node attributes



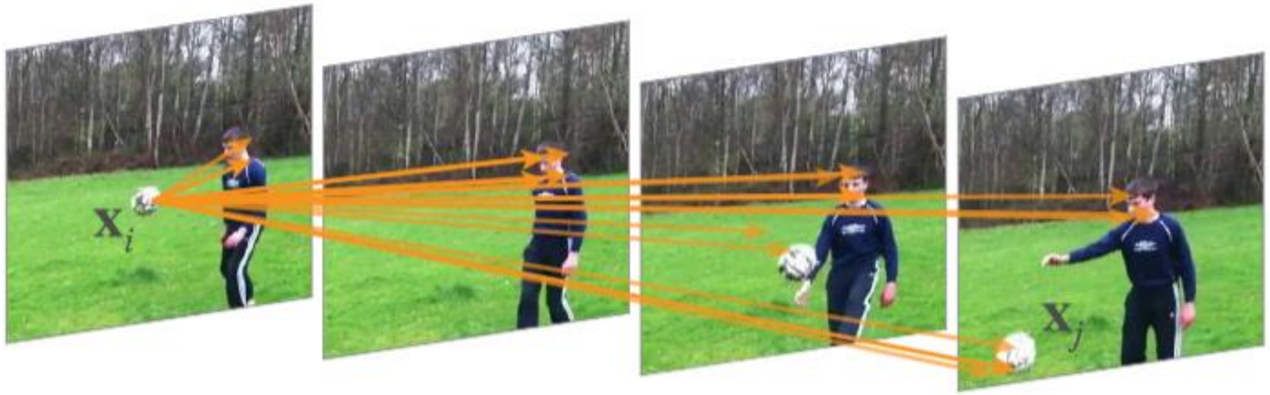
$$H^{(l+1)} = \sigma\left(\sum_{j \in N(i)} H_j^{(l)} W^{(l)}\right)$$



$$H^{(l+1)} = \sigma\left(\sum_{j \in N(i)} \alpha_{ij} H_j^{(l)} W^{(l)}\right)$$

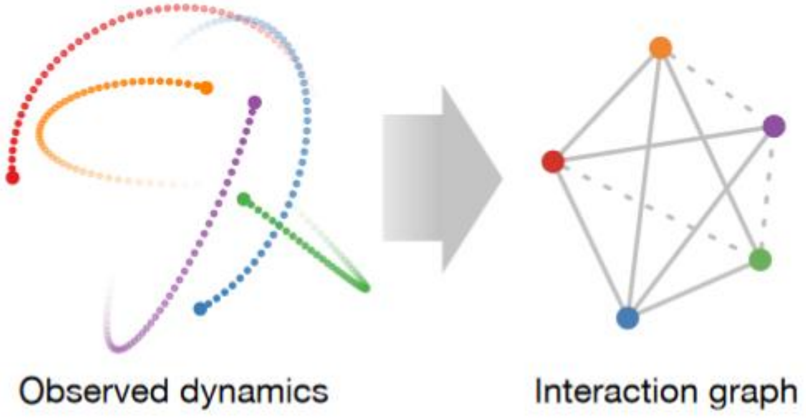
Relational inference

Literatures

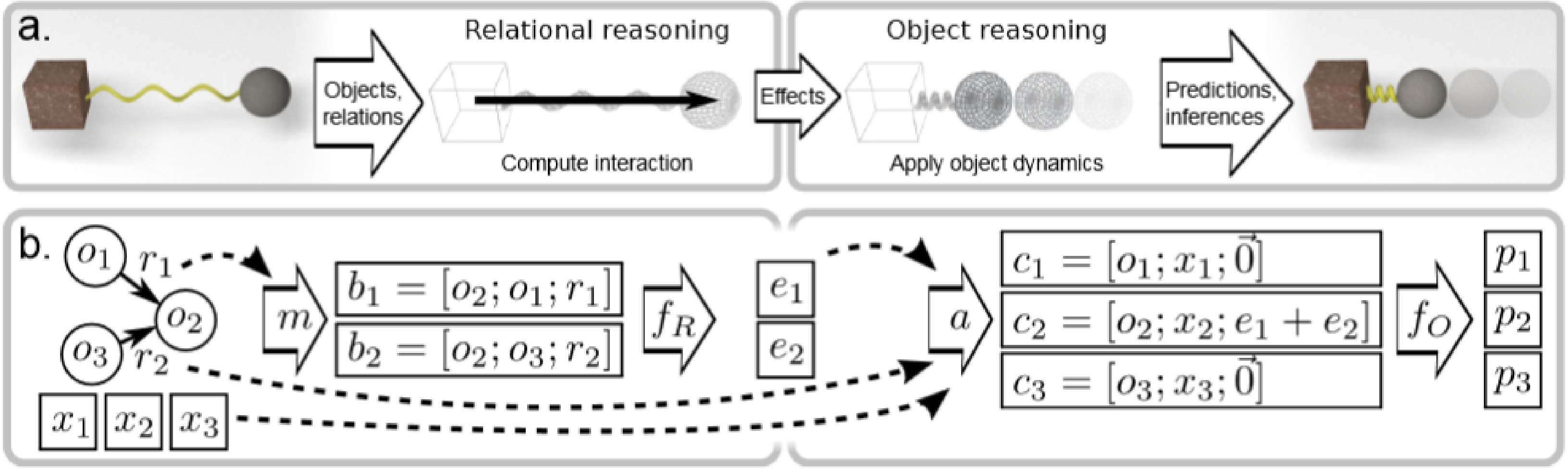


Vision

Wang, Xiaolong, et al. "Non-local neural networks." *arXiv preprint arXiv:1711.07971* 10 (2017).



Kipf, Thomas, et al. "Neural relational inference for interacting systems." *arXiv preprint arXiv:1802.04687* (2018).



Physics modeling

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

Non-local neural network

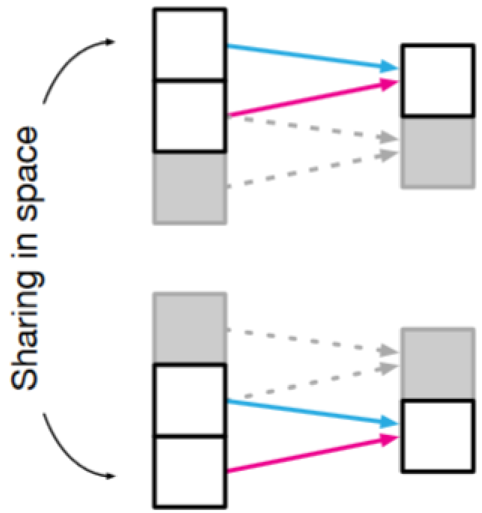
Limitations of CNN

In order to see wide regions

- CNN ought to be deep
- Receptive field must be wide
- Using pooling operations for dimensionality reduction

⟶ Requires high computational costs.

In terms of inductive biases, common CNN detects non-local regime by the local operation.



Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation
Graph network	Nodes	Edges	Arbitrary	Node, edge permutations

Battaglia, Peter W., et al. "Relational inductive biases, deep learning, and graph networks." *arXiv preprint arXiv:1806.01261* (2018).

Non-local neural network

Non-local mean operation

Normalization factor

Representation of the input signal at position j

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

Relationship between i and j

- ✓ $\mathbf{x}$: input signal
- ✓ $\mathbf{y}$: output signal
- ✓ i : the index of an input(output) position

Wang, Xiaolong, et al. "Non-local neural networks." *arXiv preprint arXiv:1711.07971* 10 (2017).

Non-local neural network

Non-local mean operation

Normalization factor

Representation of the input signal at position j

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

Relationship between i and j

$$i - 1 \leq j \leq i + 1$$

- ✓ $\mathbf{x}$: input signal
- ✓ $\mathbf{y}$: output signal
- ✓ i : the index of an input(output) position

Convolution with kernel size 3

INPUT IMAGE

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

WEIGHT

1	0	1
0	1	0
1	0	1

429

Wang, Xiaolong, et al. "Non-local neural networks." *arXiv preprint arXiv:1711.07971* 10 (2017).

Non-local neural network

Non-local mean operation

Normalization factor

Representation of the input signal at position j

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

Relationship between i and j

- ✓ $\mathbf{x}$: input signal
- ✓ $\mathbf{y}$: output signal
- ✓ i : the index of an input(output) position

$i-1 \leq j \leq i+1$

Convolution with kernel size 3

INPUT IMAGE

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

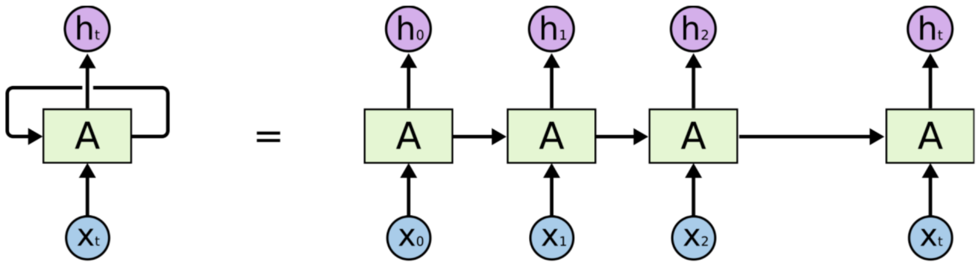
WEIGHT

1	0	1
0	1	0
1	0	1

429

$j = i$ or $j = i-1$

Recurrence



Wang, Xiaolong, et al. "Non-local neural networks." *arXiv preprint arXiv:1711.07971* 10 (2017).

Non-local neural network

Non-local mean operation

Normalization factor

Representation of the input signal at position j

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

Relationship between i and j

- ✓ $\mathbf{x}$: input signal
- ✓ $\mathbf{y}$: output signal
- ✓ i : the index of an input(output) position

Nonlocal behavior is due to the fact that **all positions ($\forall j$) are considered in the operations**
→ Weak inductive bias

$i-1 \le j \le i+1$

Convolution with kernel size 3

INPUT IMAGE

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

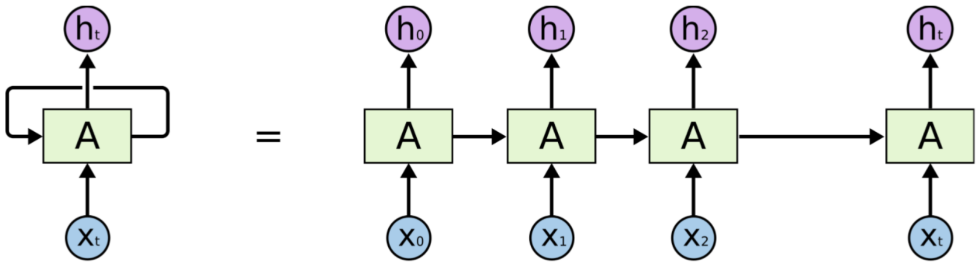
WEIGHT

1	0	1
0	1	0
1	0	1

429

$j = i$ or $j = i - 1$

Recurrence



Wang, Xiaolong, et al. "Non-local neural networks." *arXiv preprint arXiv:1711.07971* 10 (2017).

Non-local neural network

Instantiation

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

They used for $g(\mathbf{x}_j) = \mathbf{W}_g \mathbf{x}_j$,
which corresponds to 1×1 convolution.

Wang, Xiaolong, et al. "Non-local neural networks." *arXiv preprint arXiv:1711.07971* 10 (2017).

Non-local neural network

Instantiation

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

They used for $g(\mathbf{x}_j) = \mathbf{W}_g \mathbf{x}_j$,
which corresponds to 1×1 convolution.

Gaussian $f(\mathbf{x}_i, \mathbf{x}_j) = e^{\mathbf{x}_i^T \mathbf{x}_j}$

Wang, Xiaolong, et al. "Non-local neural networks."
arXiv preprint arXiv:1711.07971 10 (2017).

Non-local neural network

Instantiation

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

They used for $g(\mathbf{x}_j) = \mathbf{W}_g \mathbf{x}_j$,
which corresponds to 1×1 convolution.

Gaussian $f(\mathbf{x}_i, \mathbf{x}_j) = e^{\mathbf{x}_i^T \mathbf{x}_j}$

Embedded Gaussian $f(\mathbf{x}_i, \mathbf{x}_j) = e^{\theta(\mathbf{x}_i)^T \phi(\mathbf{x}_j)}$

Wang, Xiaolong, et al. "Non-local neural networks."
arXiv preprint arXiv:1711.07971 10 (2017).

Non-local neural network

Instantiation

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

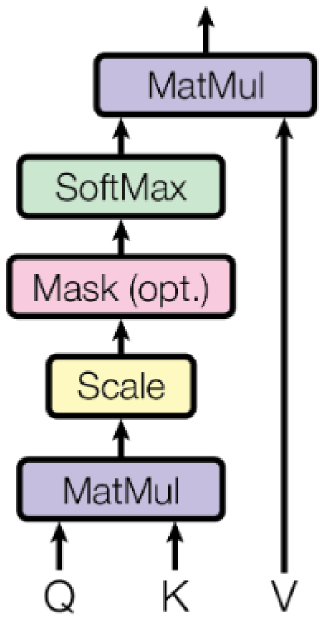
They used for $g(\mathbf{x}_j) = \mathbf{W}_g \mathbf{x}_j$,
which corresponds to 1×1 convolution.

Gaussian $\quad f(\mathbf{x}_i, \mathbf{x}_j) = e^{\mathbf{x}_i^T \mathbf{x}_j}$

Embedded Gaussian $\quad f(\mathbf{x}_i, \mathbf{x}_j) = e^{\theta(\mathbf{x}_i)^T \phi(\mathbf{x}_j)}$

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}$$

Scaled Dot-Product Attention



Self-attention in the Transformer is special case of non-local operations in the embedded Gaussian.

$$\mathbf{y} = \text{softmax}\left((\mathbf{W}_\theta \mathbf{x})^T (\mathbf{W}_\phi \mathbf{x})\right) g(\mathbf{x})$$

Wang, Xiaolong, et al. "Non-local neural networks." *arXiv preprint arXiv:1711.07971* 10 (2017).

Non-local neural network

Instantiation

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

They used for $g(\mathbf{x}_j) = \mathbf{W}_g \mathbf{x}_j$, which corresponds to 1×1 convolution.

Gaussian $f(\mathbf{x}_i, \mathbf{x}_j) = e^{\mathbf{x}_i^T \mathbf{x}_j}$

Embedded Gaussian $f(\mathbf{x}_i, \mathbf{x}_j) = e^{\theta(\mathbf{x}_i)^T \phi(\mathbf{x}_j)}$

Dot product $f(\mathbf{x}_i, \mathbf{x}_j) = \theta(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$

Concatenation $f(\mathbf{x}_i, \mathbf{x}_j) = \mathrm{ReLU}\left(\mathbf{w}_f^T [\theta(\mathbf{x}_i), \phi(\mathbf{x}_j)]\right)$

Wang, Xiaolong, et al. "Non-local neural networks." *arXiv preprint arXiv:1711.07971* 10 (2017).

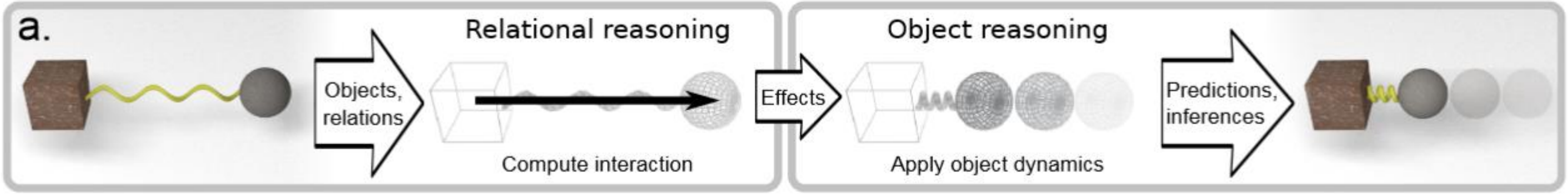
Non-local neural network

Figure 3. Examples of the behavior of a non-local block in res_3 computed by a 5-block non-local model trained on Kinetics. These examples are from held-out validation videos. The starting point of arrows represents one $\mathbf{x}_i$, and the ending points represent $\mathbf{x}_j$. The 20 highest weighted arrows for each $\mathbf{x}_i$ are visualized. The 4 frames are from a 32-frame input, shown with a stride of 8 frames. These visualizations show how the model finds related clues to support its prediction.

→ The 20 highest weighted arrows for each $\mathbf{x}_i$

Wang, Xiaolong, et al. "Non-local neural networks." *arXiv preprint arXiv:1711.07971* 10 (2017).

Interaction network

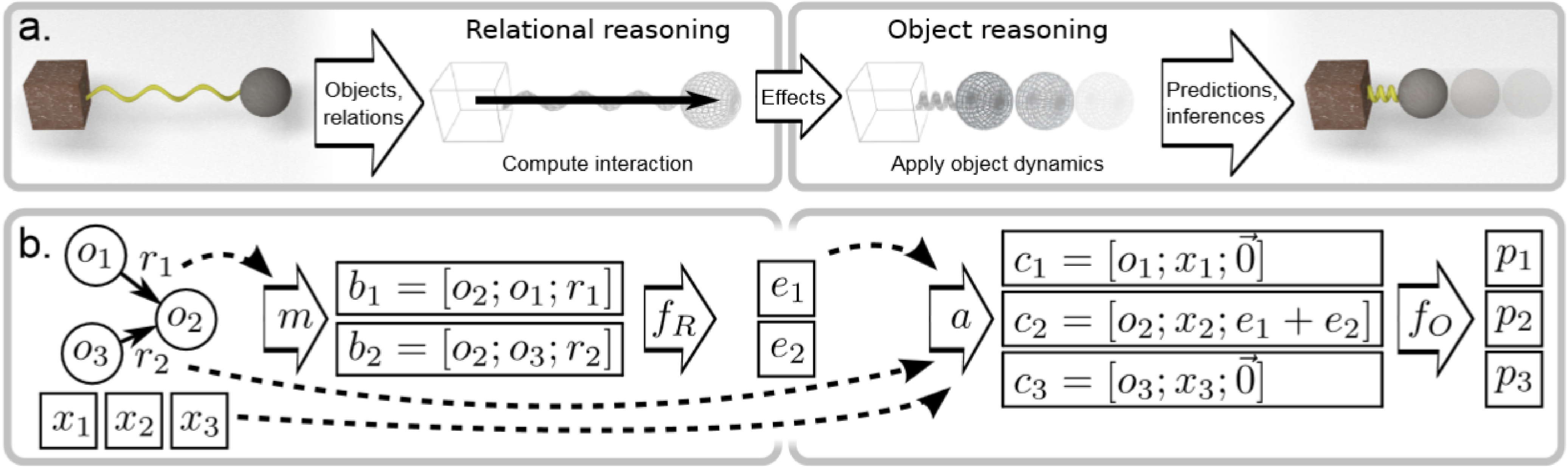


For physical reasoning

1) The model takes objects and relations as input
2) reasons about their interactions
3) applies the effects and physical dynamics to predict new states

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

Interaction network



For more complex systems

1) The model takes an input a graph that represents a system of objects o_j and relations $\langle i, j, r_k \rangle_k$
2) instantiates the pairwise interaction terms b_k
3) and computes their effects e_k via a relational model $f_R(\cdot)$.
4) The e_k are then aggregated and combined with o_j and external effects x_j to generate input (as c_j) for an object model
5) f_O predicts how the interactions and dynamics influence the objects p.

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

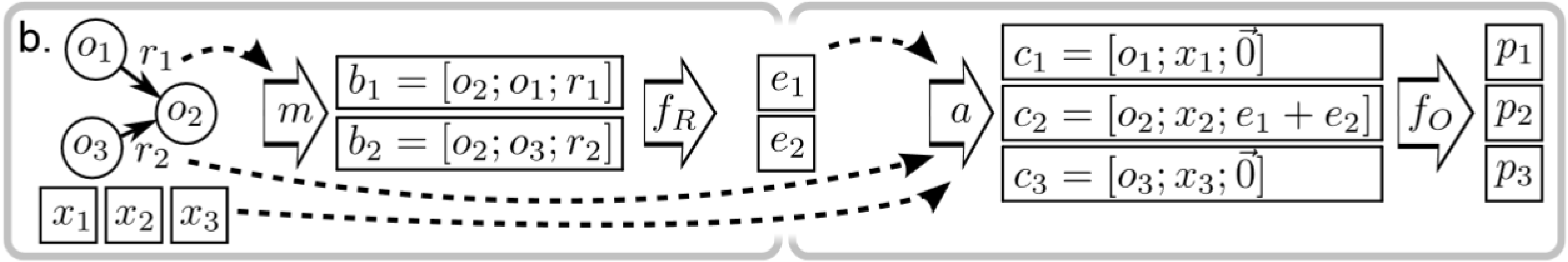
Interaction network

The input to the interaction network (IN) are

$$O = \{o_j\}_{j=1,\ldots,N_O}, \qquad R = \{\langle i, j, r_k\rangle_k\}_{k=1,\ldots,N_R} \text{ where } i \neq j, 1 \leq i,j \leq N_O, \qquad X = \{x_j\}_{j=1,\ldots,N_O}$$

Basic IN is defined as

$$\mathrm{IN}(G) = \phi_O\Big(a\big(G, X, \phi_R(m(G))\big)\Big) \;\; \mathit{where}\ G = \langle O, R\rangle$$



Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

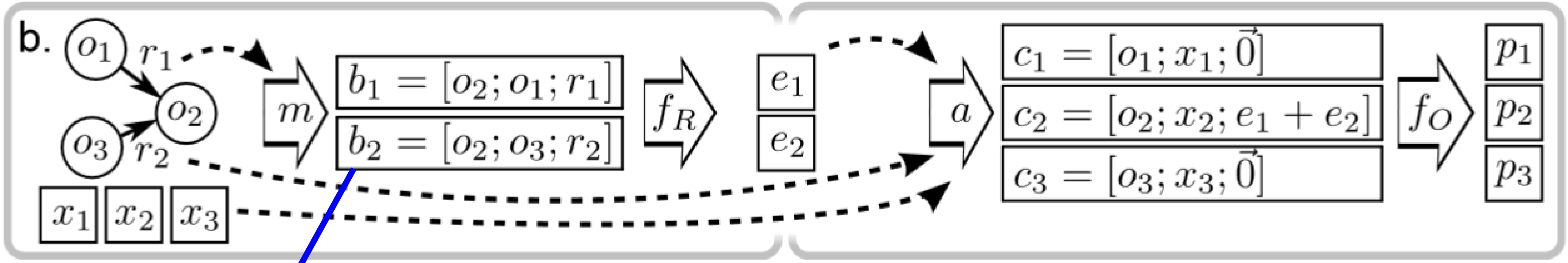
Interaction network

The input to the interaction network (IN) are

$$O = \{o_j\}_{j=1,\dots,N_O}, \qquad R = \{\langle i,j,r_k\rangle_k\}_{k=1,\dots,N_R} \text{ where } i \neq j, 1 \leq i,j \leq N_O, \qquad X = \{x_j\}_{j=1,\dots,N_O}$$

Basic IN is defined as

$$\mathrm{IN}(G) = \phi_O\left(a\left(G, X, \phi_R(m(G))\right)\right) \; where\ G = \langle O, R\rangle$$



$B = \{b_k\}_{k=1,\dots,N_R} = m(G)$

Rearranges the objects and relations into interaction terms $b_k \in B$

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

Interaction network

The input to the interaction network (IN) are

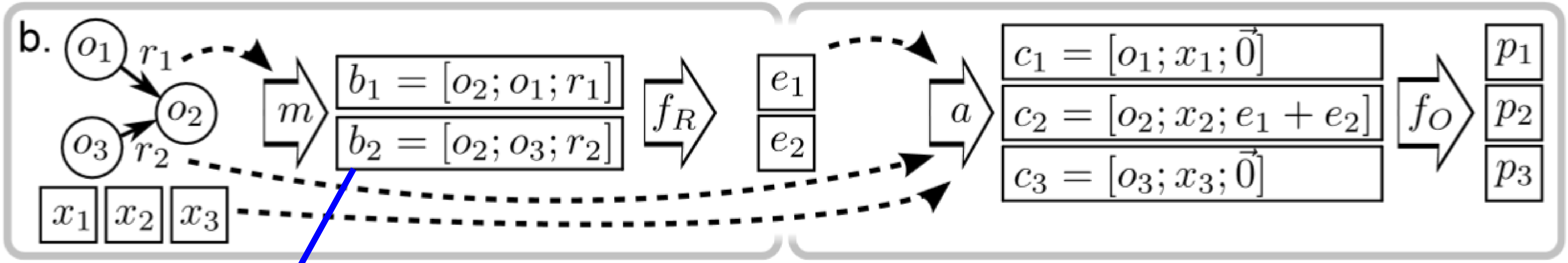
$$O = \{o_j\}_{j=1,\ldots,N_O}, \qquad R = \{\langle i,j,r_k\rangle_k\}_{k=1,\ldots,N_R} \text{ where } i \neq j, 1 \leq i,j \leq N_O, \qquad X = \{x_j\}_{j=1,\ldots,N_O}$$

Basic IN is defined as

$$\mathrm{IN}(G) = \phi_O\Big(a\big(G, X, \phi_R(m(G))\big)\Big) \;\; where\ G = \langle O, R\rangle$$

$E = \{e_k\}_{k=1,\ldots,N_R} = \{f_R(b_k)\}_{k=1,\ldots,N_R}$

Predicts the effect of each interaction $e_k \in E$



$B = \{b_k\}_{k=1,\ldots,N_R} = m(G)$

Rearranges the objects and relations into interaction terms $b_k \in B$

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

Interaction network

The input to the interaction network (IN) are

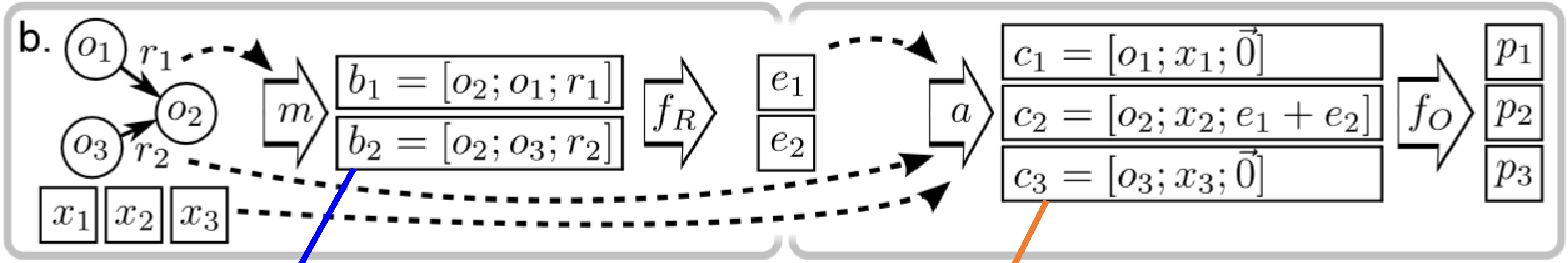
$$O = \{o_j\}_{j=1,\ldots,N_O}, \qquad R = \{\langle i, j, r_k \rangle_k\}_{k=1,\ldots,N_R} \text{ where } i \neq j, 1 \leq i, j \leq N_O, \qquad X = \{x_j\}_{j=1,\ldots,N_O}$$

Basic IN is defined as

$$\mathrm{IN}(G) = \phi_O\Big(a\big(G, X, \phi_R(m(G))\big)\Big) \ \ \textit{where } G = \langle O, R \rangle$$

$$E = \{e_k\}_{k=1,\ldots,N_R} = \{f_R(b_k)\}_{k=1,\ldots,N_R}$$

Predicts the effect of each interaction $e_k \in E$



$$B = \{b_k\}_{k=1,\ldots,N_R} = m(G)$$

Rearranges the objects and relations into interaction terms $b_k \in B$

$$C = \{c_j\}_{j=1,\ldots,N_O} = a(G, X, E)$$

The aggregation function combines E with O and X to form a set of object model inputs $c_j \in C$

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

Interaction network

The input to the interaction network (IN) are

$$O = \{o_j\}_{j=1,\dots,N_O}, \qquad R = \{\langle i,j,r_k\rangle_k\}_{k=1,\dots,N_R} \text{ where } i \neq j, 1 \leq i,j \leq N_O, \qquad X = \{x_j\}_{j=1,\dots,N_O}$$

Basic IN is defined as

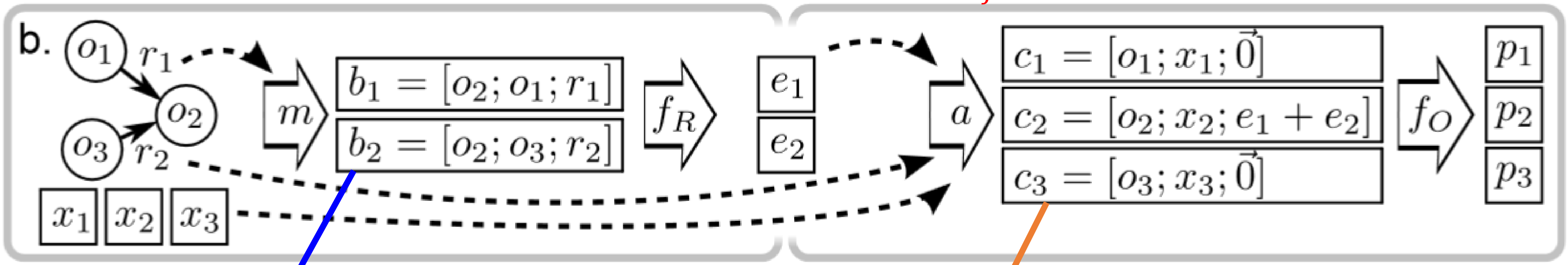
$$\mathrm{IN}(G) = \phi_O\Big(a\big(G, X, \phi_R(m(G))\big)\Big) \; where\; G = \langle O, R\rangle$$

$$E = \{e_k\}_{k=1,\dots,N_R} = \{f_R(b_k)\}_{k=1,\dots,N_R}$$

Predicts the effect of each interaction $e_k \in E$

$$P = \{p_j\}_{j=1,\dots,N_O} = \{f_O(c_j)\}_{j=1,\dots,N_O}$$

The object model predicts the how the interactions and dynamics influence the objects and returning the results $p_j \in P$



$$B = \{b_k\}_{k=1,\dots,N_R} = m(G)$$

Rearranges the objects and relations into interaction terms $b_k \in B$

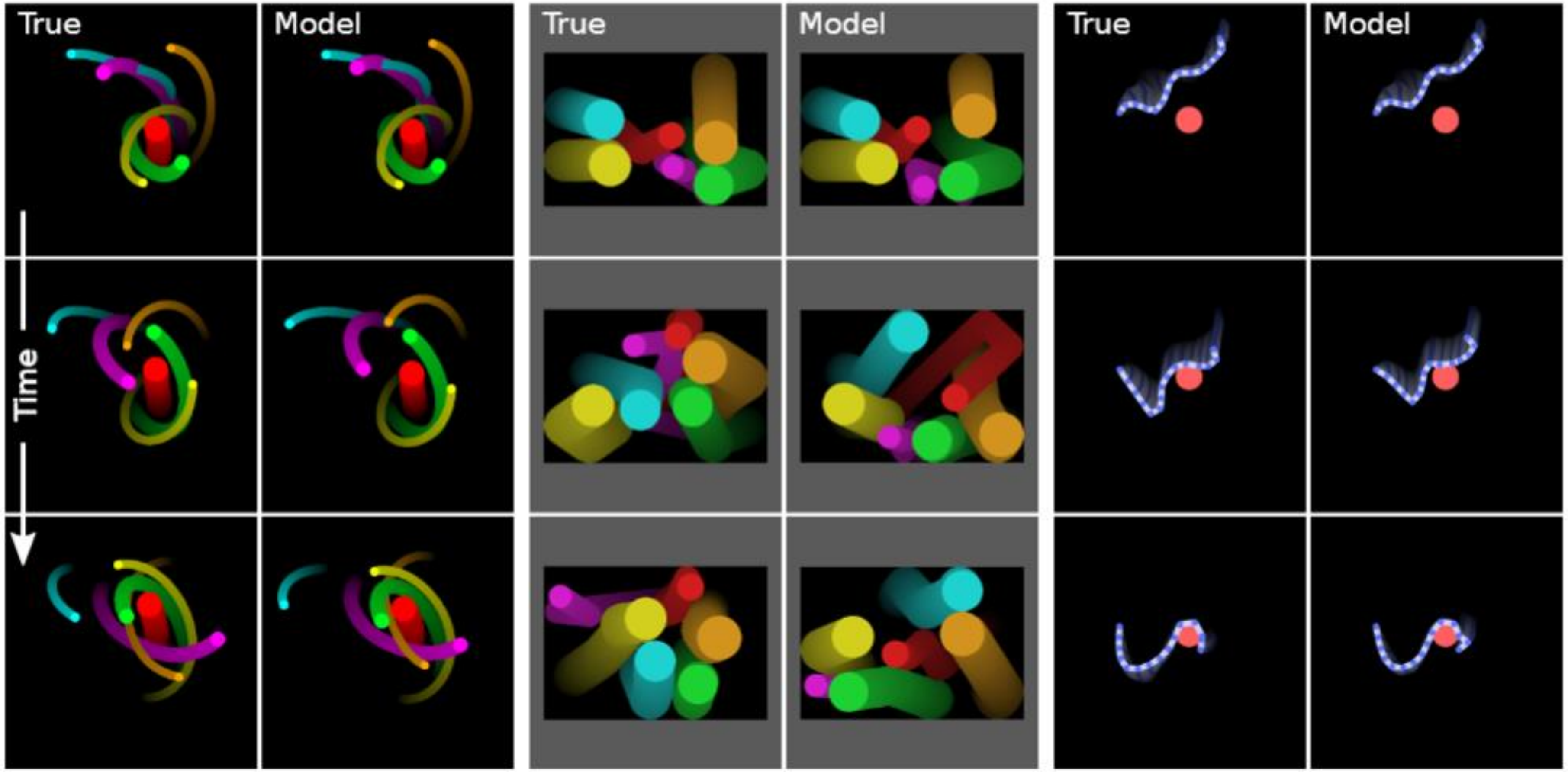
$$C = \{c_j\}_{j=1,\dots,N_O} = a(G, X, E)$$

The aggregation function combines E with O and X to form a set of object model inputs $c_j \in C$

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

Interaction network

Results



N-body system

$$F_{ij} = \frac{Gm_i m_j (x_i - x_j)}{\|x_i - x_j\|^2}$$

Bouncing balls

String

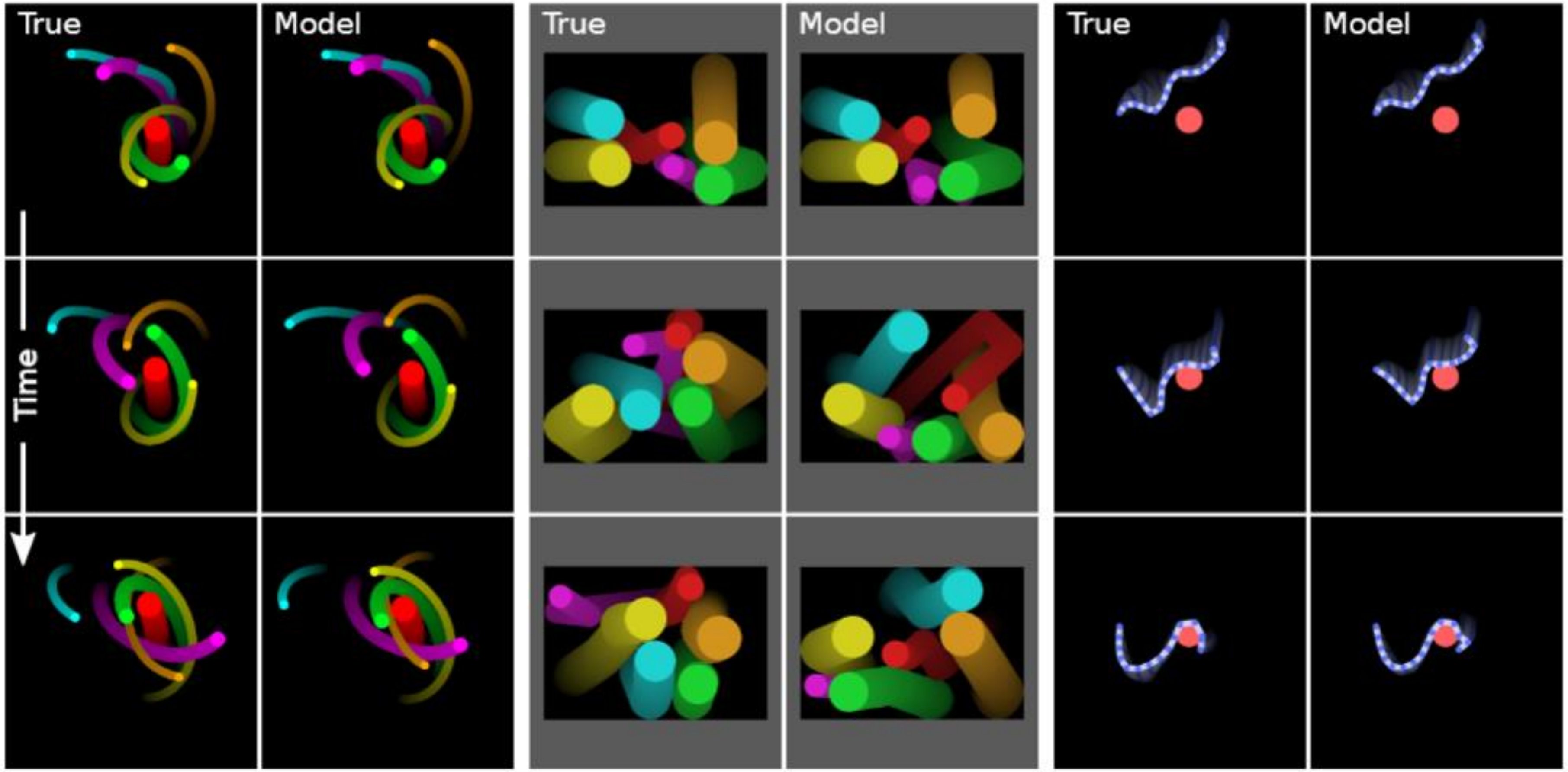
$$F_{ij} = C_s \left(1 - \frac{L}{\|x_i - x_j\|^2}\right)(x_i - x_j)$$

Appendix

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

Interaction network

Results

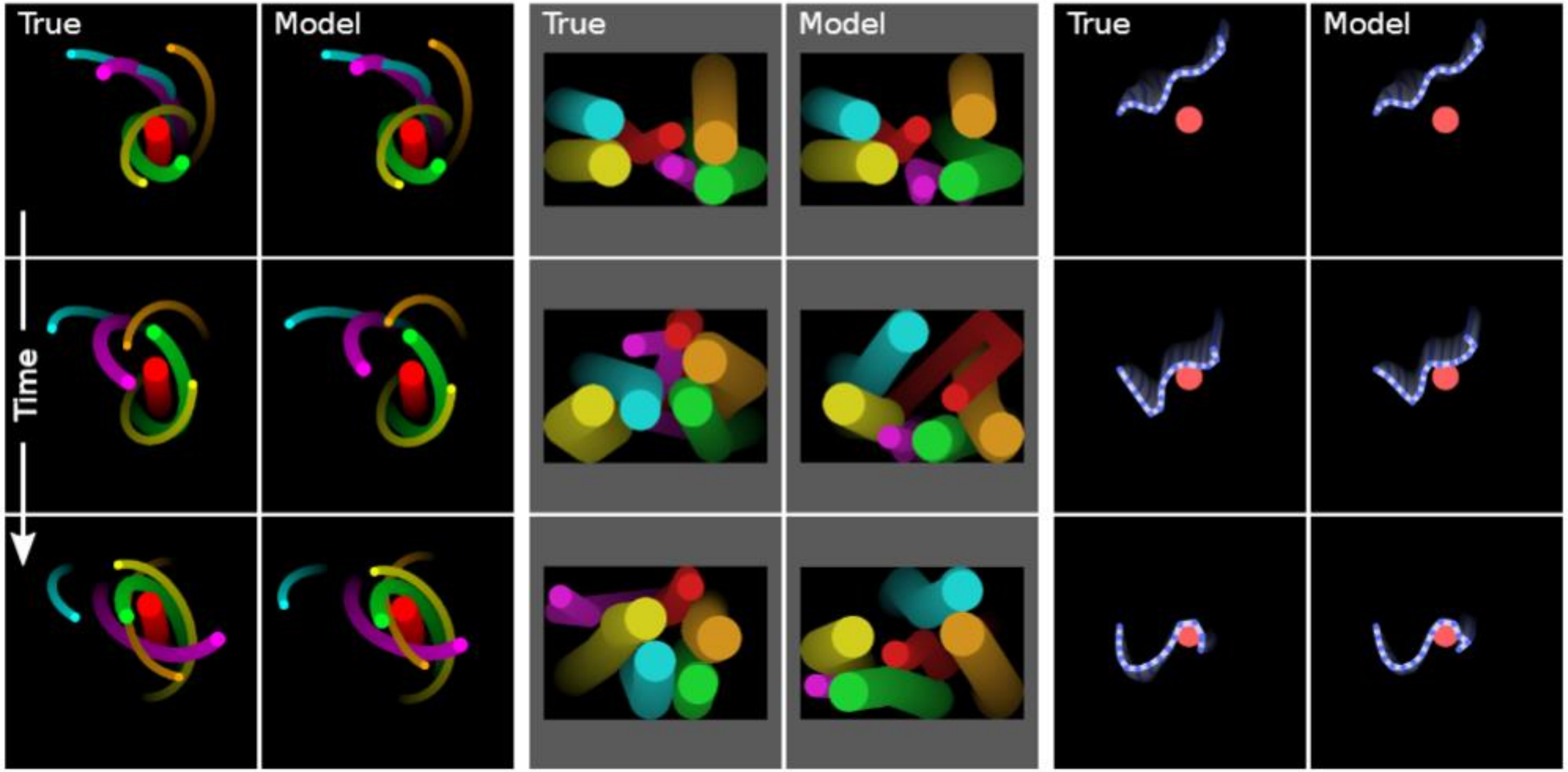


Limitations : relation between objects R must be given.
Question) Cannot we infer the relation between objects from a neural network?

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

Interaction network

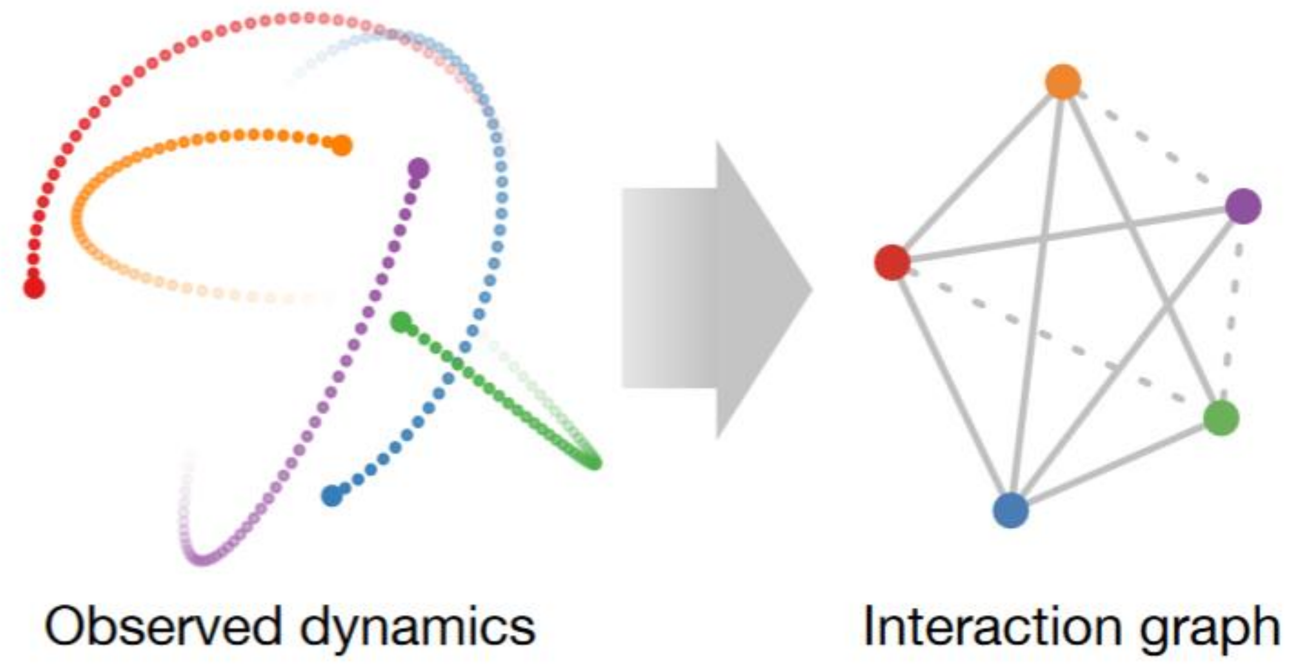
Results



Limitations : relation between objects R must be given.
Question) Cannot we infer the relation between objects from a neural network?
Answer) “Neural Relational Inference (NRI)”

Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.

Neural relational inference



Interactions between particles (entities) can be represented by the interaction graph.

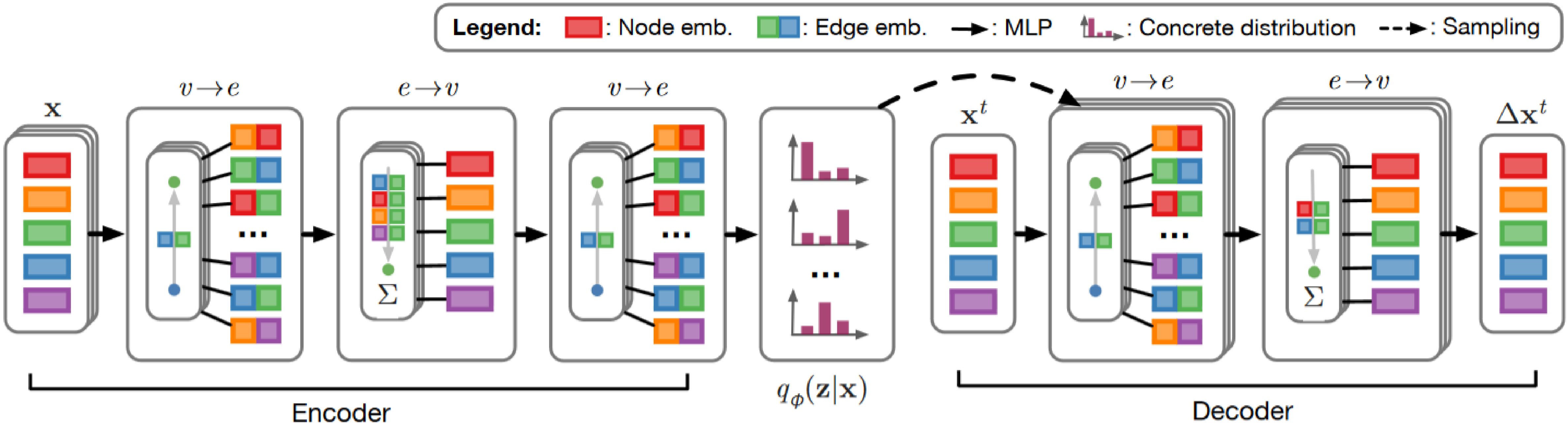
- ✓ Nodes - particles (entities)
- ✓ Edges - interactions (relations)

In this work, the interactions which corresponds to the edge states are inferred from physical dynamics data, so-called neural relational inference (NRI).

Kipf, Thomas, et al. "Neural relational inference for interacting systems." *arXiv preprint arXiv:1802.04687* (2018).

Neural relational inference

Overall procedure



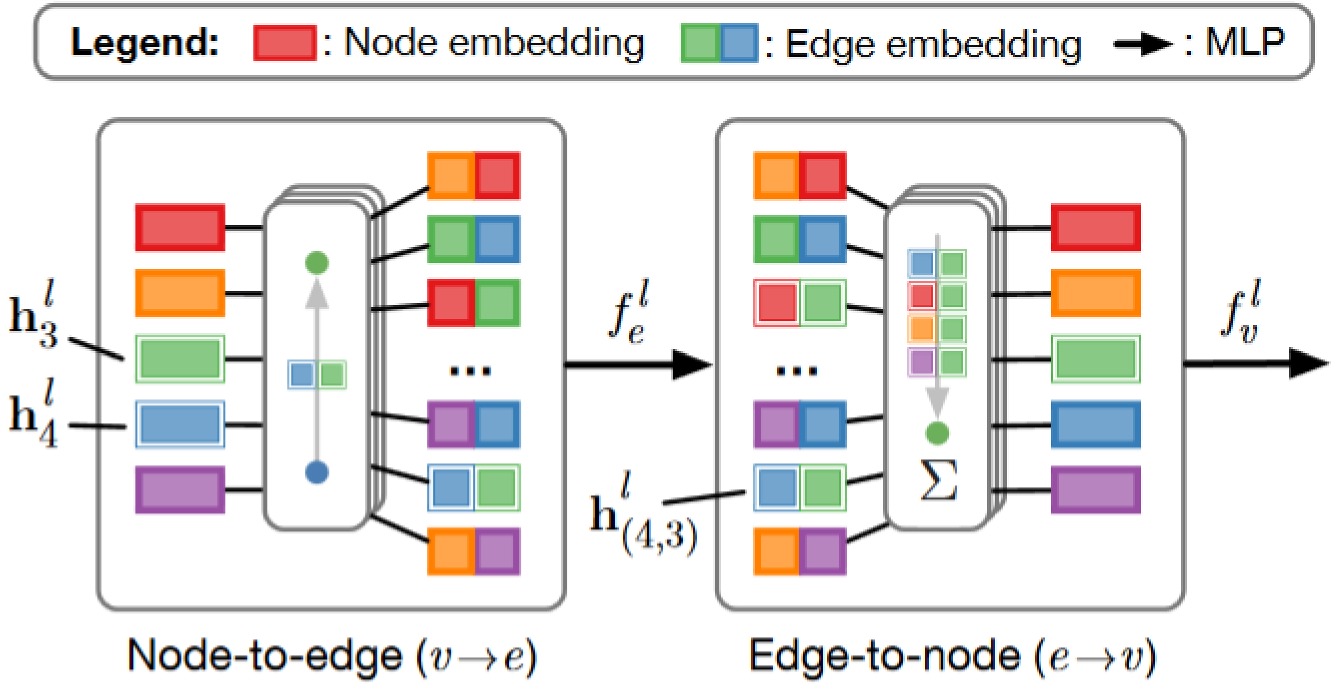
✓ The encoder updates edge states and embeds the relations as latent distributions.

✓ The decoder updates future particles state (changes) using the latent relation distributions.

Kipf, Thomas, et al. "Neural relational inference for interacting systems." *arXiv preprint arXiv:1802.04687* (2018).

Neural relational inference

Basic building blocks of NRI



Node-to-edge

$$v \to e : \mathbf{h}_{(i,j)}^{(l)} = f_e^{(l)}\left(\left[\mathbf{h}_i^{(l)}, \mathbf{h}_j^{(l)}, \mathbf{x}_{(i,j)}\right]\right)$$

Edge-to-noge (e $\to v$)

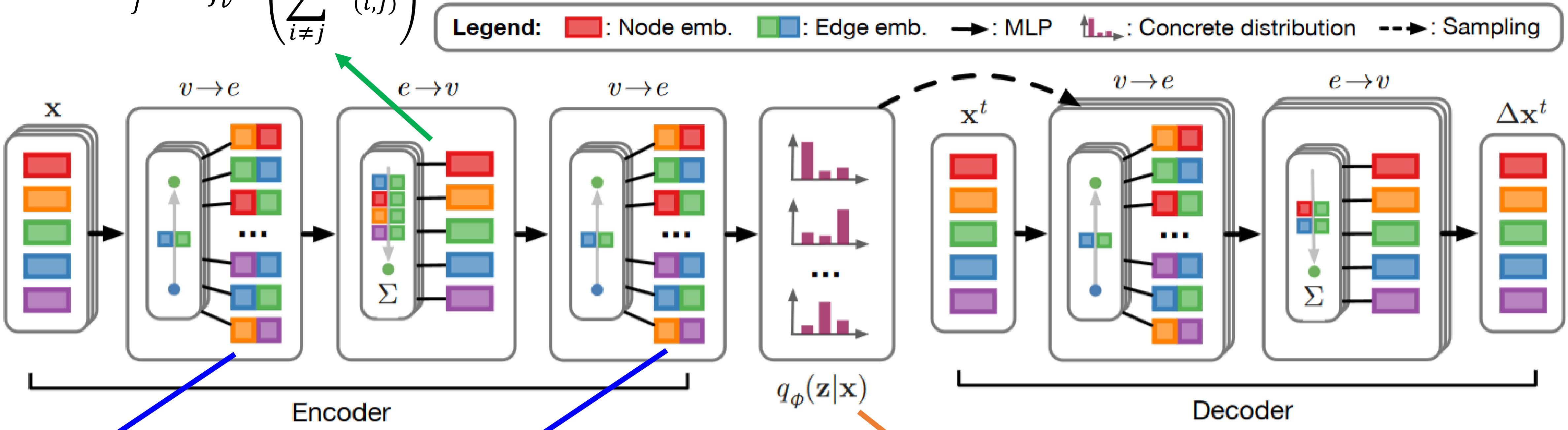
$$e \to v : \mathbf{h}_j^{(l+1)} = f_v^{(l)}\left(\left[\sum_{i \in \mathcal{N}_j} \mathbf{h}_{(i,j)}^{(l)}, \mathbf{x}_j\right]\right)$$

Kipf, Thomas, et al. "Neural relational inference for interacting systems." *arXiv preprint arXiv:1802.04687* (2018).

Neural relational inference

Encoder

$$\mathbf{h}_j^{(2)} = f_v^{(1)}\left(\sum_{i \neq j} \mathbf{h}_{(i,j)}^{(1)}\right)$$



$$\mathbf{h}_{(i,j)}^{(1)} = f_e^{(1)}\left(\left[\mathbf{h}_i^{(1)}, \mathbf{h}_j^{(1)}\right]\right)$$

$$\mathbf{h}_{(i,j)}^{(2)} = f_e^{(2)}\left(\left[\mathbf{h}_i^{(2)}, \mathbf{h}_j^{(2)}\right]\right)$$

$$\mathbf{z}_{ij} = \mathrm{softmax}\left(\frac{\left(\mathbf{h}_{(i,j)}^{(2)} + \mathbf{g}\right)}{\tau}\right)$$

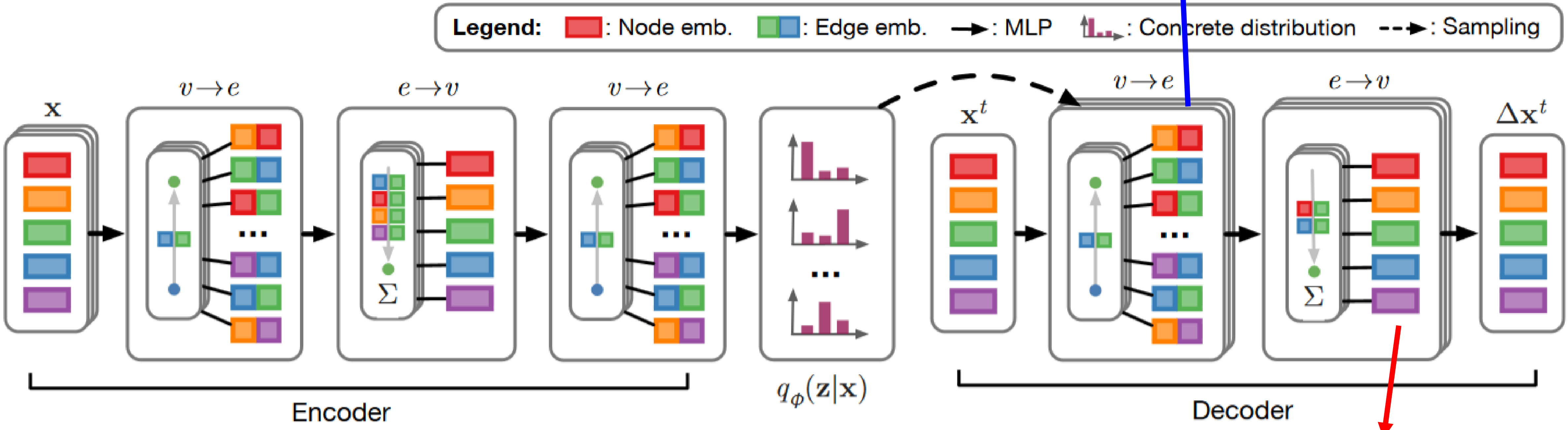
(pseudo-)discrete distribution from the Gumbel-softmax

Kipf, Thomas, et al. "Neural relational inference for interacting systems." *arXiv preprint arXiv:1802.04687* (2018).

Neural relational inference

Decoder

$$\tilde{\mathbf{h}}_{(i,j)}^{(t)} = \sum_k z_{ij,k} \tilde{f}_e^k \left([\mathbf{x}_i^{(t)}, \mathbf{x}_i^{(t)}] \right)$$



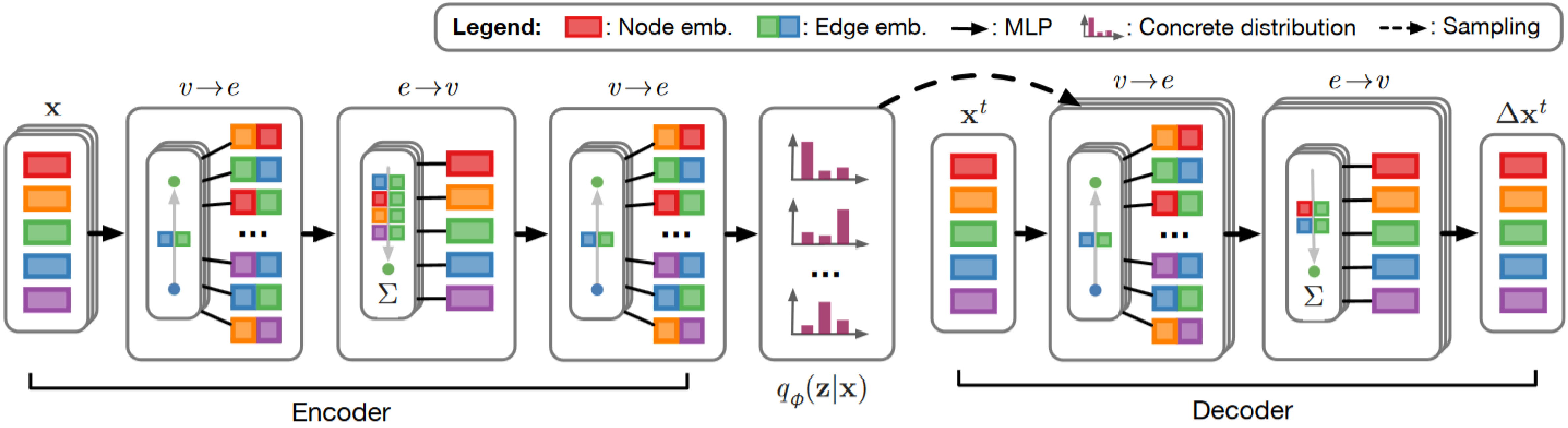
$$\boldsymbol{\mu}_j^{(t+1)} = \mathbf{x}_j^{(t)} + \tilde{f}_v \left(\sum_{i \neq j} \tilde{\mathbf{h}}_{(i,j)}^{(t)} \right)$$

$$p\left(\mathbf{x}_j^{(t+1)} | \mathbf{x}_j, \mathbf{z}\right) = \mathcal{N}\left(\boldsymbol{\mu}_j^{(t+1)}, \sigma^2 \mathbf{I}\right)$$

Kipf, Thomas, et al. "Neural relational inference for interacting systems." *arXiv preprint arXiv:1802.04687* (2018).

Neural relational inference

Decoder



- ✓ Encoder find uncerlying physical law, which is the relation between particles
- ✓ Decoder updates and inferes updates the particle's position at next time step at training and test stage, respectively.

Kipf, Thomas, et al. "Neural relational inference for interacting systems." *arXiv preprint arXiv:1802.04687* (2018).

KAIST CHEMISTRY

Thank you!

I am grateful to Ryan, YJ and Chloe at Kakao Brain for their fruitful comments and discussions.

KAIST