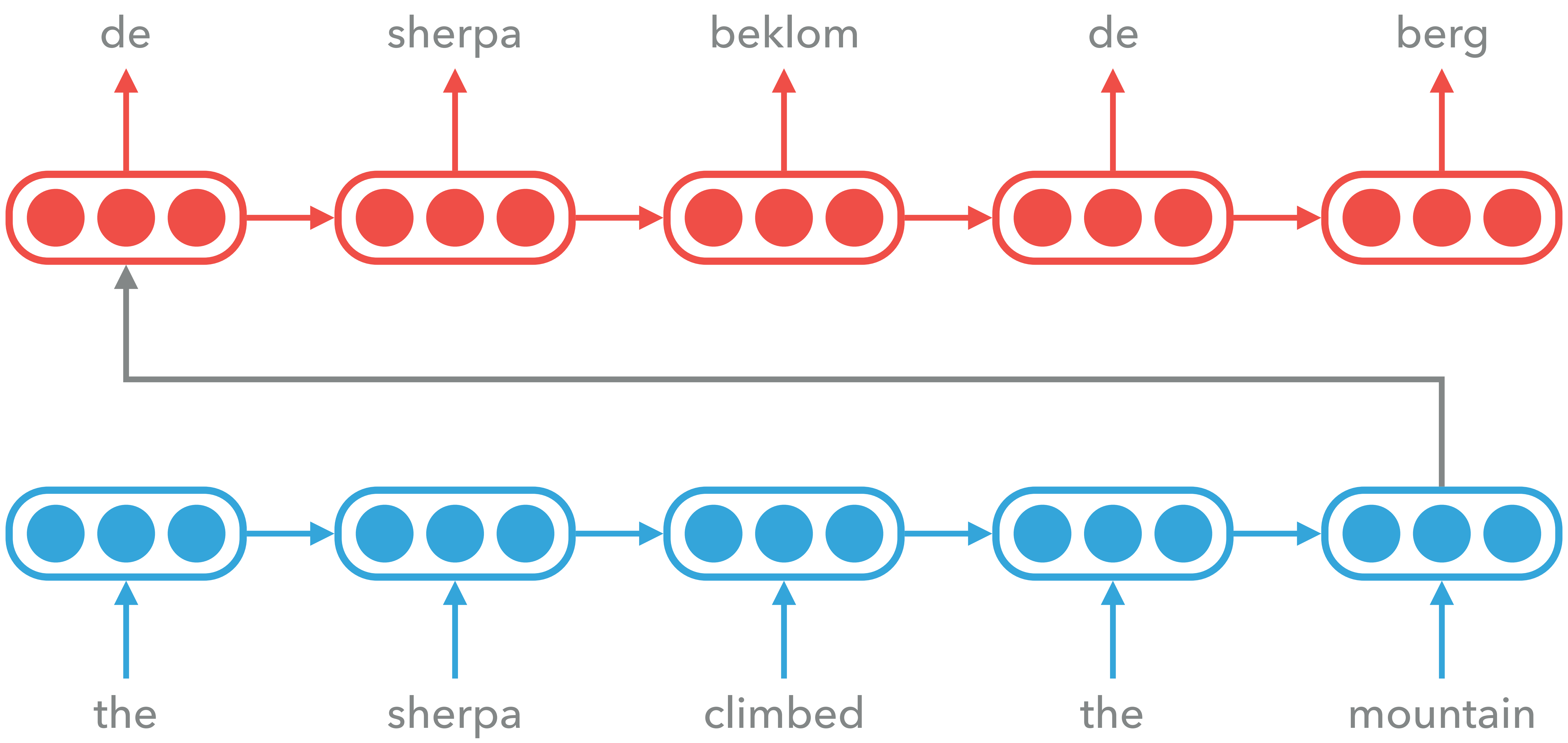
Graph Neural Networks in NLP

Ivan Titov

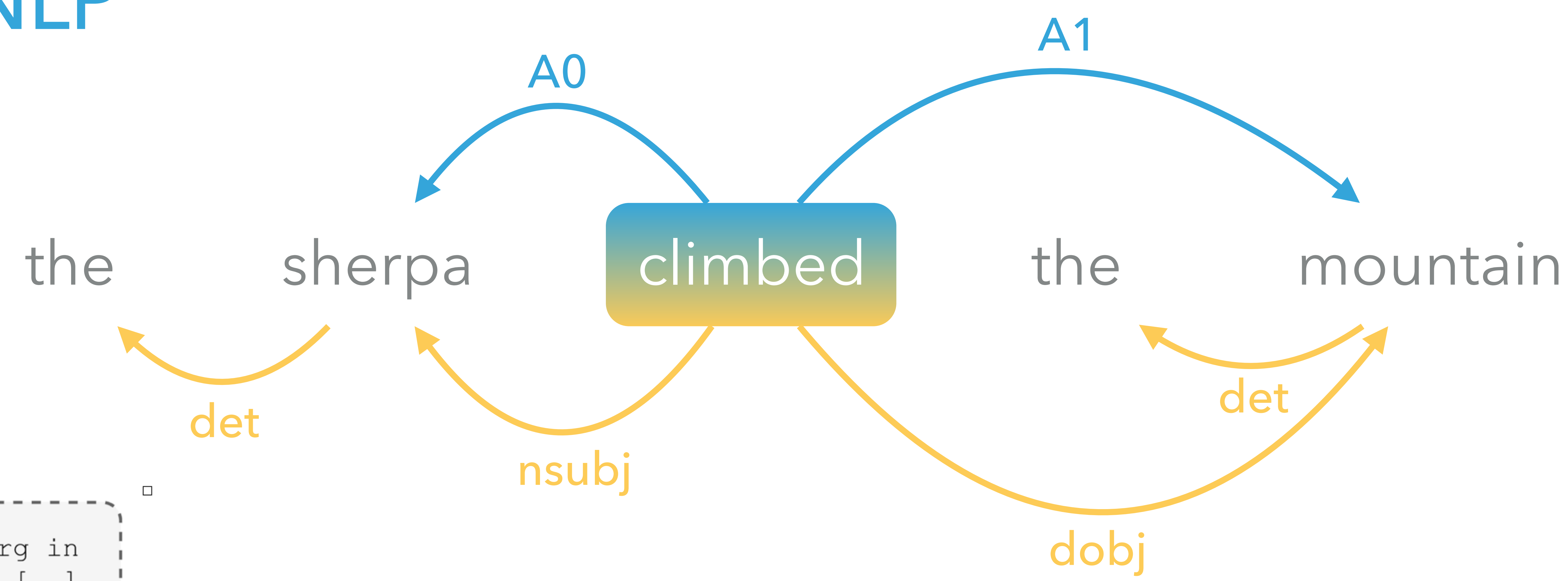
with Nicola De Cao, Diego Marcheggiani, Jasmijn Bastings, Michael Schlichtkrull, Wilker Aziz, …

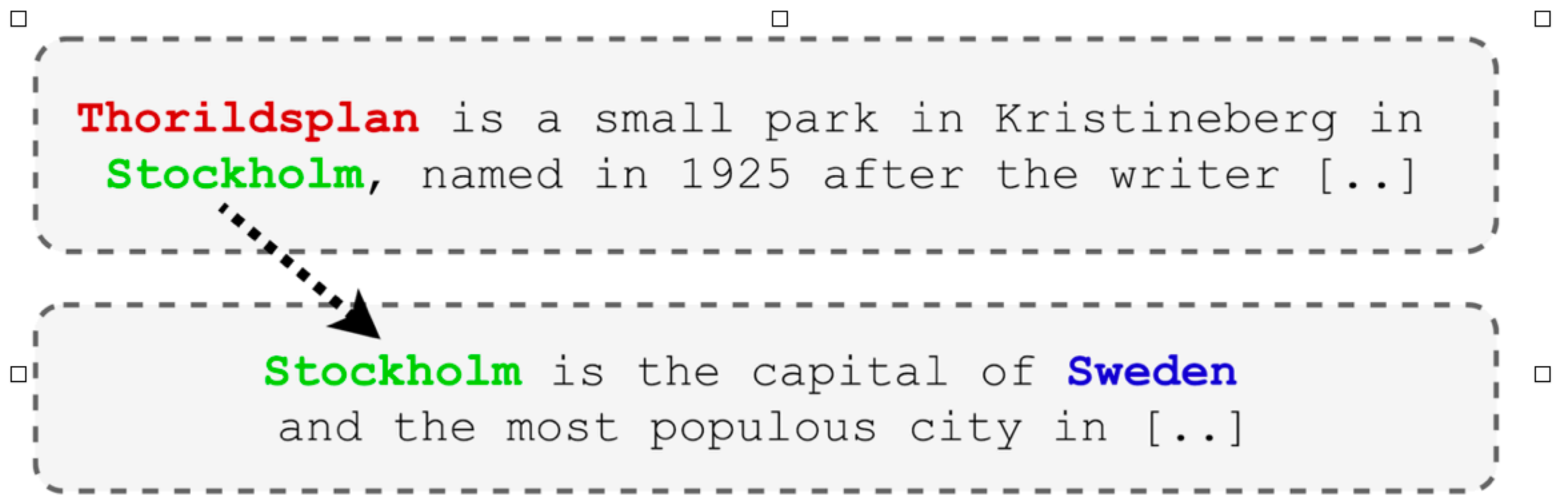
NLP in 2020

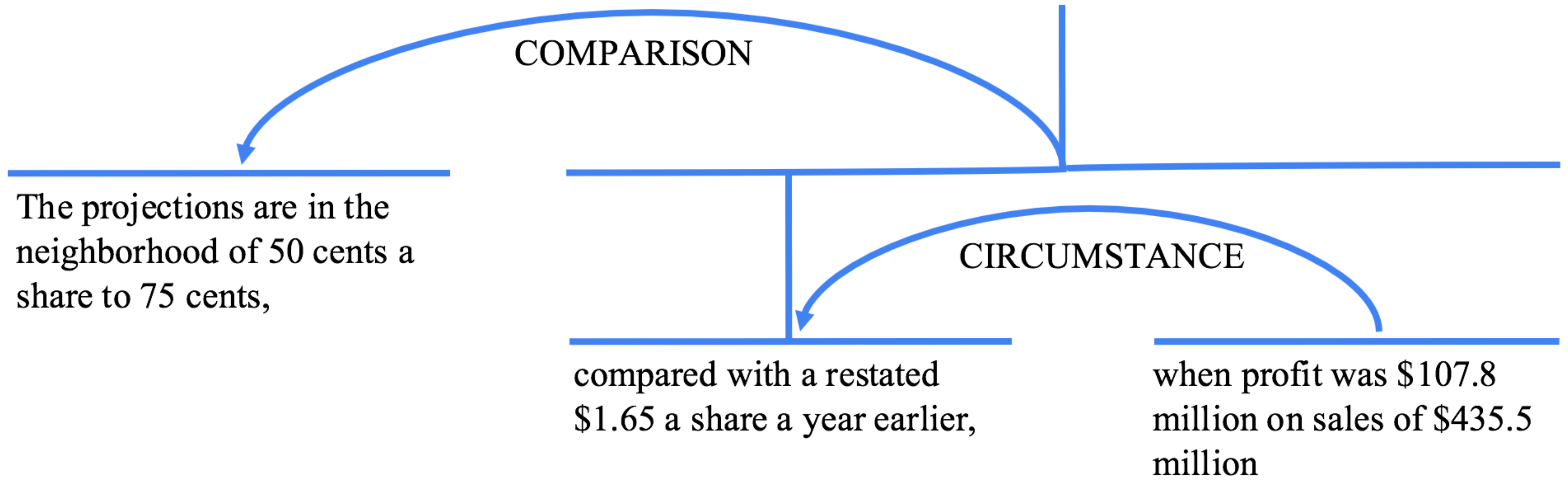


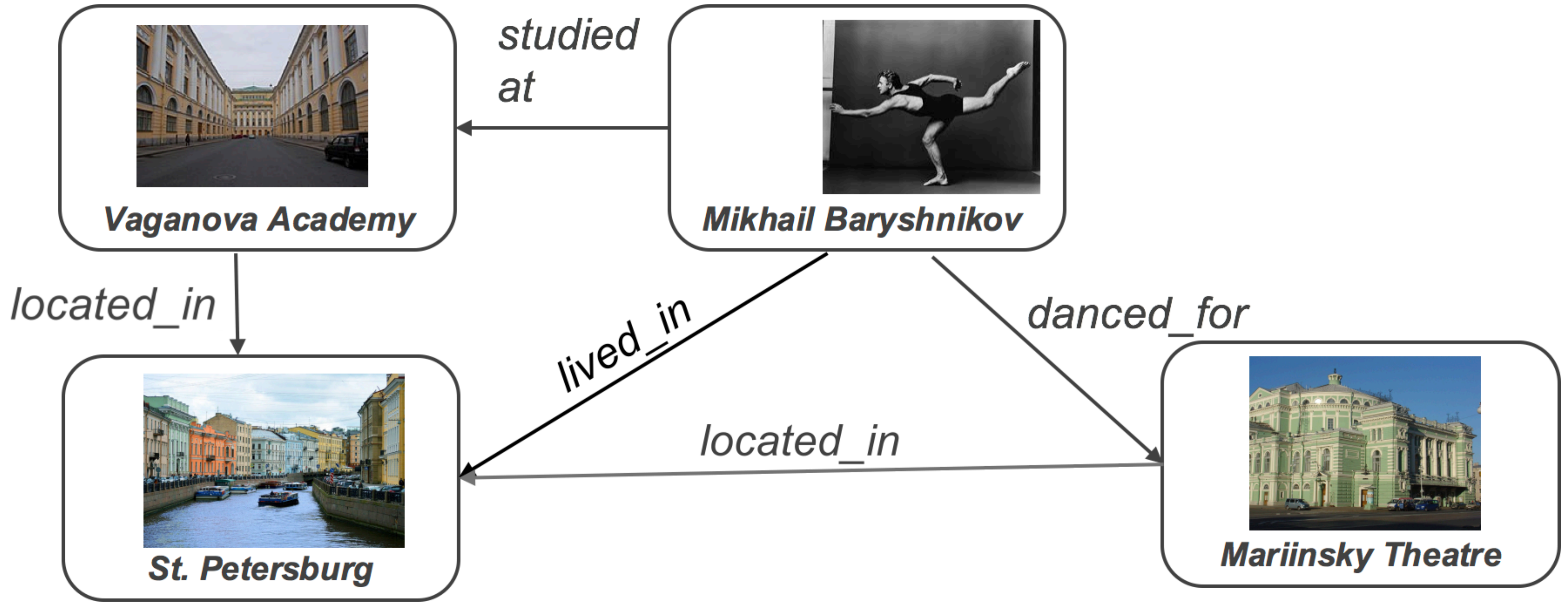
- Can we inject prior knowledge about language or world into NLP models?

Graph Structures in NLP







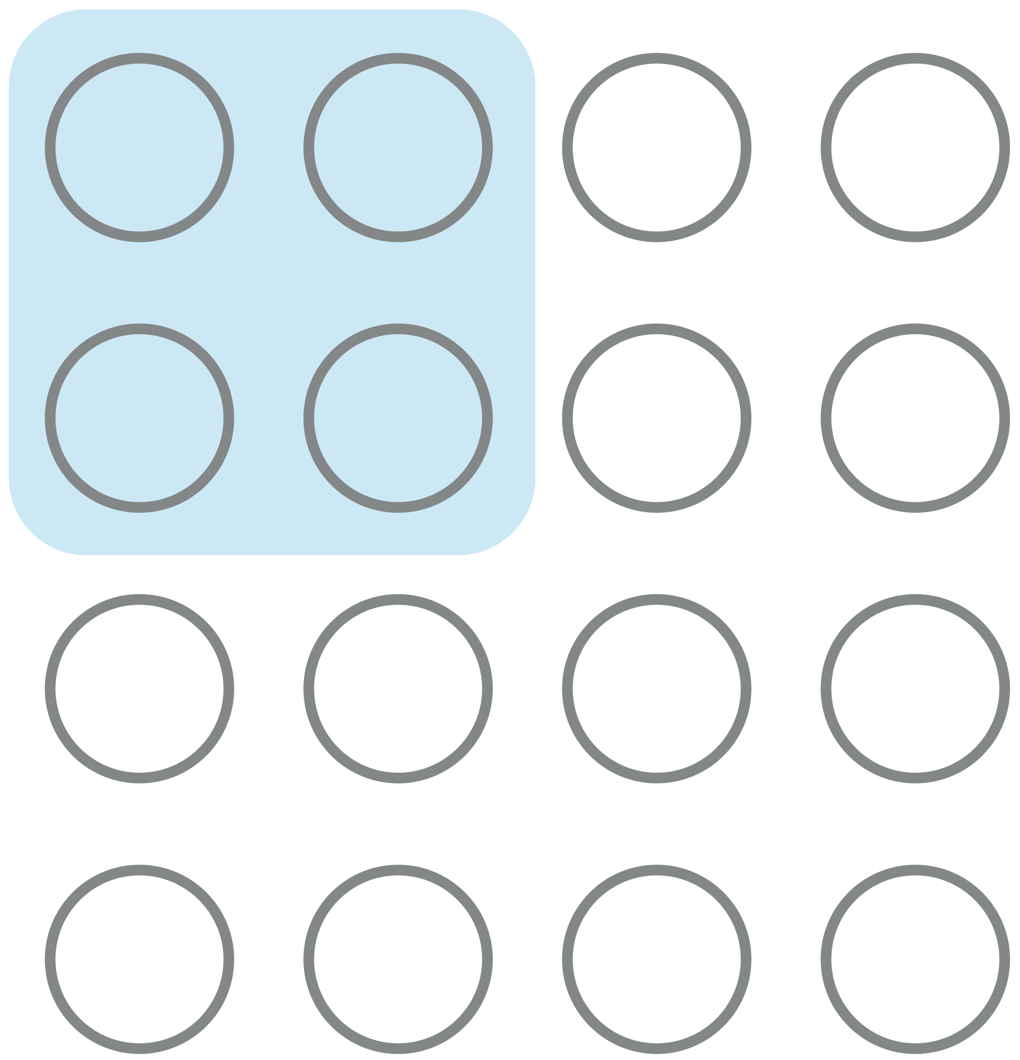


Outline

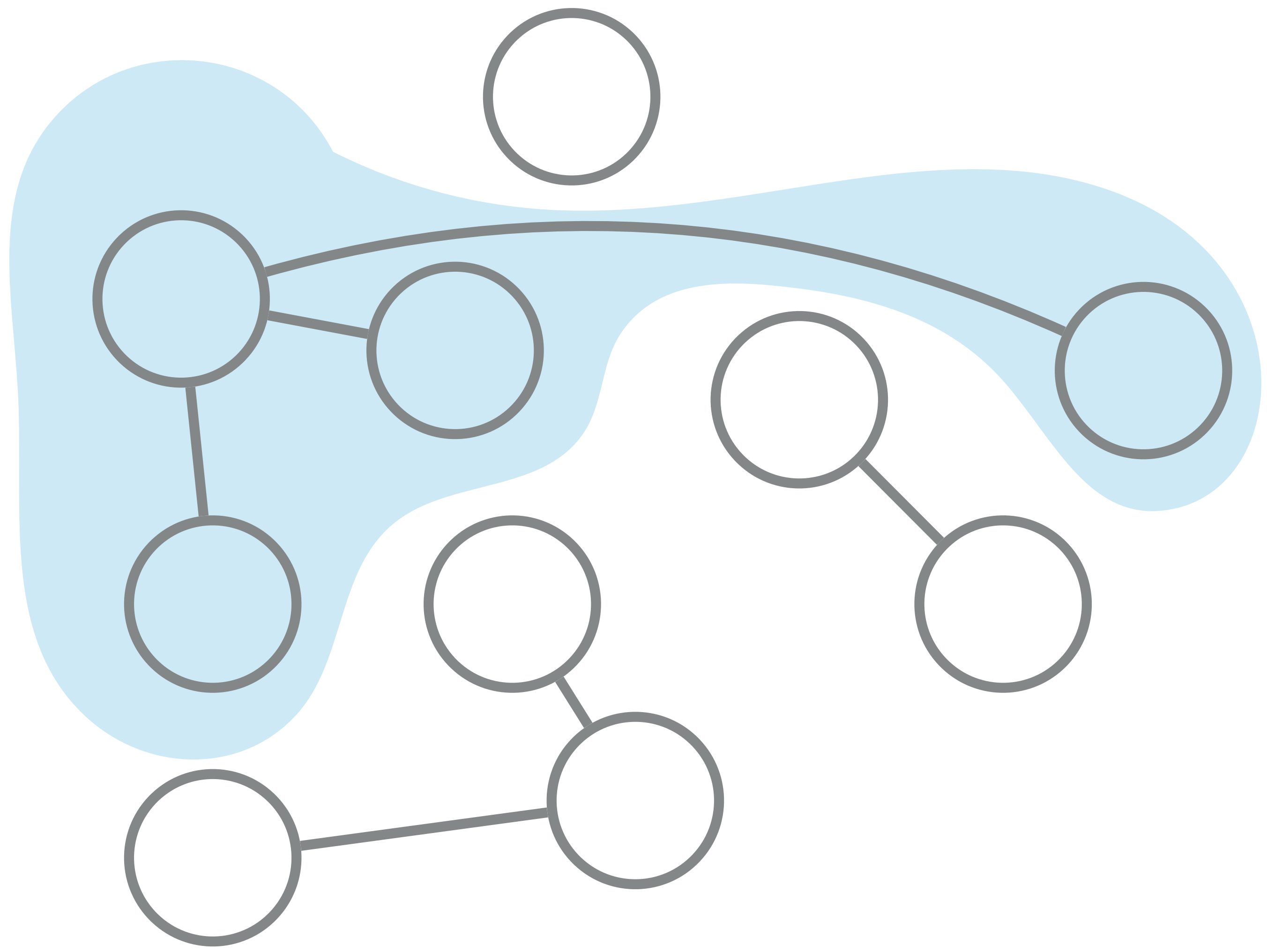
- Graph neural networks
- Incorporating structure in neural encoders
- Modeling and integrating knowledge (e.g., knowledge bases)
- Advanced topics

Graph Neural Networks

Convolution vs Graph Convolution



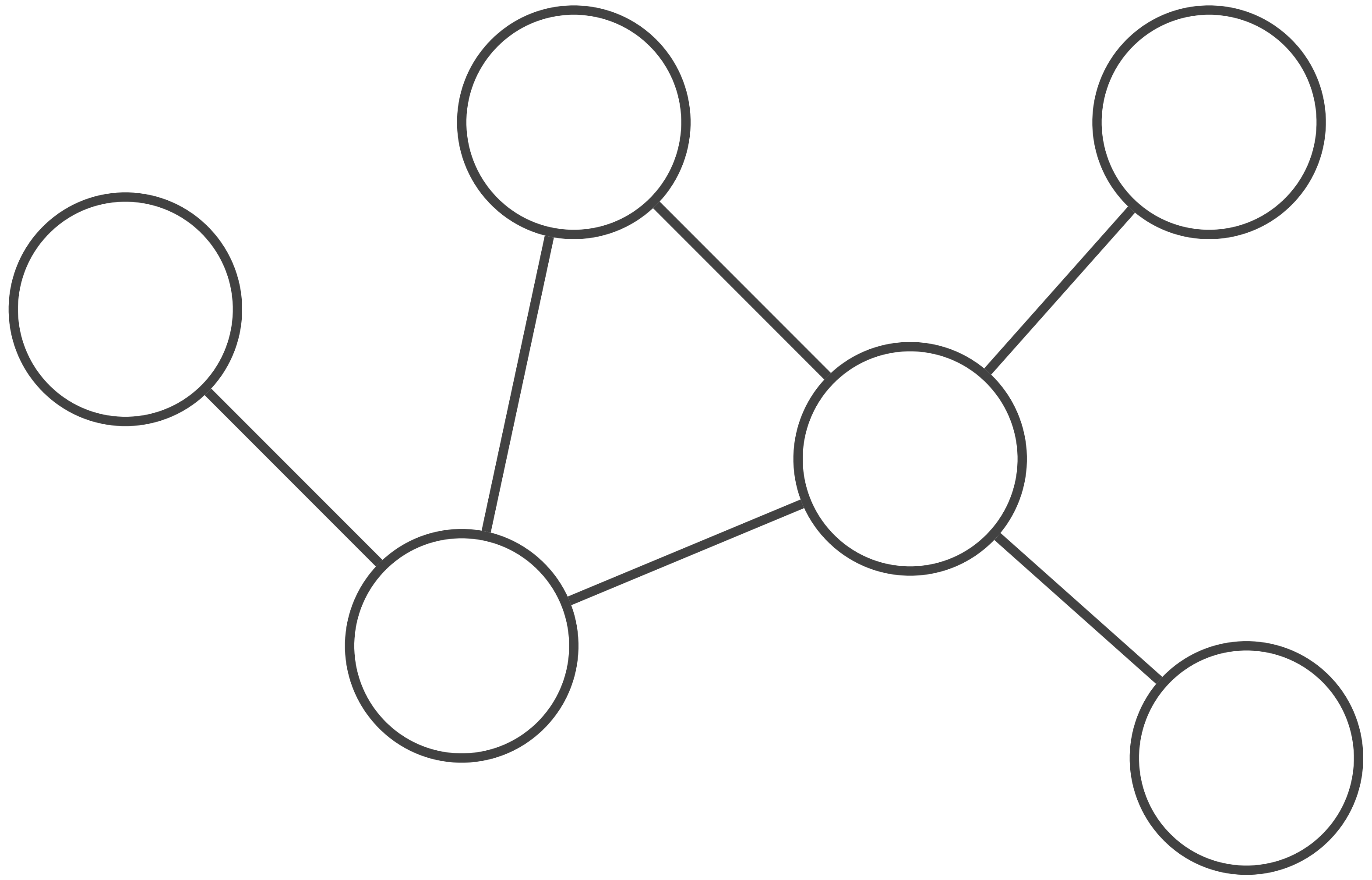
2D Convolution



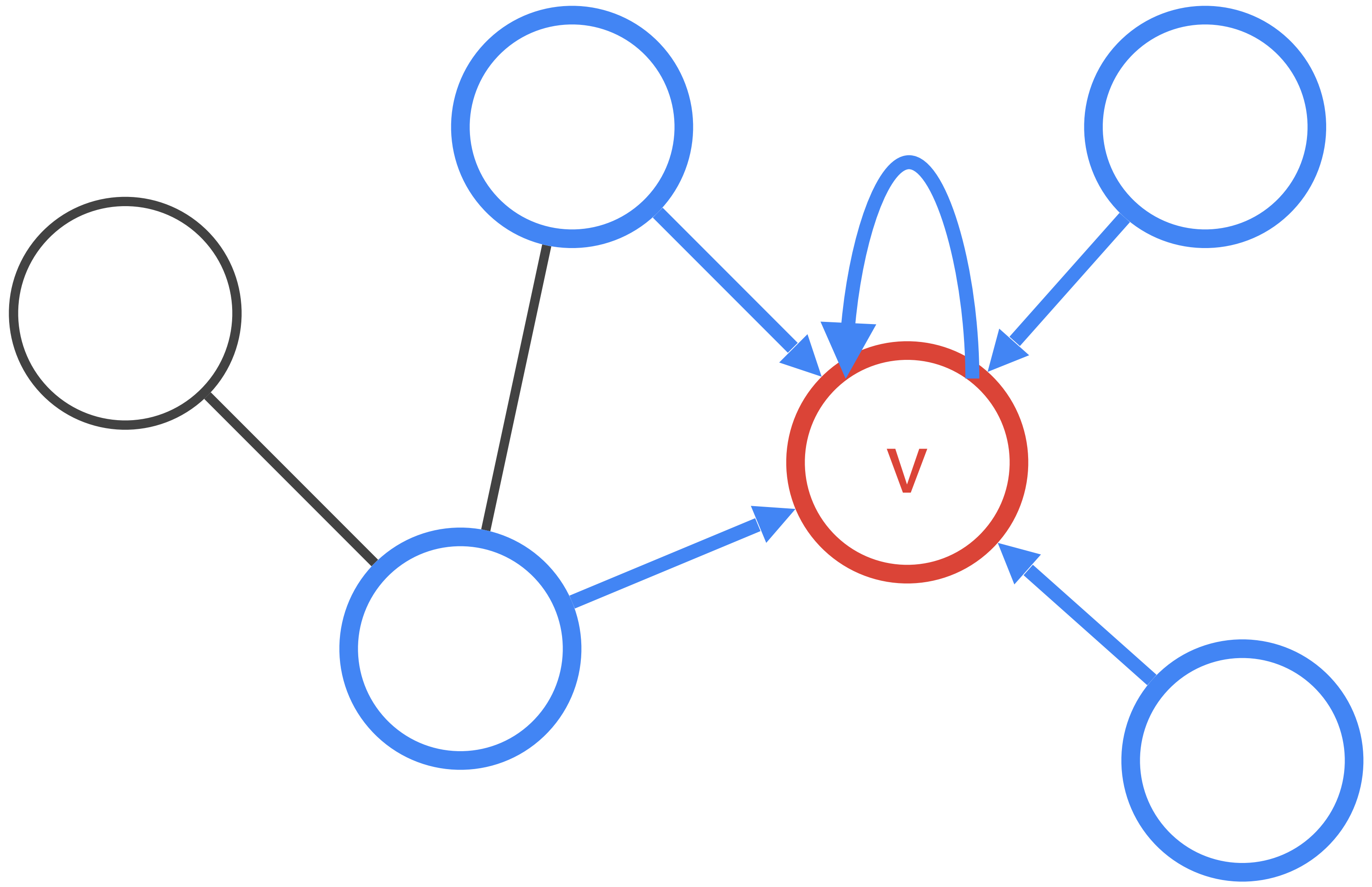
Graph convolution

We will treat terms graph convolutional networks (GCNs) and graph neural networks (GNNs) as synonyms

Graph Neural Networks: message passing



Undirected graph

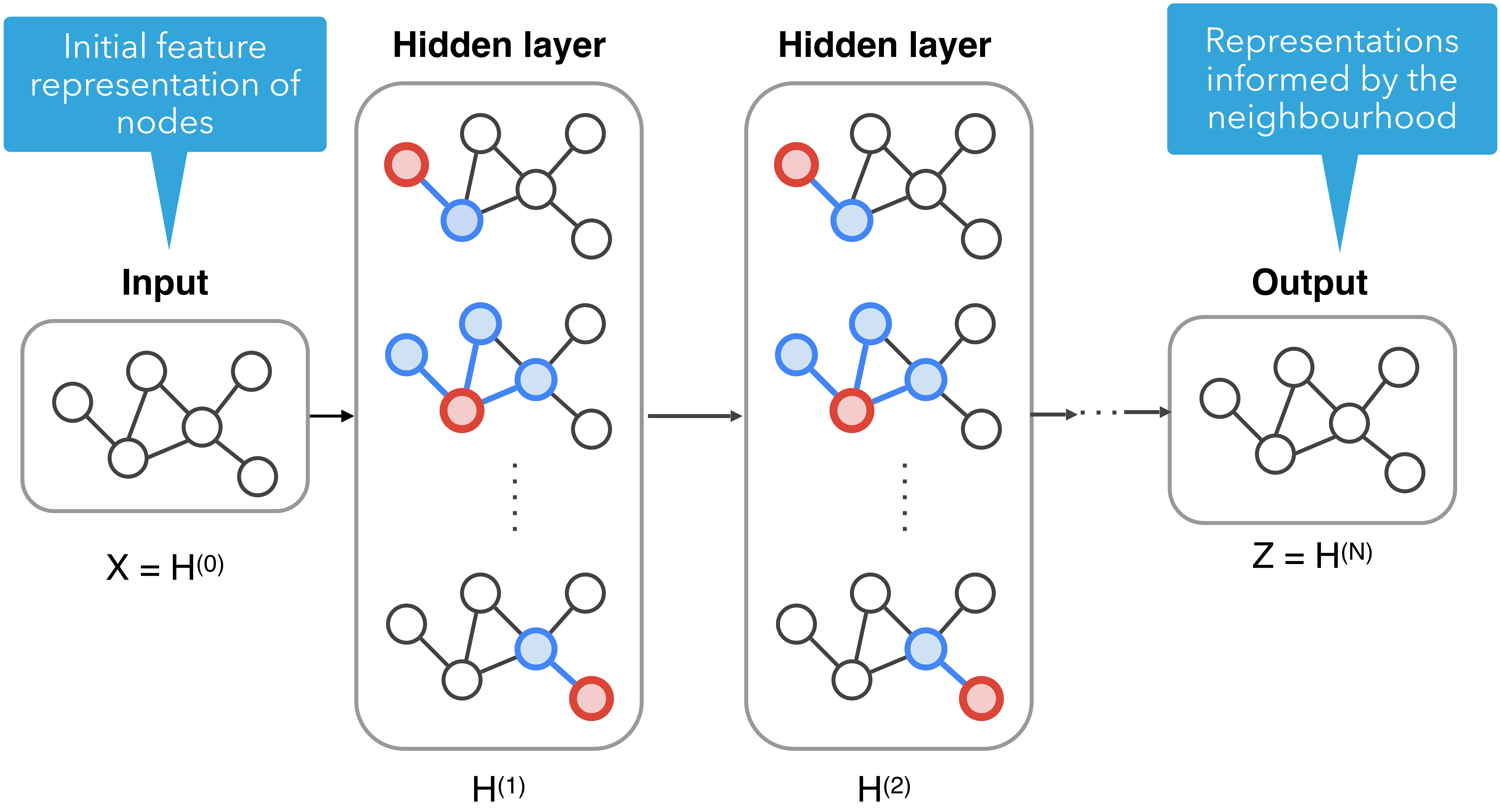


Update for node v

$$\mathbf{h}'_v = \mathrm{ReLU}\left(\sum_{u \in \mathrm{neighbors}(v)} W\mathbf{h}_u\right)$$

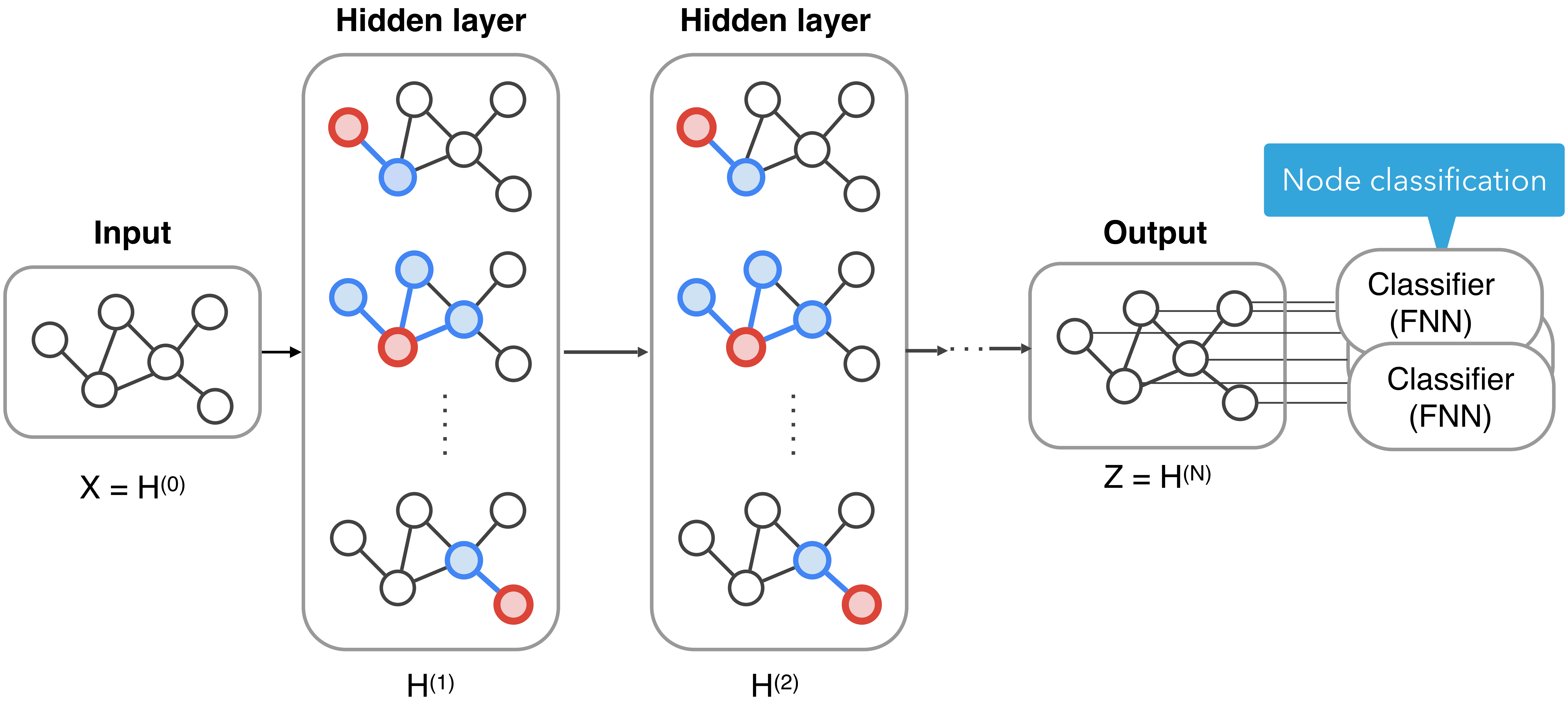
Kipf & Welling (2017). Related ideas earlier, e.g., Scarselli et al. (2009).

Graph Neural Networks: multiple layers



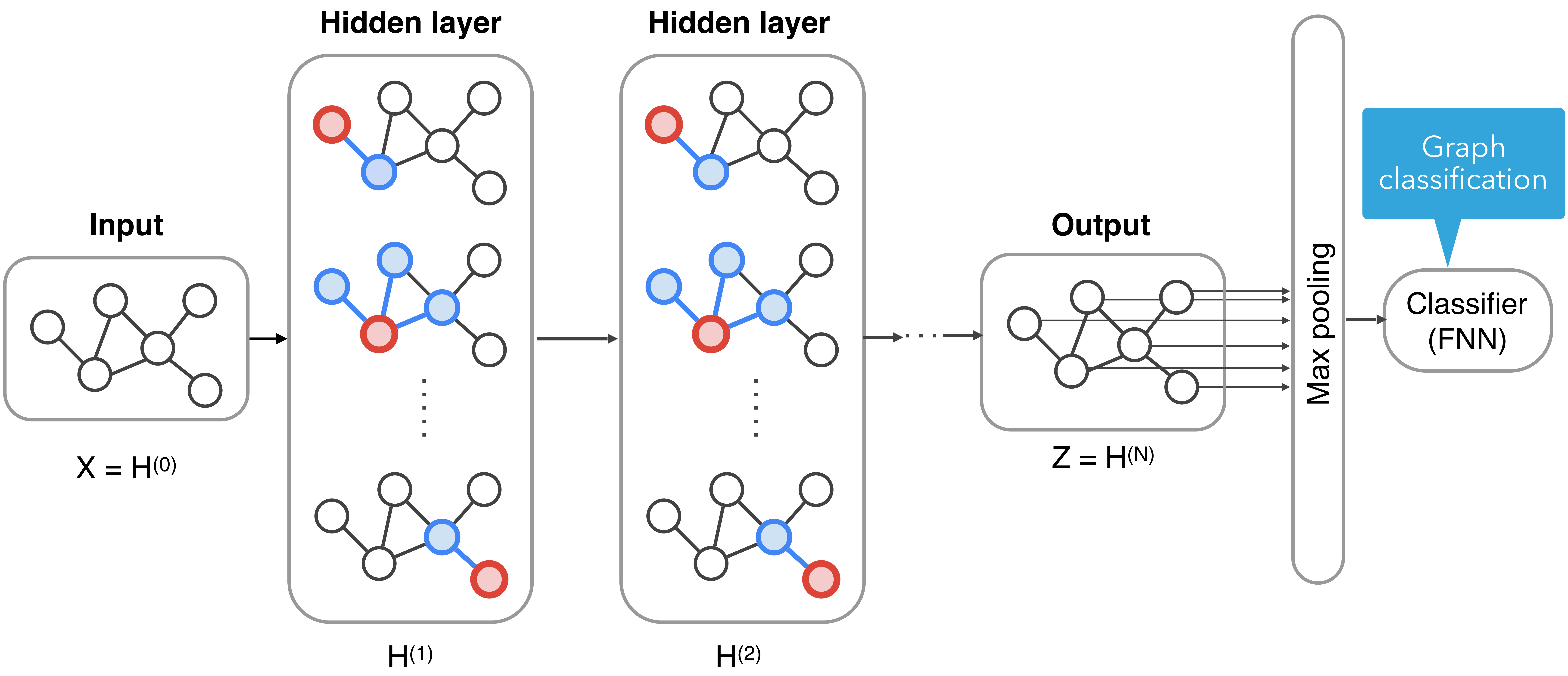
Parallelizable computation, can be made quite efficient (e.g., Hamilton, Ying and Leskovec (2017)).

Graph Neural Networks: multiple layers of message passing



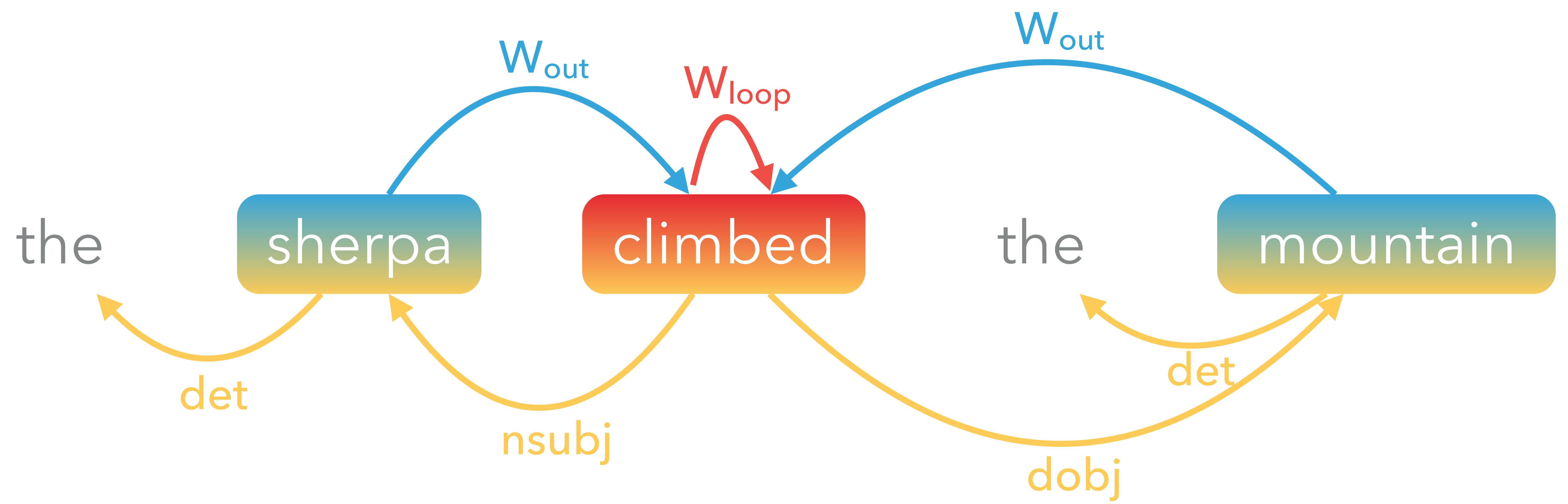
Parallelizable computation, can be made quite efficient (e.g., Hamilton, Ying and Leskovec (2017)).

Graph Neural Networks: multiple layers of message passing



Parallelizable computation, can be made quite efficient (e.g., Hamilton, Ying and Leskovec (2017)).

Incorporating edge labels and directions



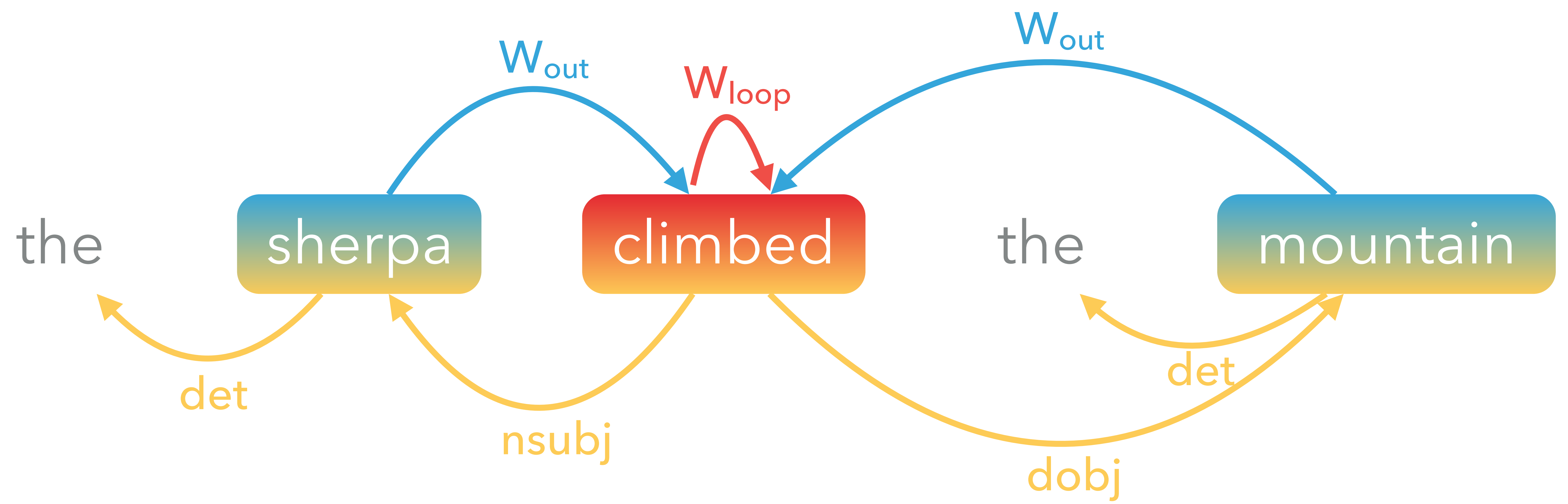
$$\mathbf{h}_v^{(t)} = \mathrm{ReLU}\left(\sum_{u \in \mathrm{neighbors}(v)} W_{lab(u,v)} \mathbf{h}_u^{(t-1)} \right)$$

Weight matrix for each direction and label

Syntactic GCN: Marcheggiani and Titov (EMNLP 2017)

Relational GCNs: Schlichtkrull*, Kipf*, Bloem, vd Berg, Titov, Welling (ESWC 2018)

Incorporating edge labels and directions



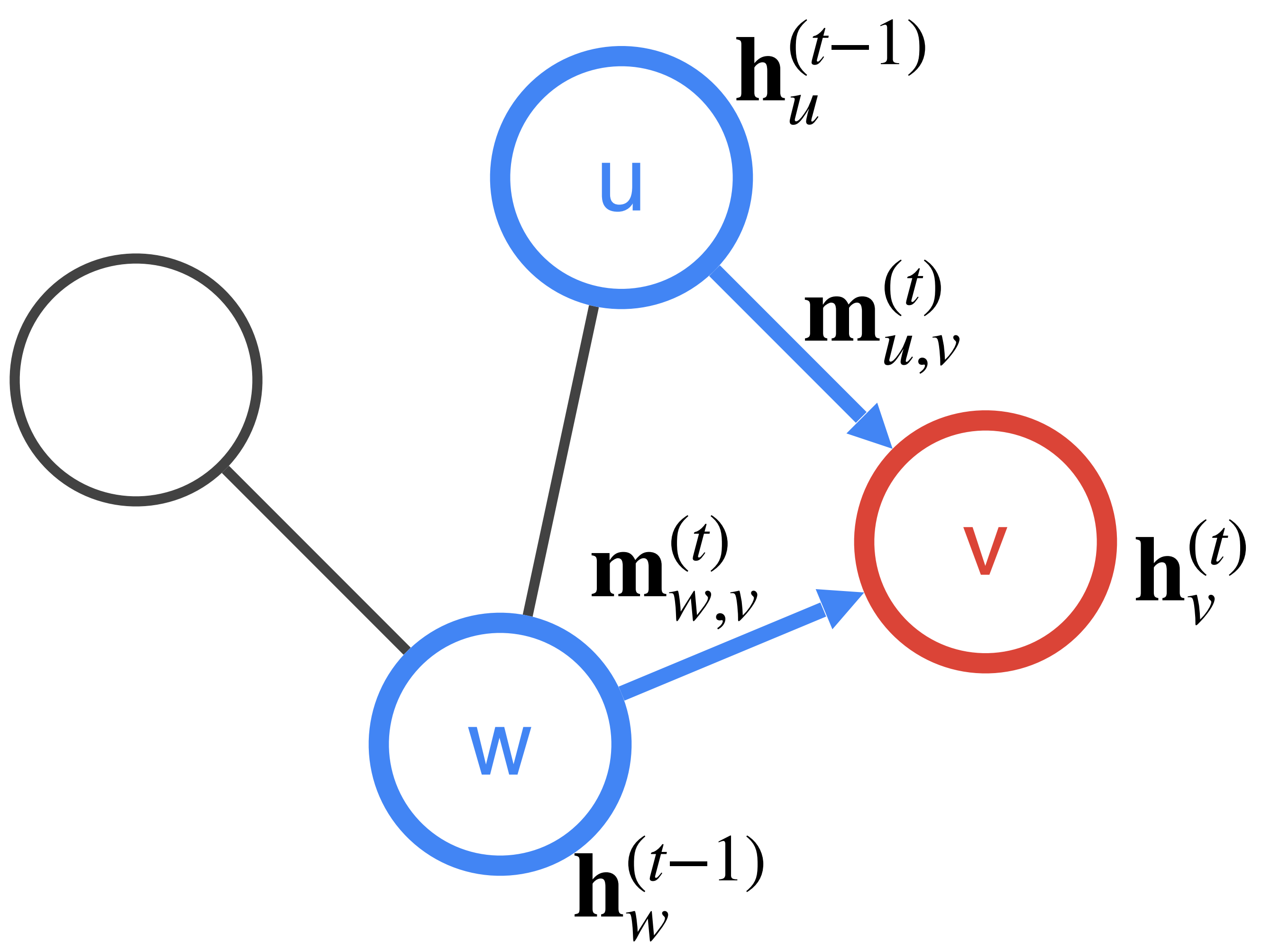
$$\mathbf{h}_v^{(t)} = \mathrm{ReLU}\left(\sum_{u \in \mathrm{neighbors}(v)} \gamma(u,v) \cdot W_{lab(u,v)} \mathbf{h}_u^{(t-1)} \right)$$

$$\gamma(u,v) = \sigma(\mathbf{u}_{lab(u,v)} \mathbf{h}_u^{(t-1)})$$

Sigmoidal 'gates' for edges; weight messages according to their importance

Syntactic GCN: Marcheggiani and Titov (EMNLP 2017)

Message passing GNNs



Message (layer *t*): $\mathbf{m}_{u,v}^{(t)} = \text{message}_t\left(\mathbf{h}_u^{(t-1)}, \mathbf{h}_v^{(t-1)}, lab(u,v)\right)$

Node representation: $\mathbf{h}_v^{(t)} = \text{aggregate}_t\left(\sum_u \mathbf{m}_{u,v}^{(t)}\right)$

The summation ensures invariance to permutation of neighbours

See comparison of labeled-graph GNNs in Brockschmidt (ICML 2020)

Expressivity of GNNs

Roughly: GNNs are Turing universal (roughly) if

- aggregation and message functions are “sufficiently expressive”
- nodes can distinguish each other (*non-anonymity*)

E.g., Maron, Fetaya, Segol, and Lipman, 2019
Keriven and Peyre, 2019, Loukas, 2020

Given that we learn GNNs with SGD from finite-size datasets the universality results are not so practically relevant (?)

What about what is not learnable with a given architecture?

What is not learnable?

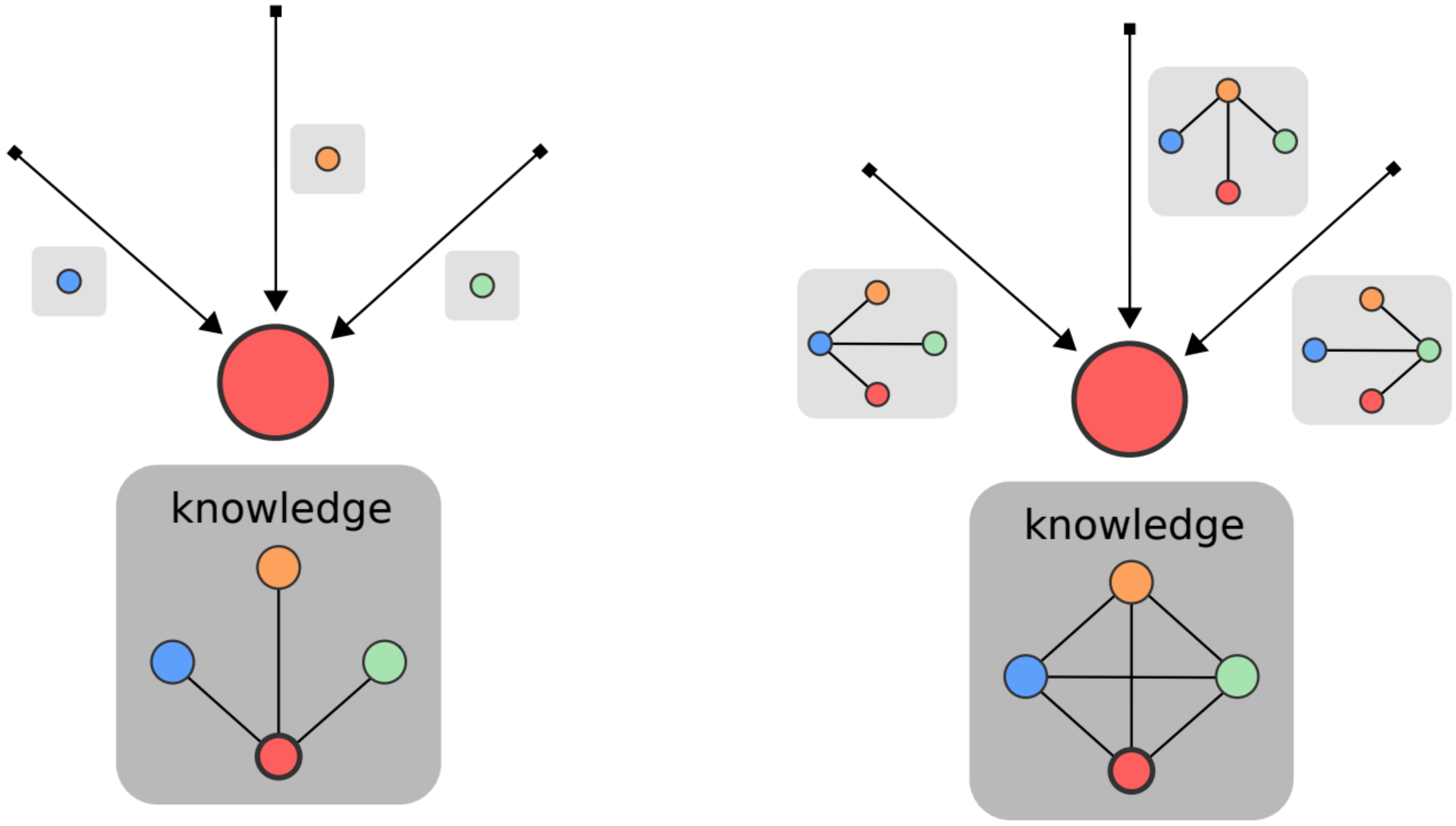
Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, and Grohe (AAAI 2018)
Xu, Hu, Leskovec, and Jegelka (ICLR 2019)

A large difference depending on whether nodes can distinguish each other and not ("anonymity")

Anonymity can be broken by (unique) features or ids (e.g., position ids if nodes are words)

What does red node know about topology?

No anonymity



After one hop

After two hops

Picture from Loukas (ICLR 2020)

What is not learnable?

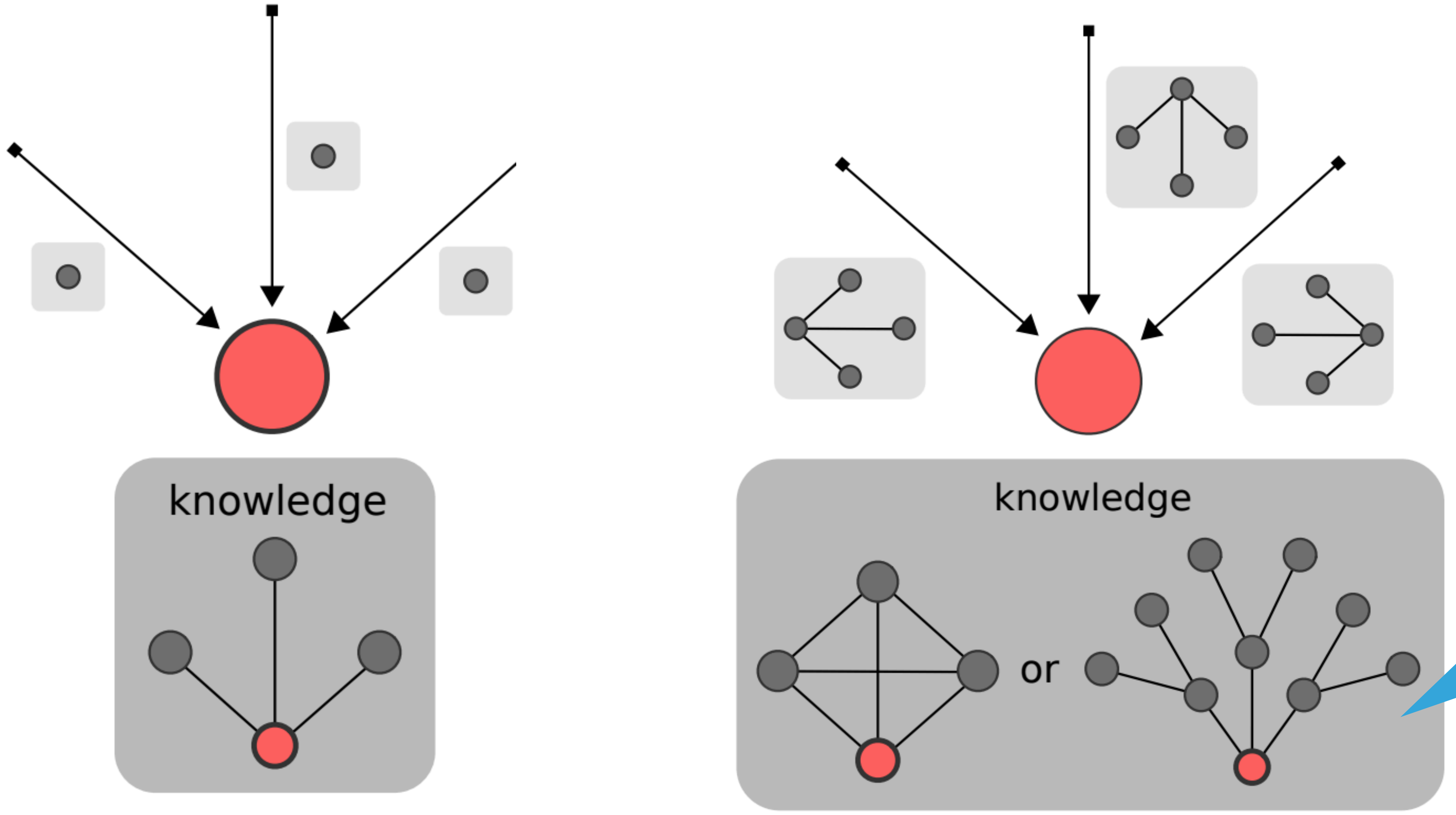
Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, and Grohe (AAAI 2018)
Xu, Hu, Leskovec, and Jegelka (ICLR 2019)

A large difference depending on whether nodes can distinguish each other and not ("anonymity")

Anonymity can be broken by (unique) features or ids (e.g., position ids if nodes are words)

What does red node know about topology?

Anonymity



After one hop After two hops

c.f. Weisferler-Leman heuristic for graph isomorphism

Picture from Loukas (ICLR 2020)

What is not learnable?

Loukas 2020 (ICLR 2020)

Characterization of depth *d* and width *w* of GNNs required to solve problems on graphs with *n* vertices

problem	*bound*	*problem*	*bound*
cycle detection (odd)	$dw = \Omega(n/\log n)$	shortest path	$d\sqrt{w} = \Omega(\sqrt{n}/\log n)$
cycle detection (even)	$dw = \Omega(\sqrt{n}/\log n)$	max. indep. set	$dw = \Omega(n^2/\log^2 n)$ for $w = O(1)$
subgraph verification*	$d\sqrt{w} = \Omega(\sqrt{n}/\log n)$	min. vertex cover	$dw = \Omega(n^2/\log^2 n)$ for $w = O(1)$
min. spanning tree	$d\sqrt{w} = \Omega(\sqrt{n}/\log n)$	perfect coloring	$dw = \Omega(n^2/\log^2 n)$ for $w = O(1)$
min. cut	$d\sqrt{w} = \Omega(\sqrt{n}/\log n)$	girth 2-approx.	$dw = \Omega(\sqrt{n}/\log n)$
diam. computation	$dw = \Omega(n/\log n)$	diam. $3/2$-approx.	$dw = \Omega(\sqrt{n}/\log n)$

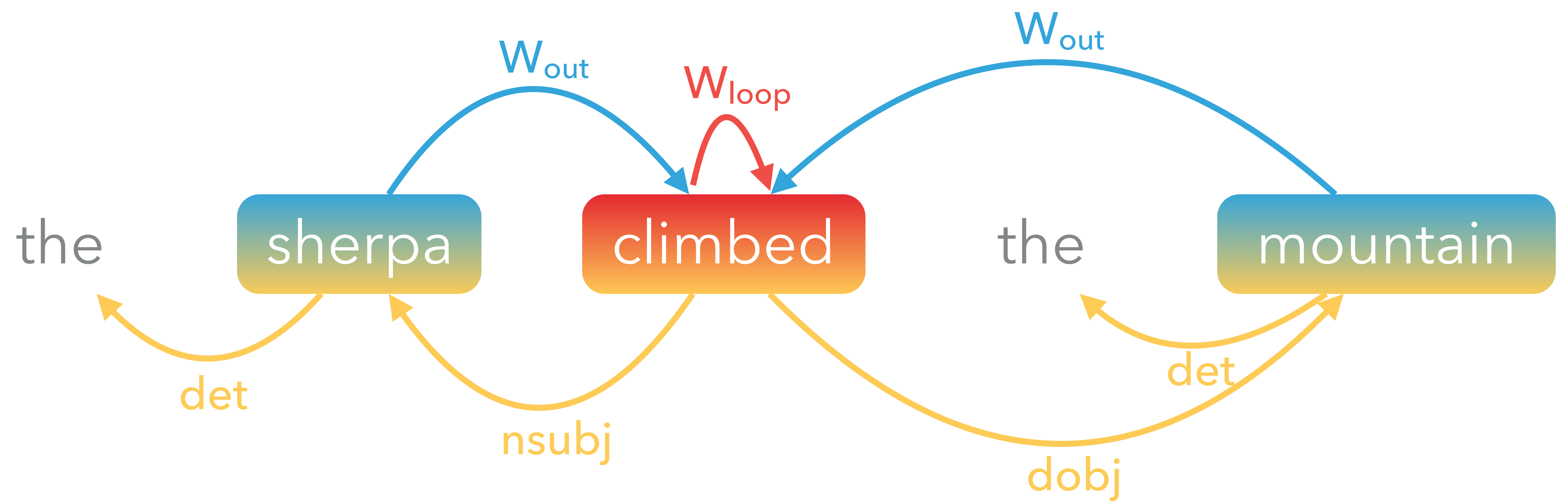
Why relevant to ML? Gives an idea whether a GNN can capture classes of features for a given problem

So far

- We defined GNNs, including for directed labeled graphs
- Looked in several versions
- Discussed their properties and limitations

GNNs as Encoders

Recall: Syntactic GCNs

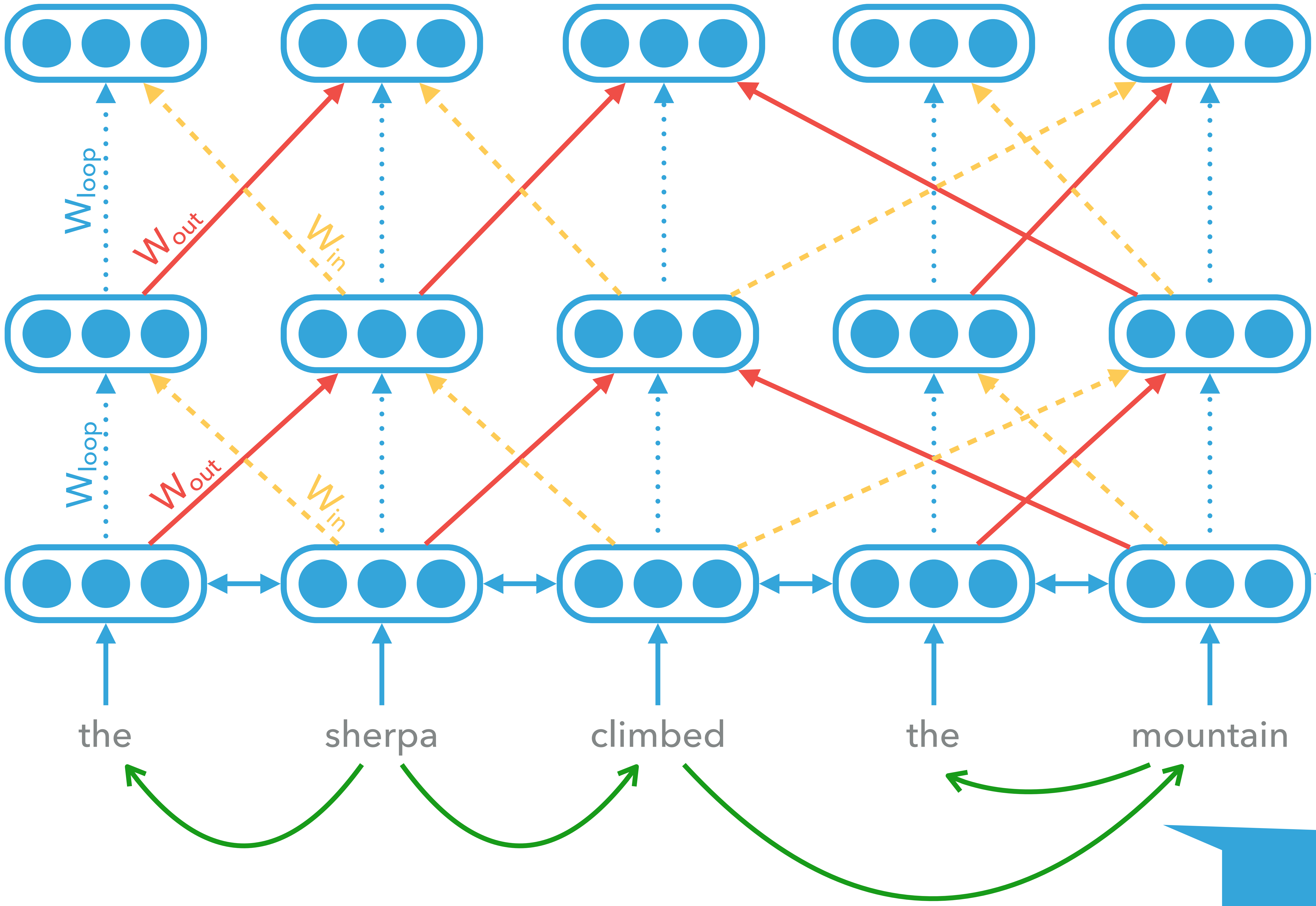


$$\mathbf{h}_v^{(t)} = \mathrm{ReLU}\left(\sum_{u \in \mathrm{neighbors}(v)} \gamma(u,v) \cdot W_{lab(u,v)} \mathbf{h}_u^{(t-1)} \right)$$

Syntactic GCN: Marcheggiani and Titov (EMNLP 2017)

Relational GCNs: Schlichtkrull*, Kipf*, Bloem, vd Berg, Titov, Welling (ESWC 2018)

Graph Convolutional Encoder

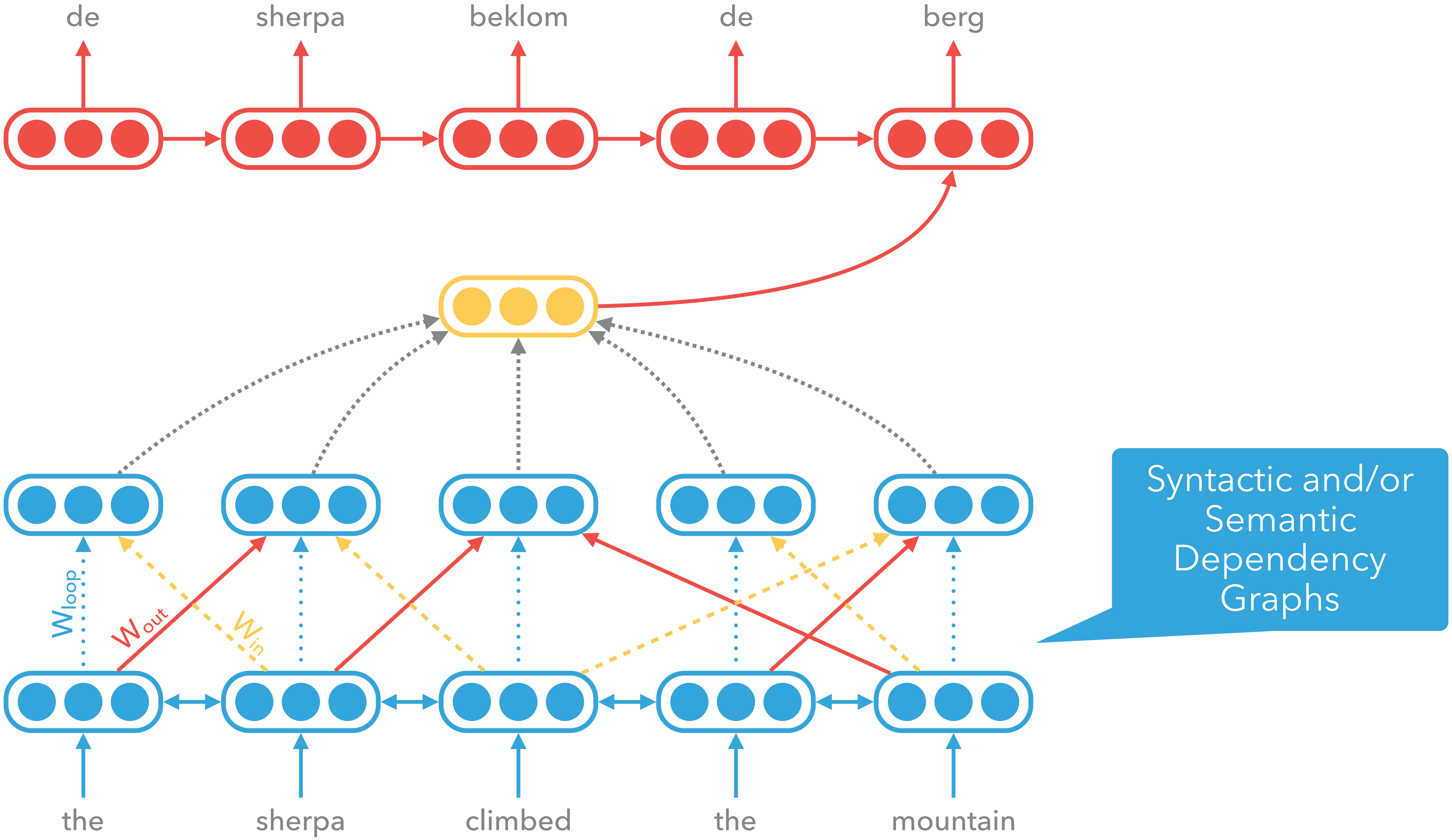


(deep) sentence encoder (e.g., LSTM / Transformer)

The sentence structure provides inductive bias

Marcheggiani and Titov (EMNLP, 2017)

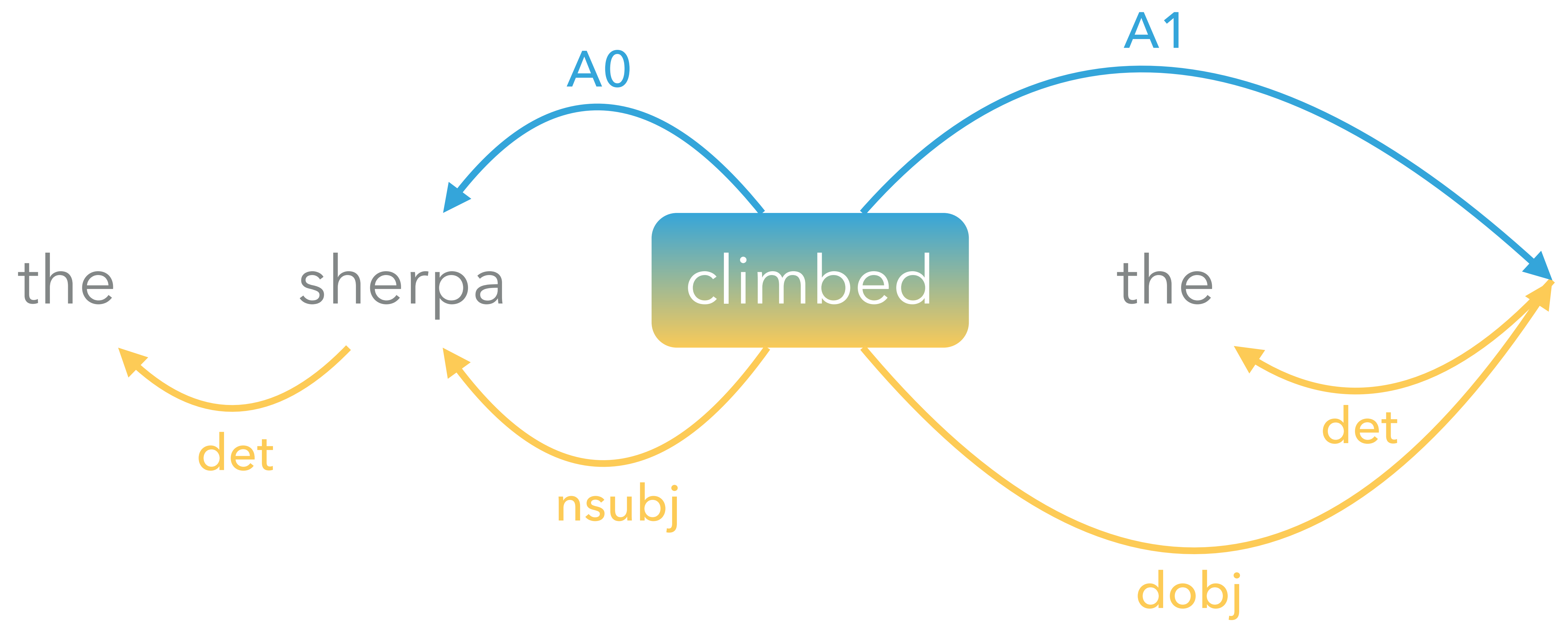
Syntactic- / Semantic -Aware Neural Machine Translation



Bastings et al. (EMNLP, 2017) / Marcheggiani, Bastings, Titov (NAACL, 2018)

Machine Translation with Syntax *and* Semantics

Separate GCN weights for syntactic and semantic edges



WMT'16 En-De	
Baseline	23.3
+ Sem	24.5
+ Syn	23.9
+ Syn + Sem	24.9

Bastings et al. (EMNLP, 2017) / Marcheggiani, Bastings, Titov (NAACL, 2018)

Multi-hop Question answering (Wikihop)

- Set of **<documents, question>** pairs constructed form text corpus and a knowledge base
- **Task:** multiple choice QA from a set of candidate answers
- Query constructed to force **reasoning across documents**

```
Thorildsplan is a small park in Kristineberg in
Stockholm, named in 1925 after the writer [..]
```

```
Stockholm is the capital of Sweden
and the most populous city in [..]
```

```
query: country Thorildsplan
candidates: {Denmark, Finland, Sweden, Italy, ...}
answer: Sweden
```

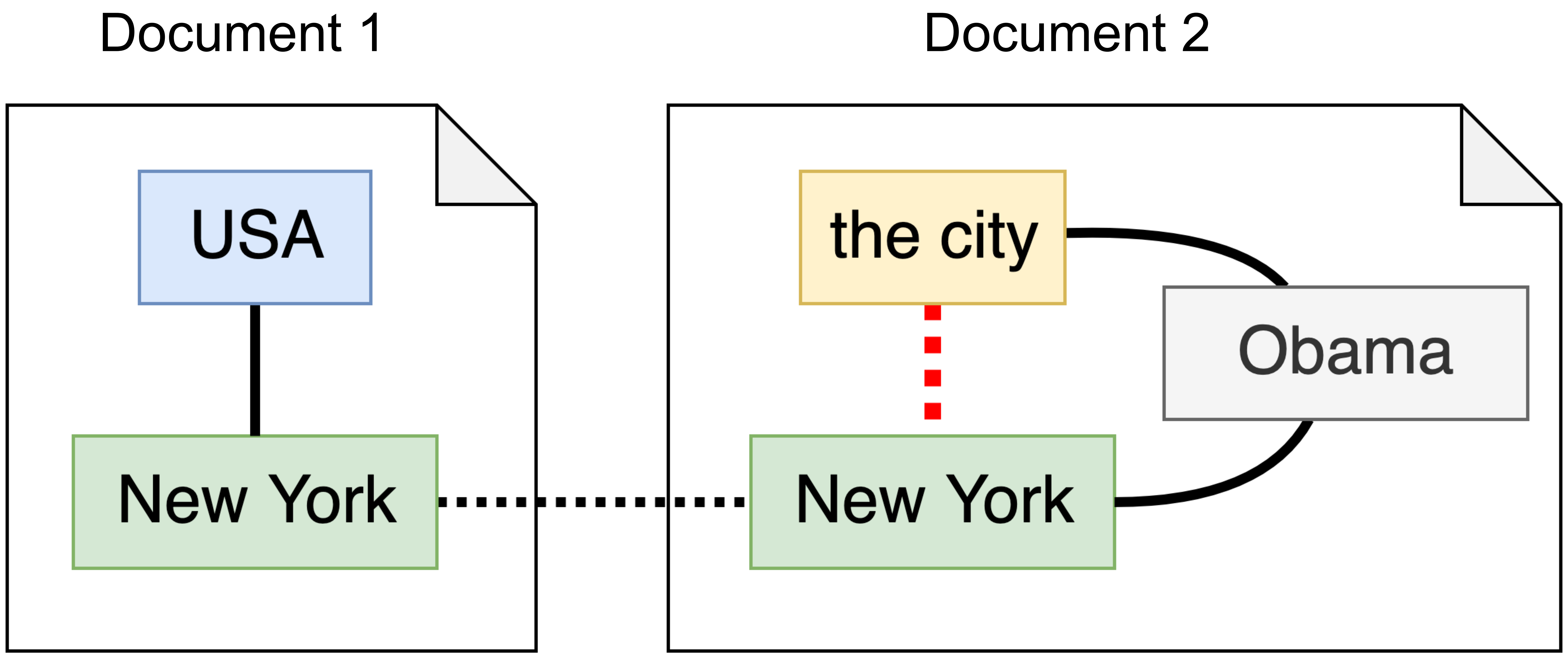
Welbl, Stenetorp, Riedel (TACL 2018)

Entity GCN

Nodes are mentions and we connect (assigning different edge-types):

- Mentions within the **same document**
- **Exact matches** across documents
- **Corefecences** (using external resolution system)
- **The complement graph**

Initial node representations: ELMo or GloVe

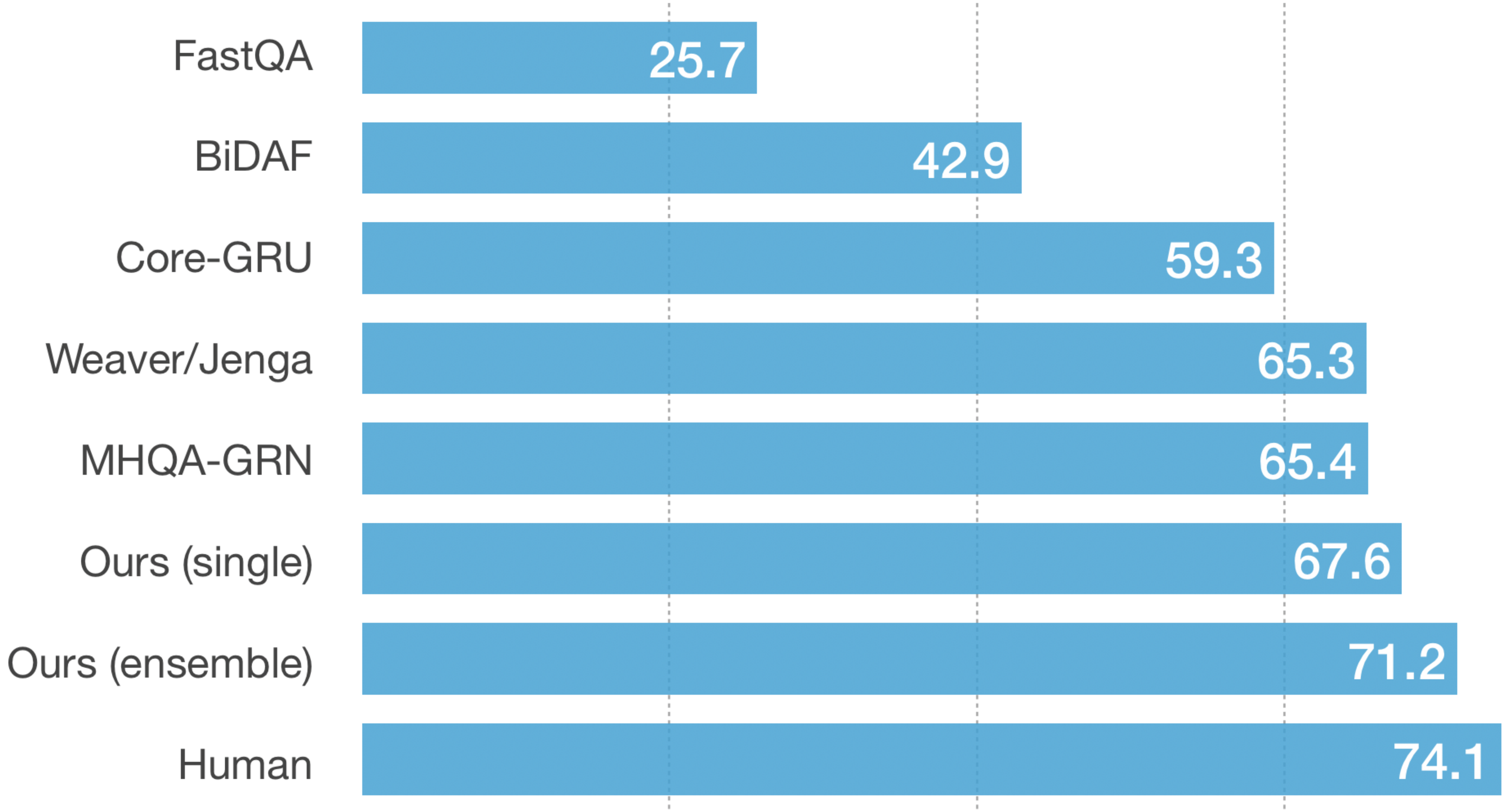


Applied a gated GNN similar to the one for syntax

(De Cao, Aziz, Titov, NAACL 2019)

Entity GCN results

- Best model’s result is close to human performance
- Entity-GCN is at least **5 times faster** to train than BiDAF
- Ensemble model **does not add overhead** since embeddings are computed only once!



Outdated but more recent methods use the same principles (e.g., graph neural networks with memory, better embeddings, …) - e.g., PathGCN (Tang et al., IJCAI 2020)

(De Cao, Aziz, Titov, NAACL 2019)

Graph-to-sequence models in NLP

GNNs encode structured data produce text, e.g.:

Meaning representations (e.g., AMR, logical form, SQL) to natural language sentences

Song, Zhang, Wang, GIldea (ACL 2018), Marcheggiani and Perez-Beltrachini (INLG 2018), Xu, Wu , Wang, Feng, Sheinin (EMNLP 2018),...

Scene graphs for images to captions

Yang, Tang, Zhang, Cai (CVPR 2019),....

Source code summarisation

Fernandes, Allamanis, Brockschmidt (ICLR 2019)

So far

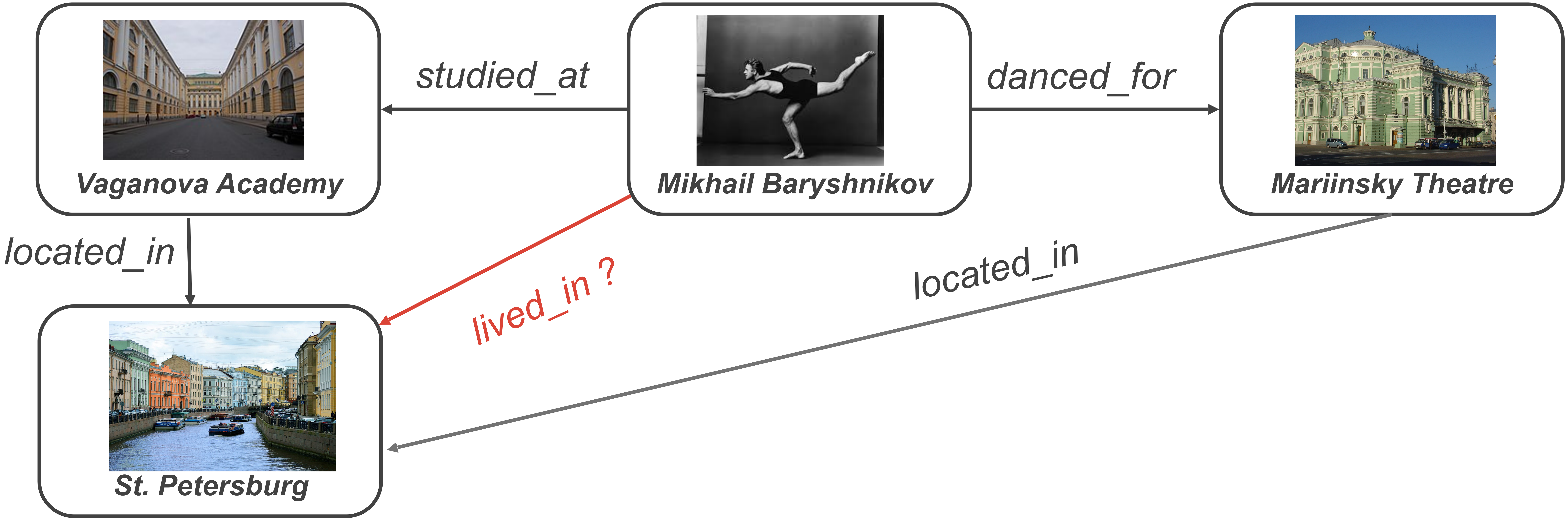
- Graphs in NLP can represent prior knowledge about text structure
- We often have tools and data to create / predict these graphs
- GNNs provide a simple way to integrate this prior knowledge

A more recent generation of models often borrow ideas from other model neural models (e.g., Transformers) but they key ingredients remain the same

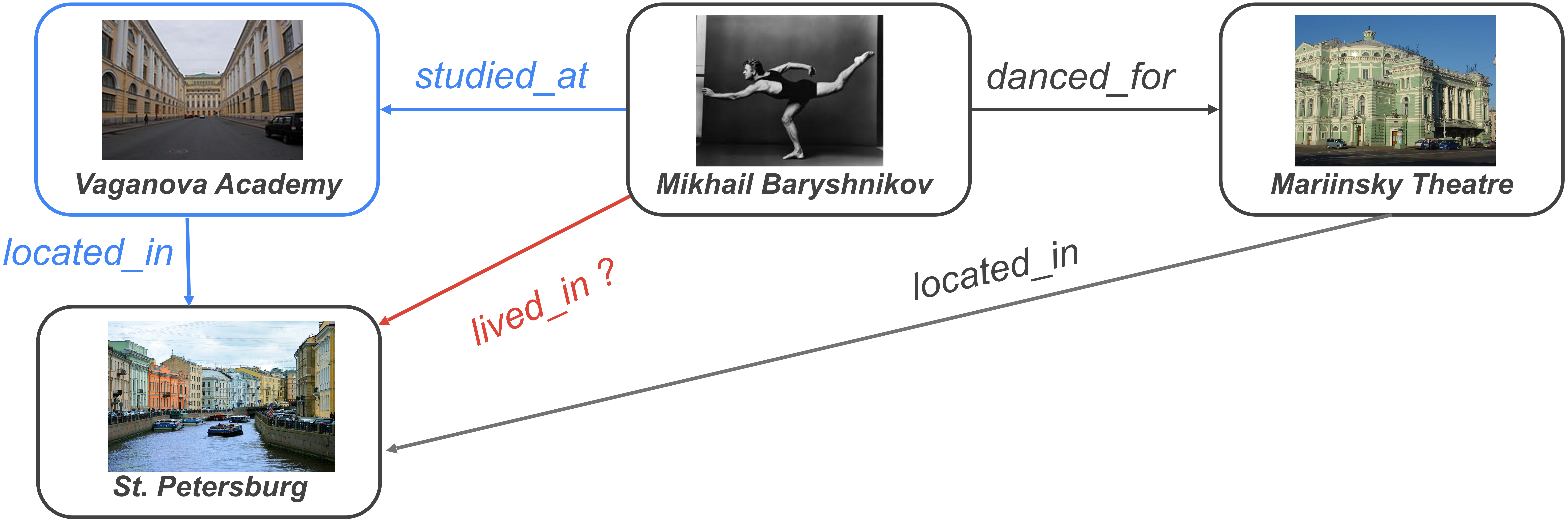
Questions?

GNNs for Knowledge Bases

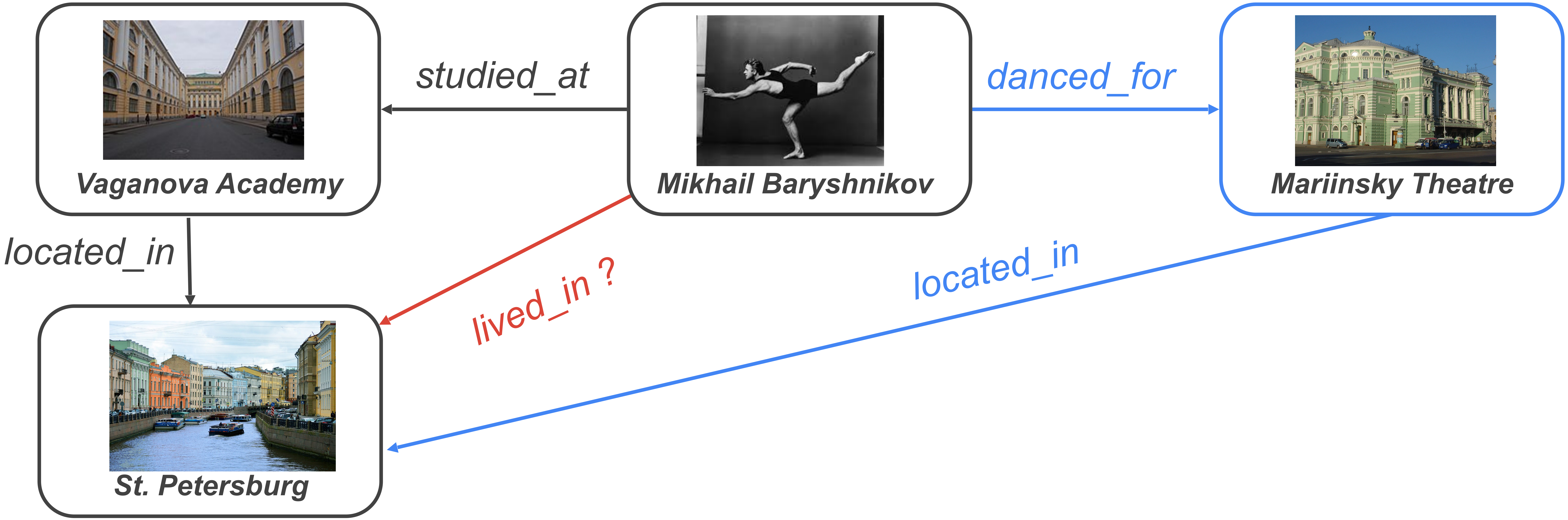
Link Prediction



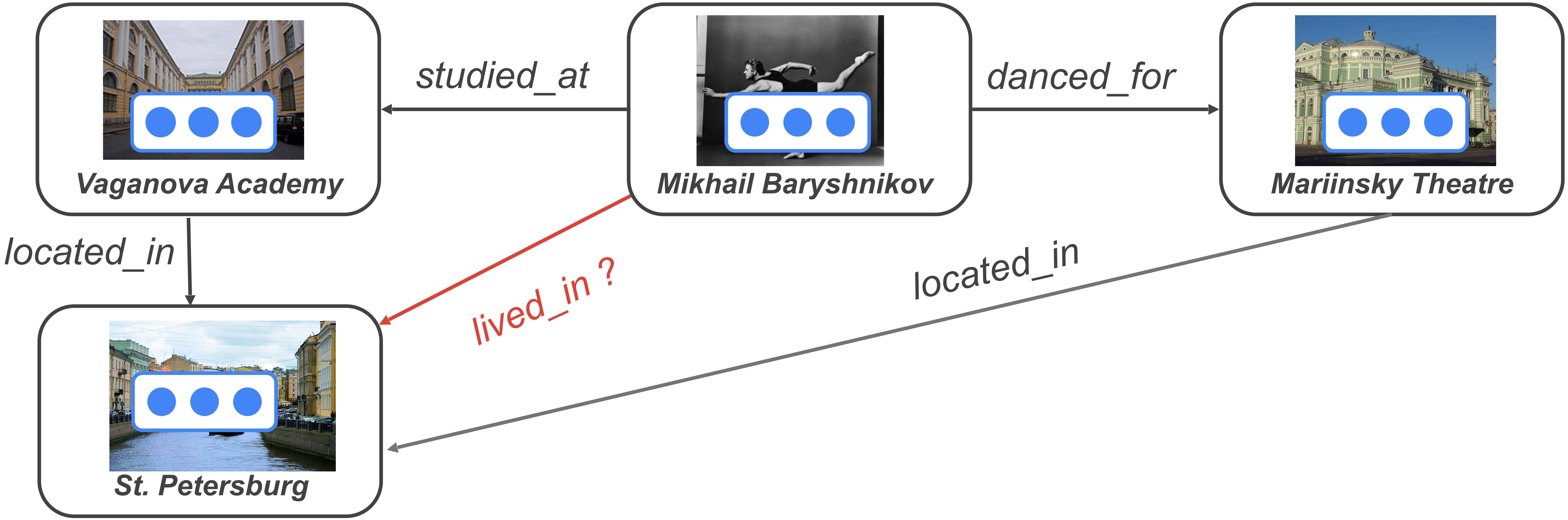
Link Prediction



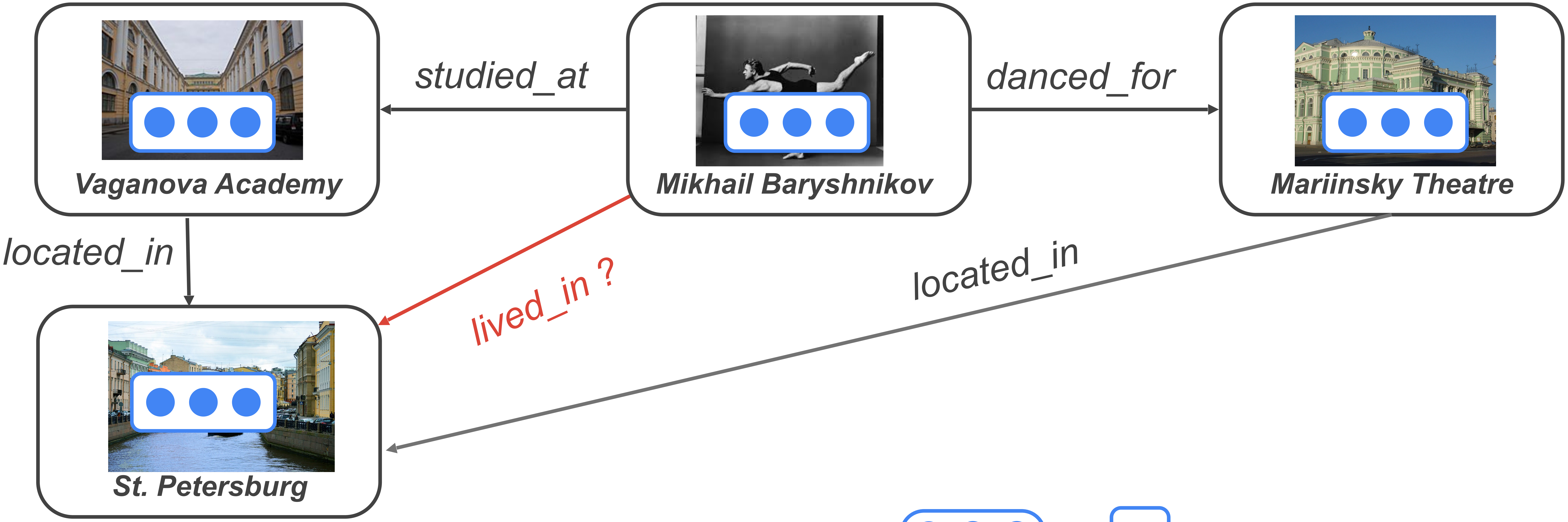
Link Prediction



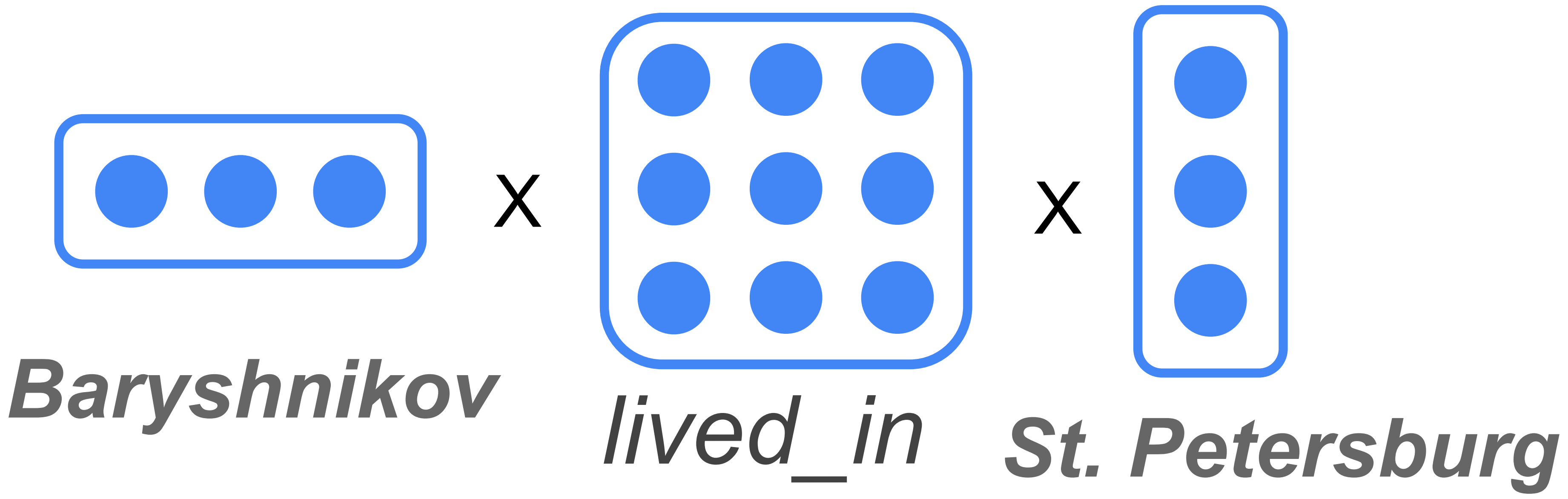
KB Factorization



KB Factorization

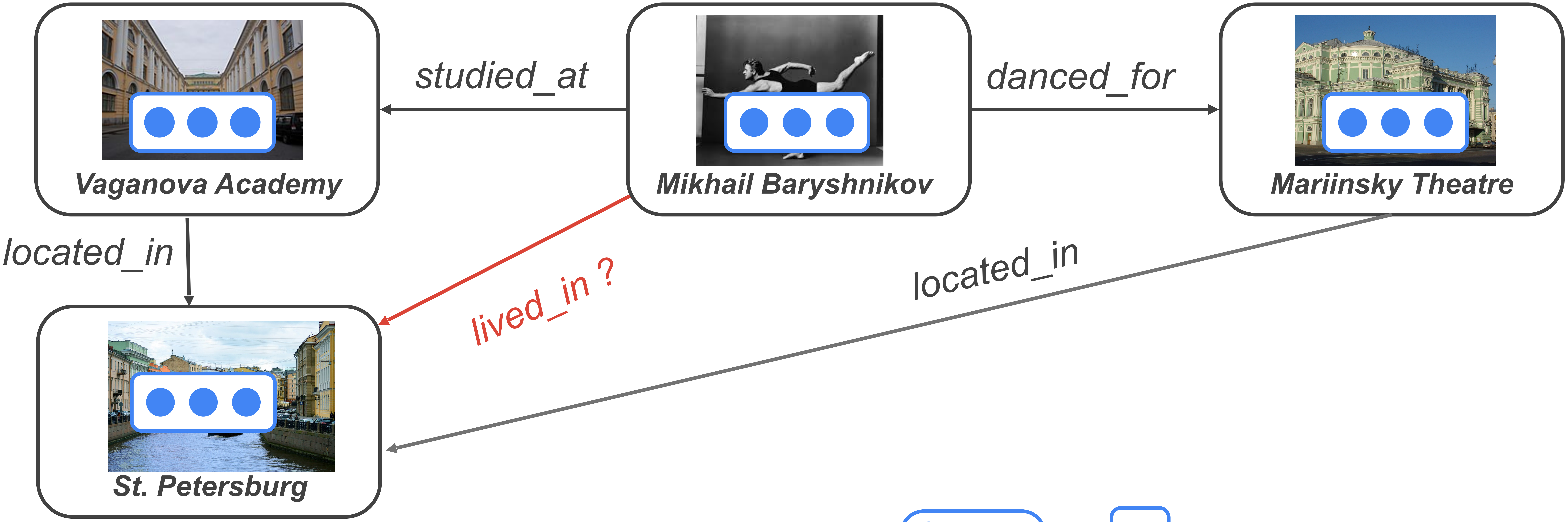


A scoring function is used to predict whether a relation holds:

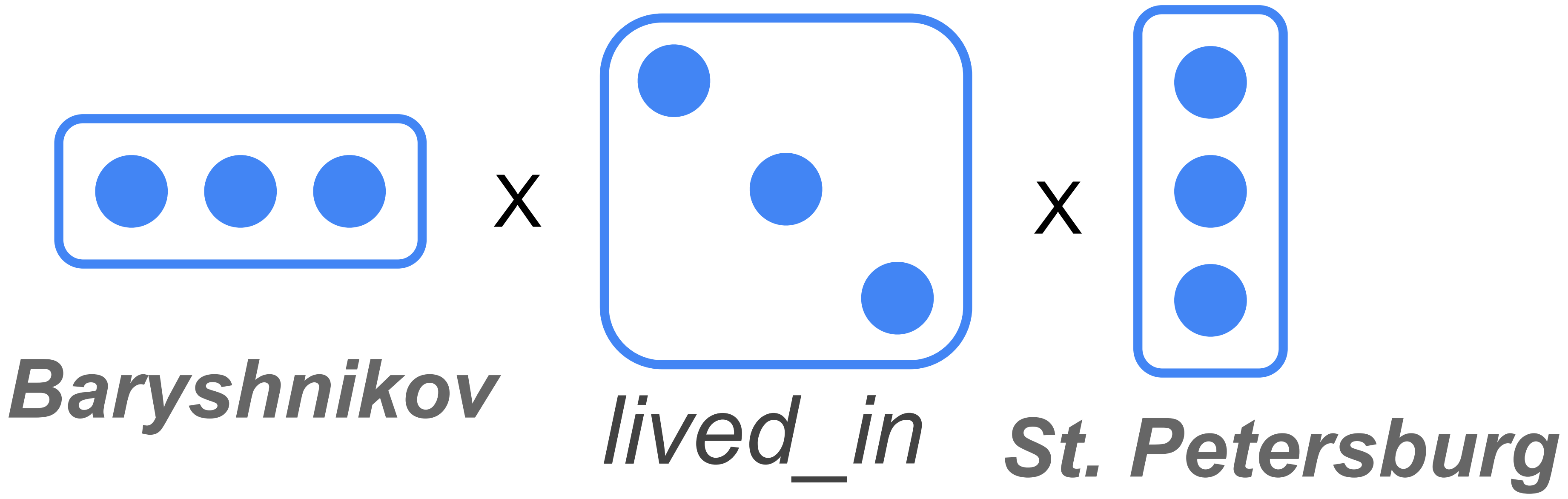


RESCAL
(Nickel et al., 2011)

KB Factorization

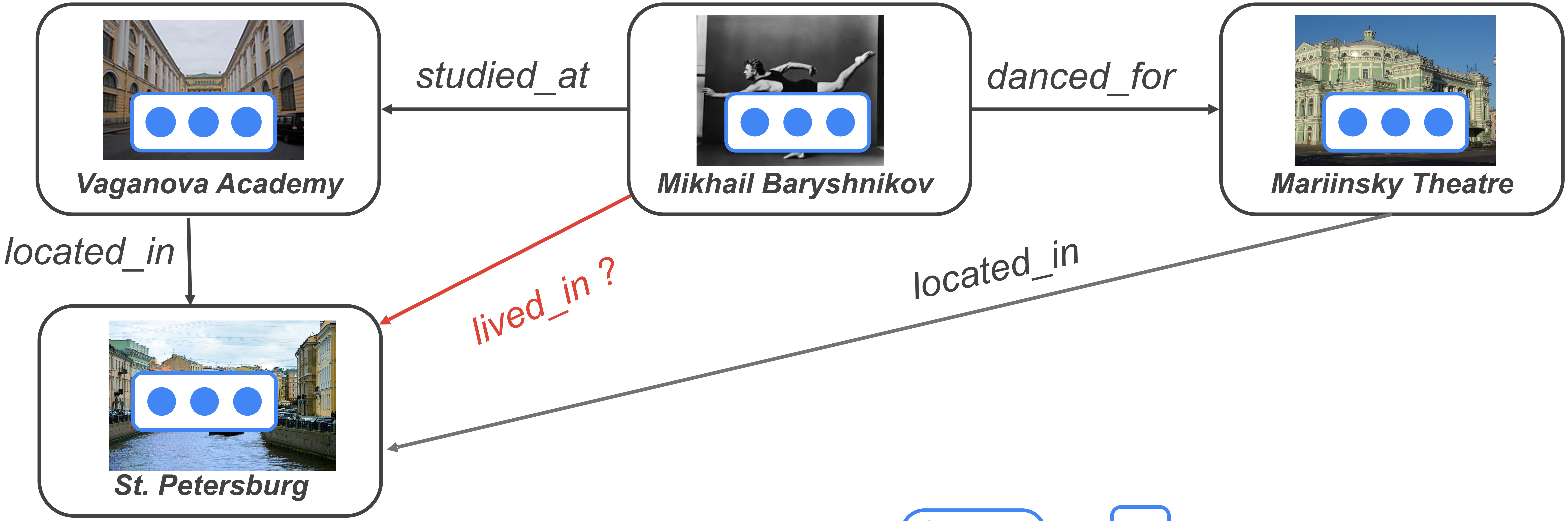


A scoring function is used to predict whether a relation holds:

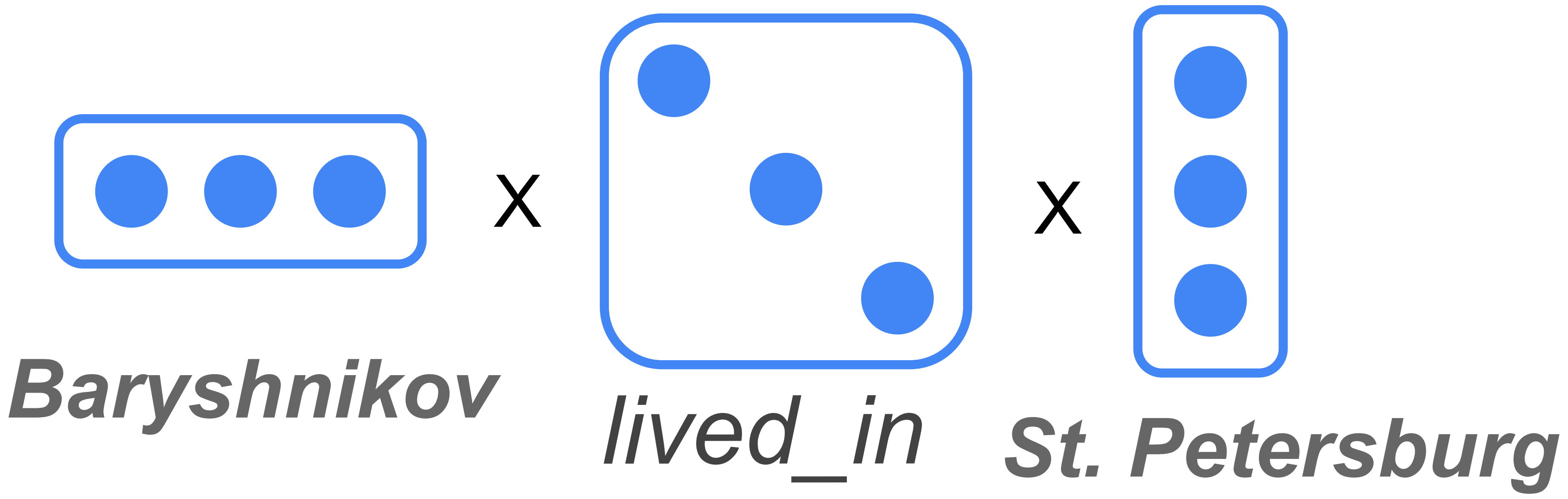


DistMult
(Yang et al., 2014)

KB Factorization



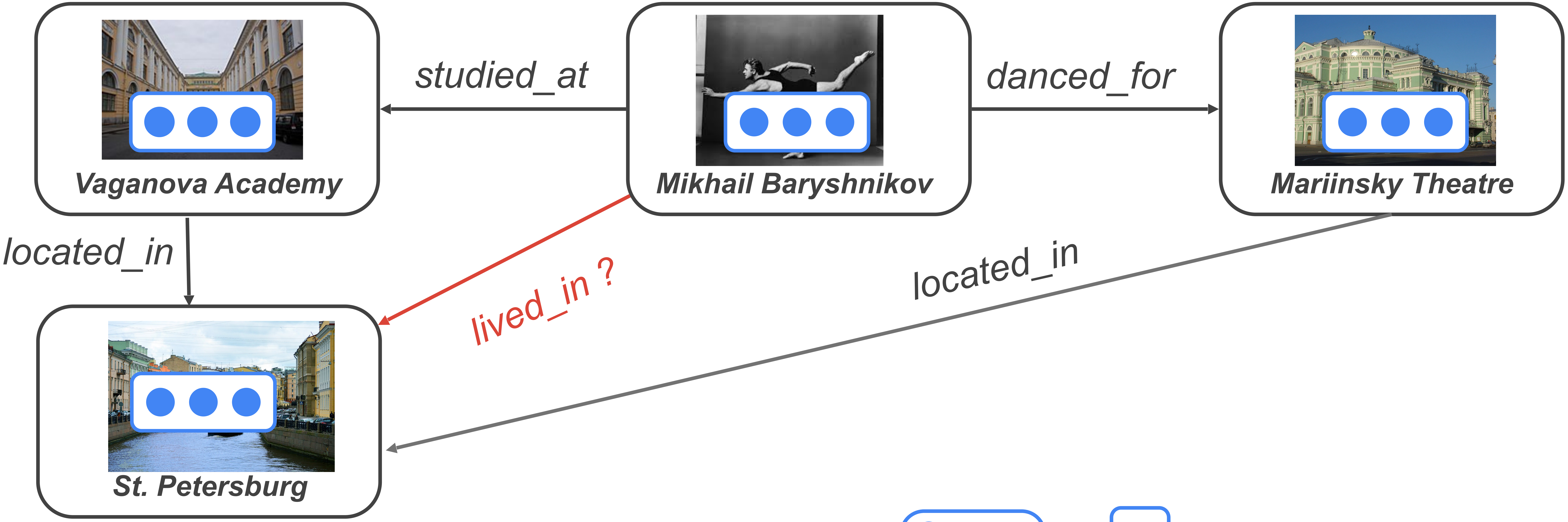
A scoring function is used to predict whether a relation holds:



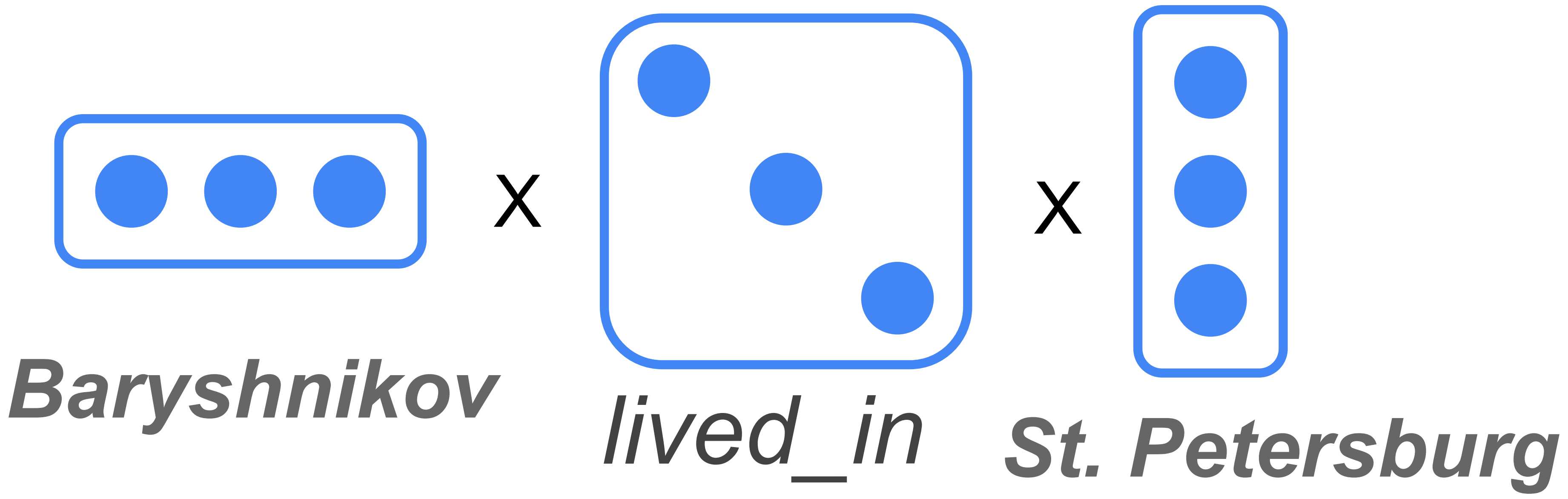
DistMult
(Yang et al., 2014)

The loss: score training triples higher than random (unobserved) ones

Relational GCNs



A scoring function is used to predict whether a relation holds:

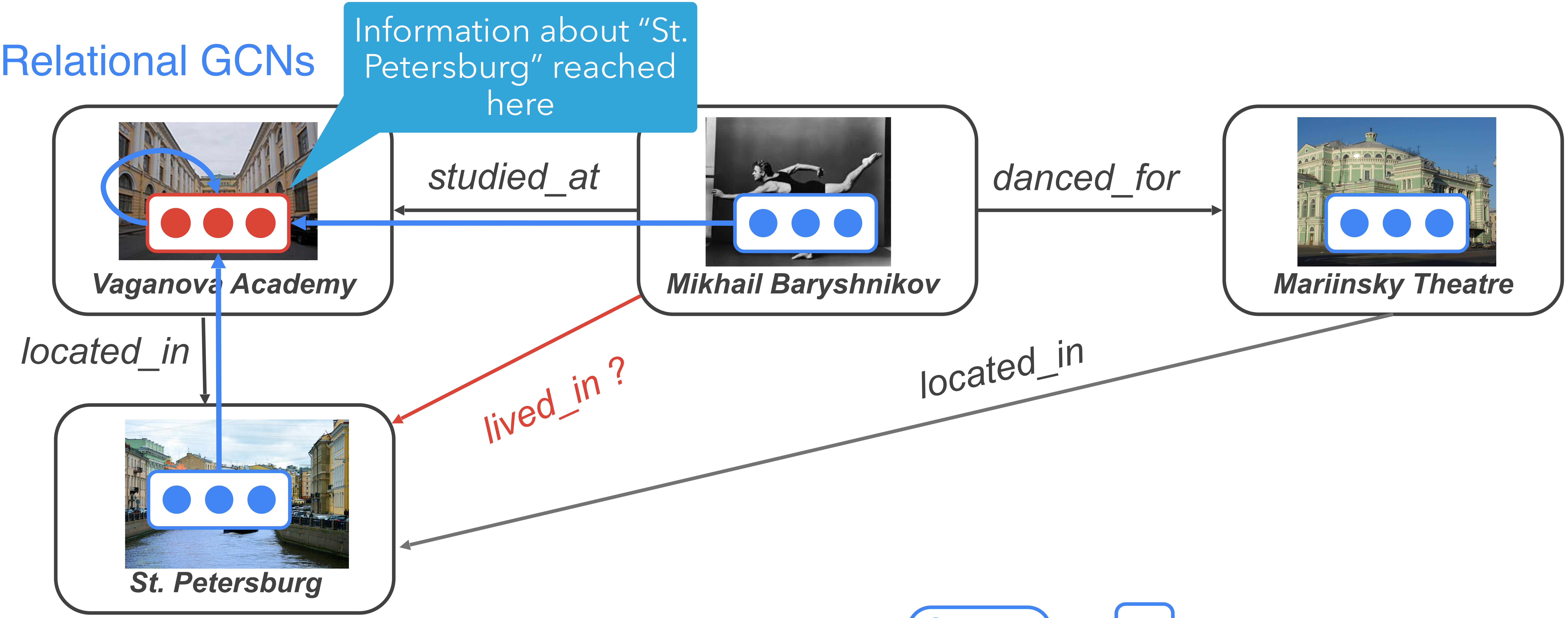


DistMult
(Yang et al., 2014)

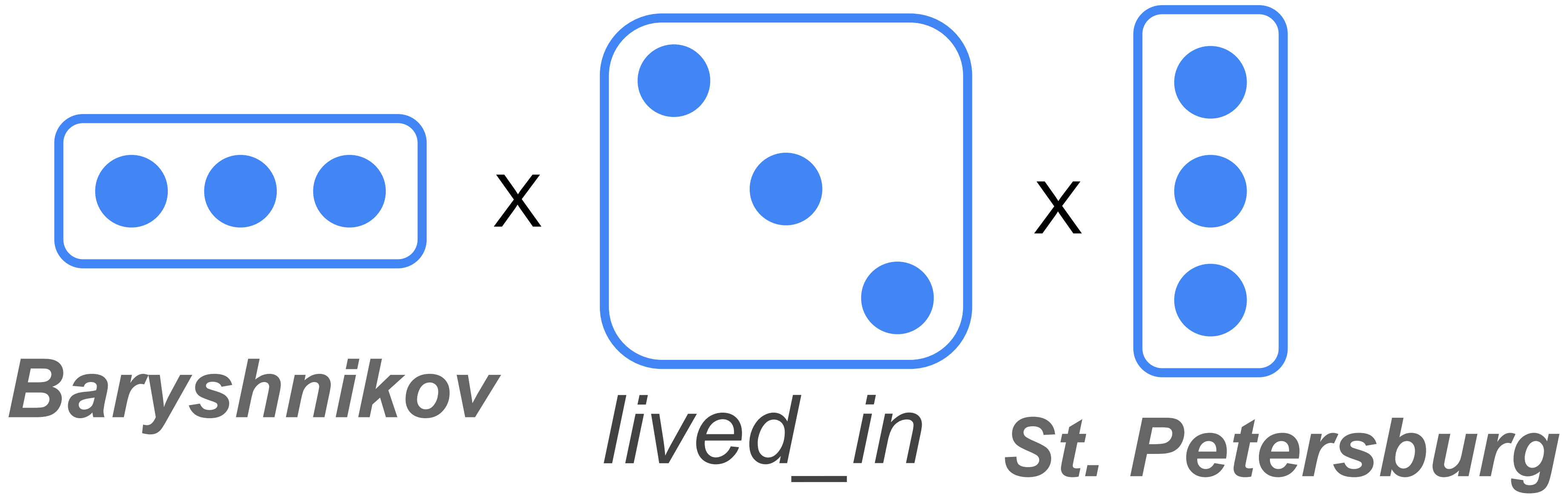
Use the same scoring function but with GNN node representations rather than parameter vectors

Schlichtkrull et al., 2017

Relational GCNs



A scoring function is used to predict whether a relation holds:

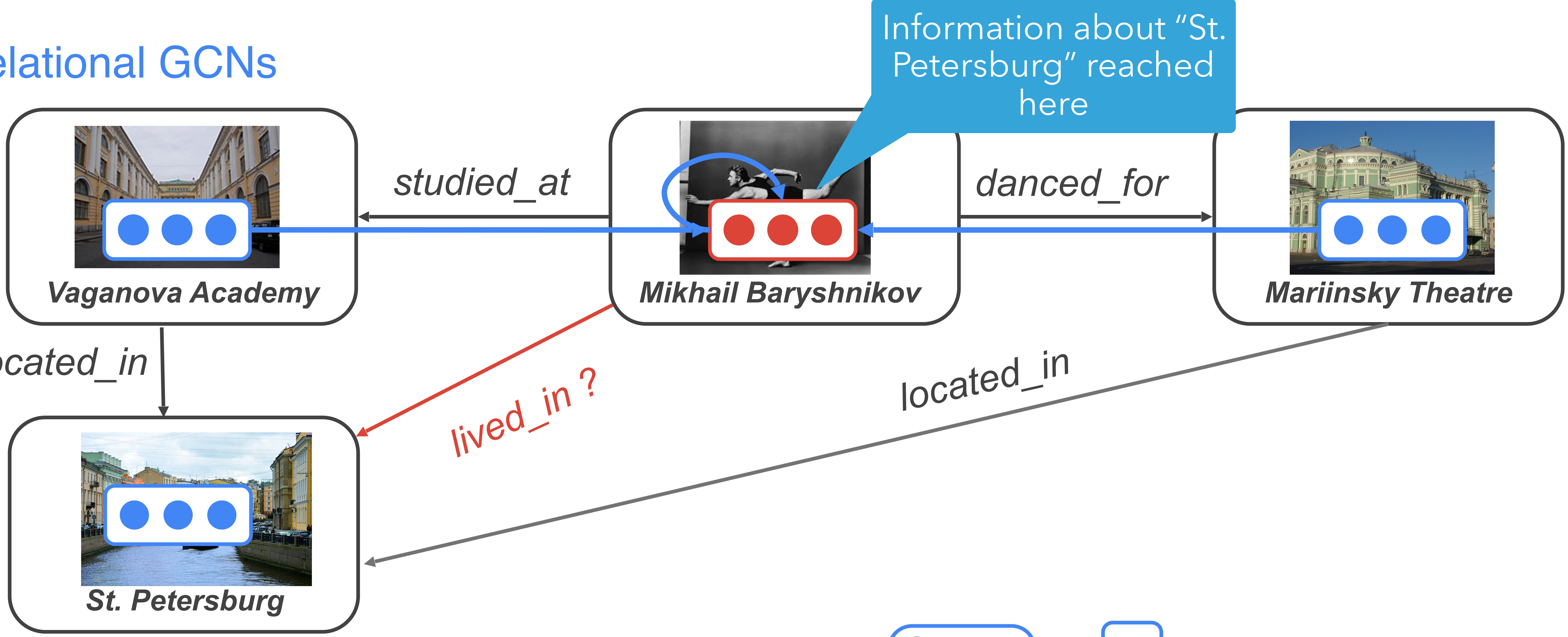


DistMult
(Yang et al., 2014)

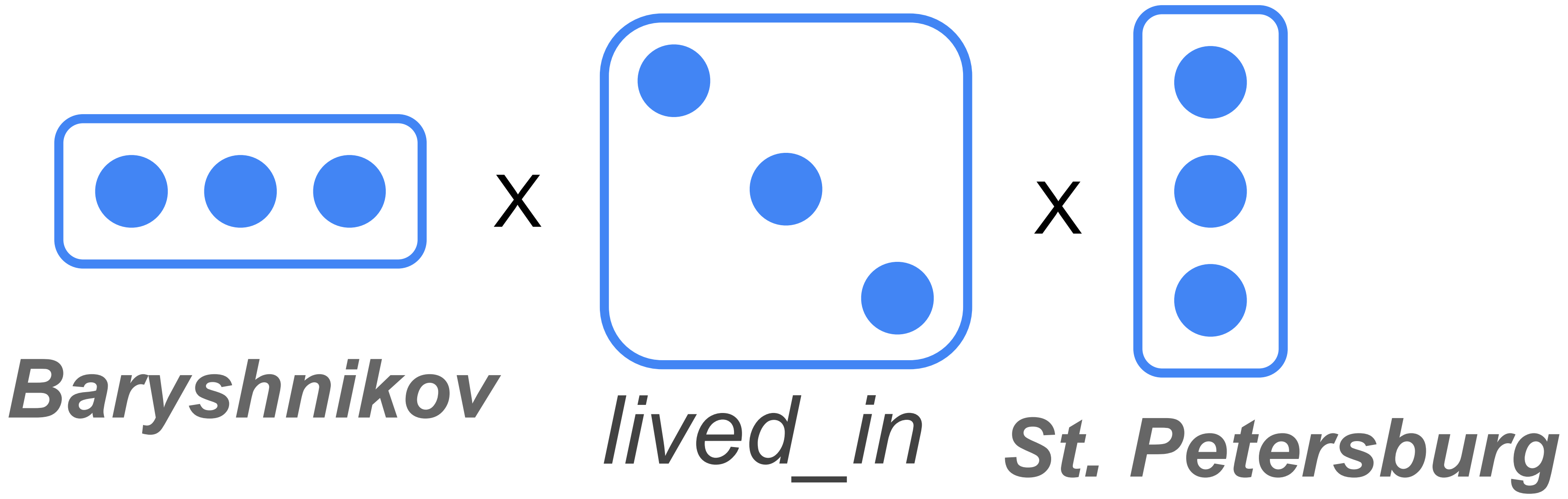
Use the same scoring function but with GNN node representations rather than parameter vectors

Schlichtkrull et al., 2017

Relational GCNs



A scoring function is used to predict whether a relation holds:

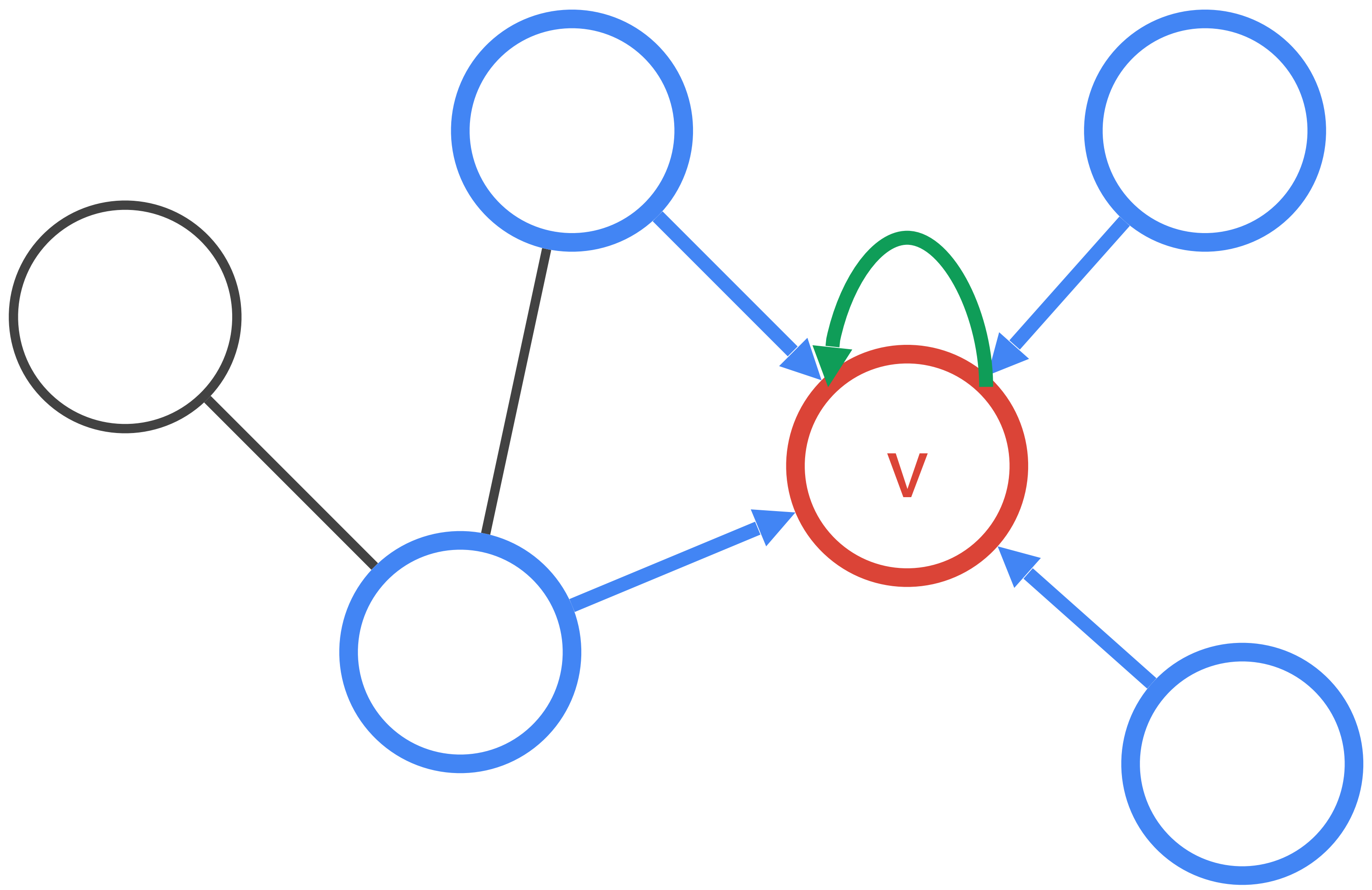


DistMult
(Yang et al., 2014)

Interestingly this is only true with non-anonymous nodes! (note: we learn node embeddings)

Schlichtkrull et al., 2017

Relational GCN



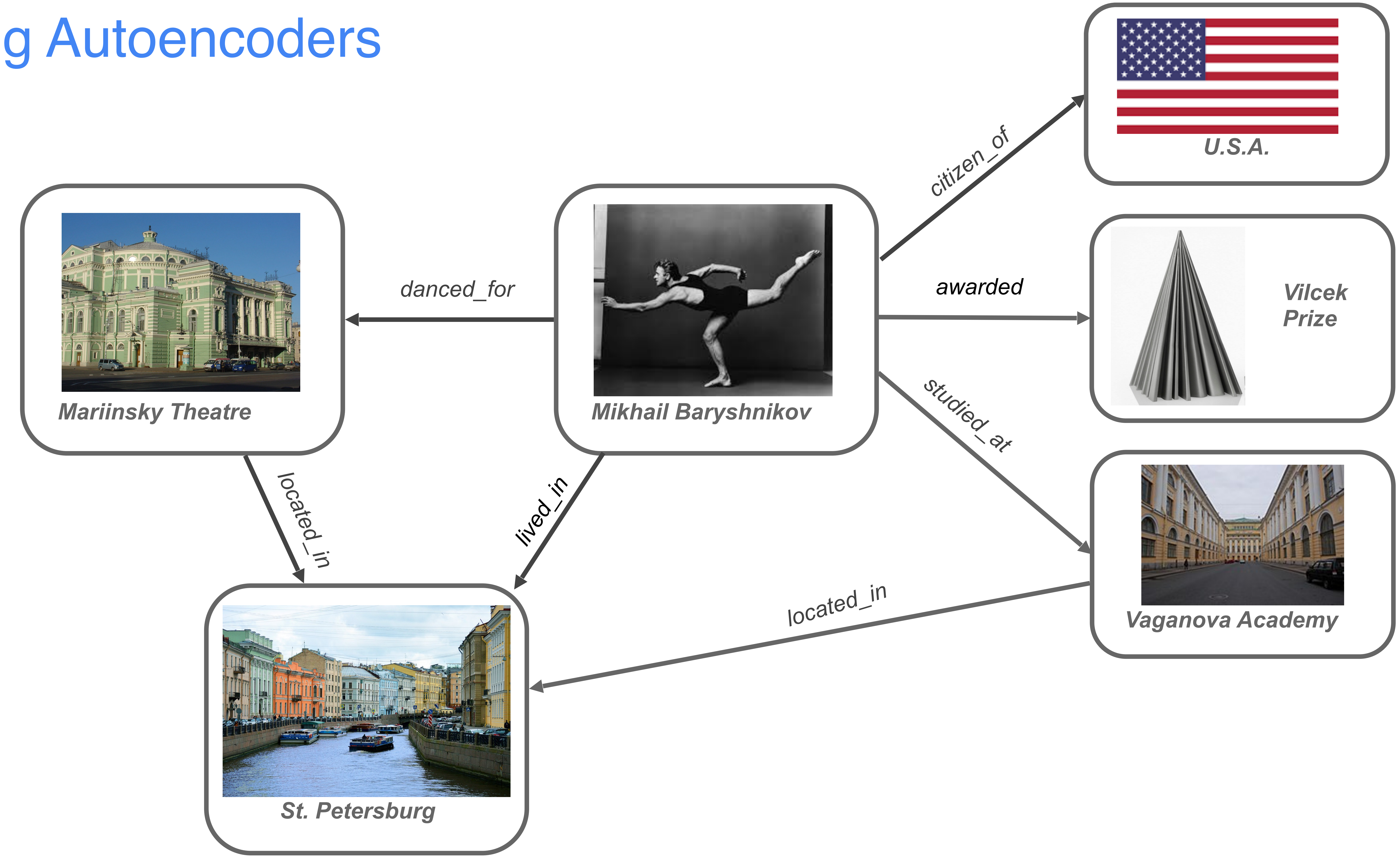
$$\mathbf{h}_v = ReLU(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)} W_{r(u,v)}\mathbf{h_u})$$

In practice, there are too many relations in realistic KBs, we cannot use full rank matrices W_r

More compact parameterisations are used in practice

Schlichtkrull et al., 2017

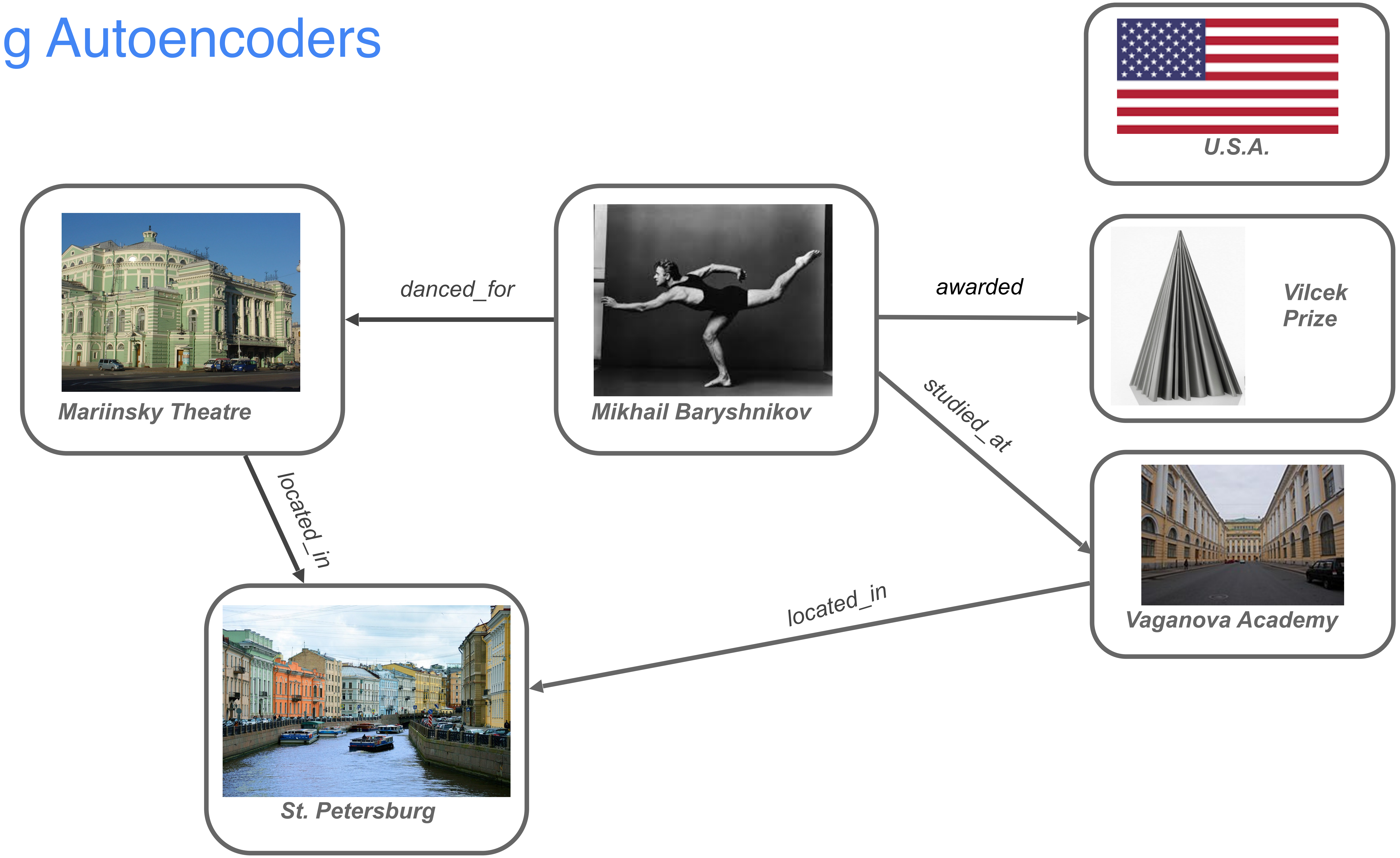
GCN Denoising Autoencoders



Take the training graph

Schlichtkrull et al (2017)

GCN Denoising Autoencoders

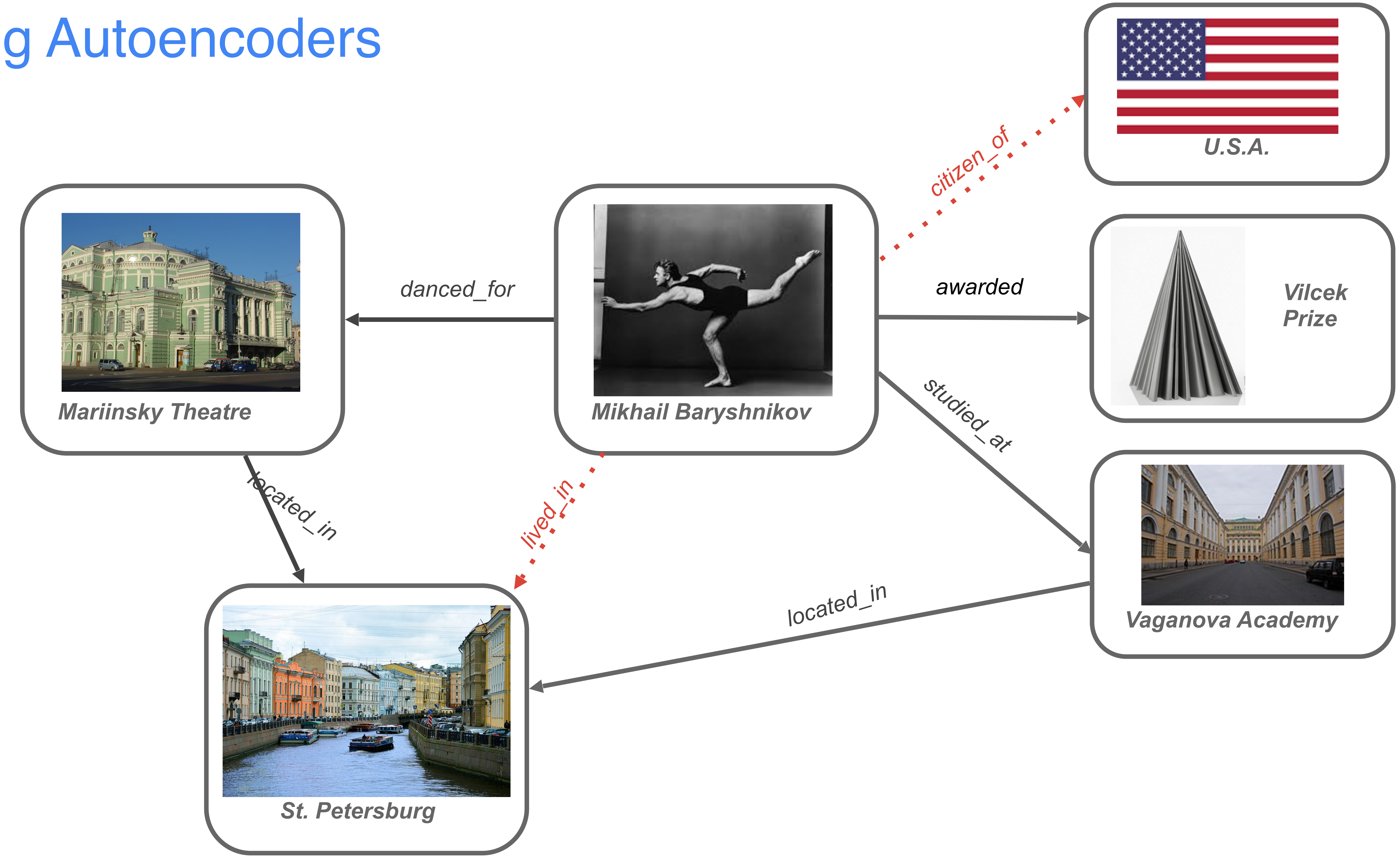


Produce a noisy version: drop some random edges

Use this graph for encoding nodes with GNN

Schlichtkrull et al., 2017

GCN Denoising Autoencoders

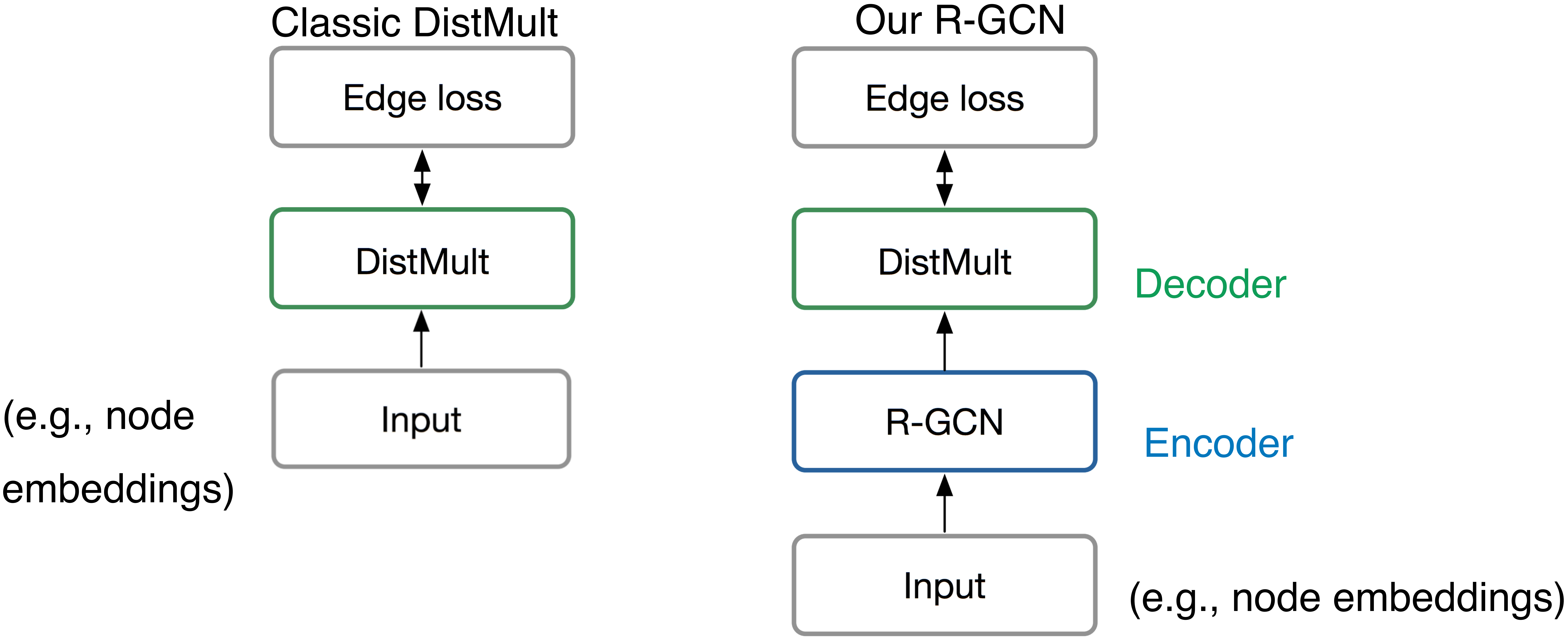


Force the model to reconstruct the original graph (including dropped edges)

(a ranking loss on edges)

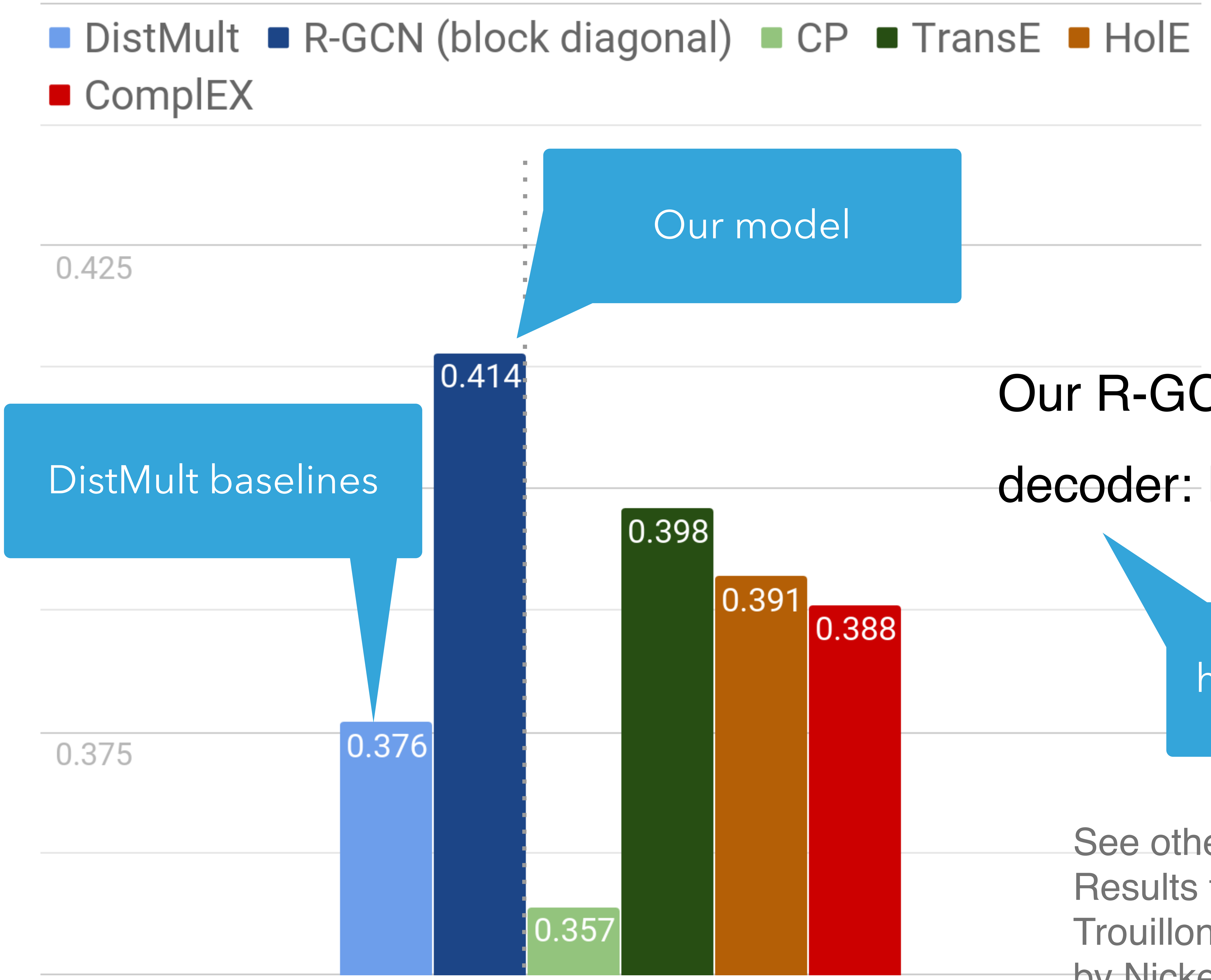
Schlichtkrull et al., 2017

Training



Schlichtkrull et al., 2017

Results on FB15k-237 (hits@10)

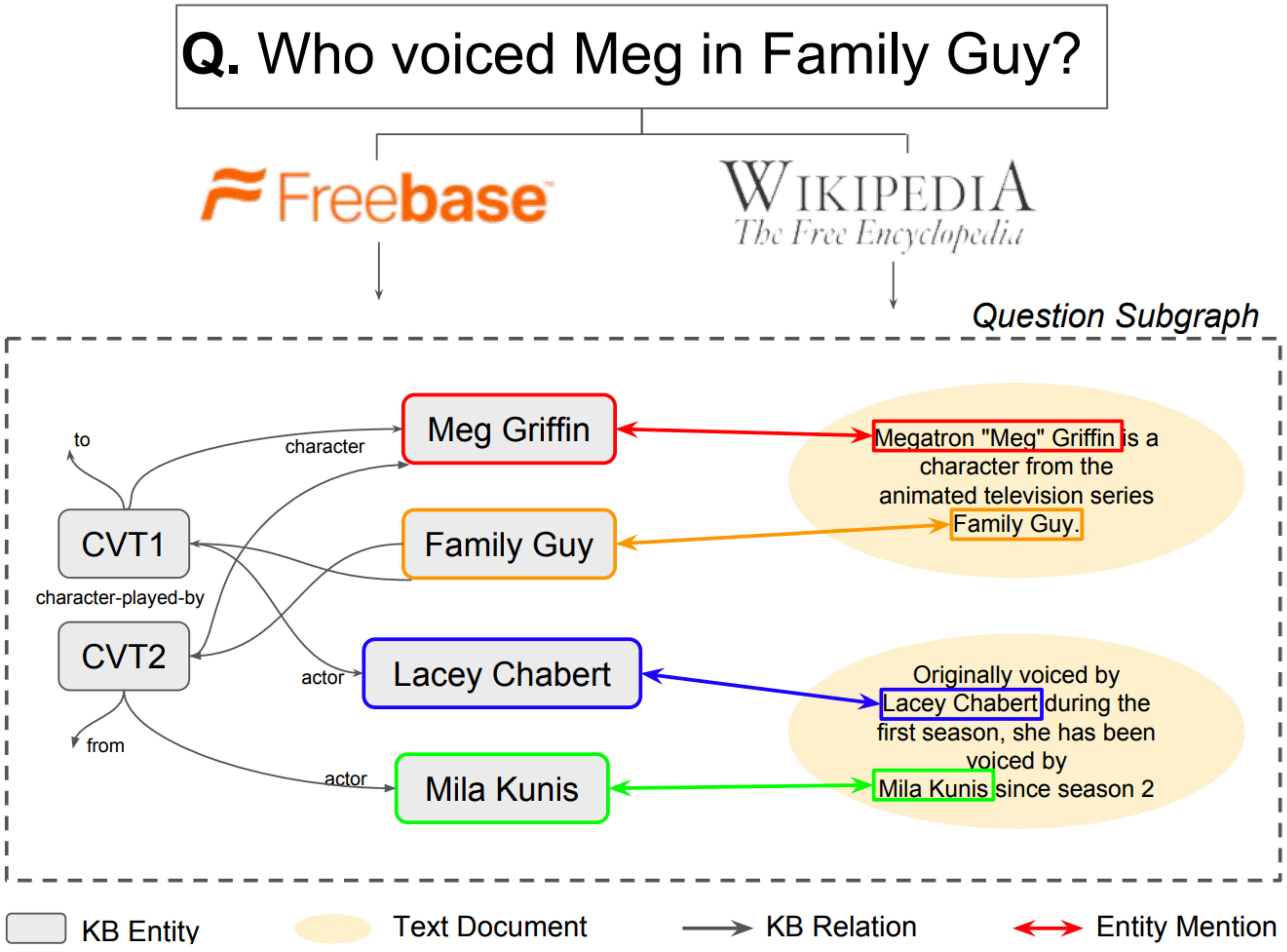


Our R-GCN relies on DistMult in the decoder: DistMult is its natural baseline

In 2020, we would have used Tucker or ConvE

See other results and metrics in the paper. Results for ComplEX, TransE and HolE from code o Trouillon et al. (2016). Results for HolE using code by Nickel et al. (2015)

Question Answering with Knowledge Bases and Text



Picture from. Sun∗, Dhingra∗, Zaheer, Mazaitis, Salakhutdinov and Cohen (EMNLP 2018)
Similar work by Sorokin and Gurevich (COLING 2018)

Summary for this section

- Link prediction task can be tackled with GNNs
- GNNs is a natural tool for ‘reasoning’ on top of knowledge bases and text
- Open problems
  - Inductive models (i.e. applying to new entities)
  - Training regimes
  - No success story yet on KB+Text for question answering

“Advanced” topics

In this section

- GNNs over induced graphs
- Extracting explanations from GNNs

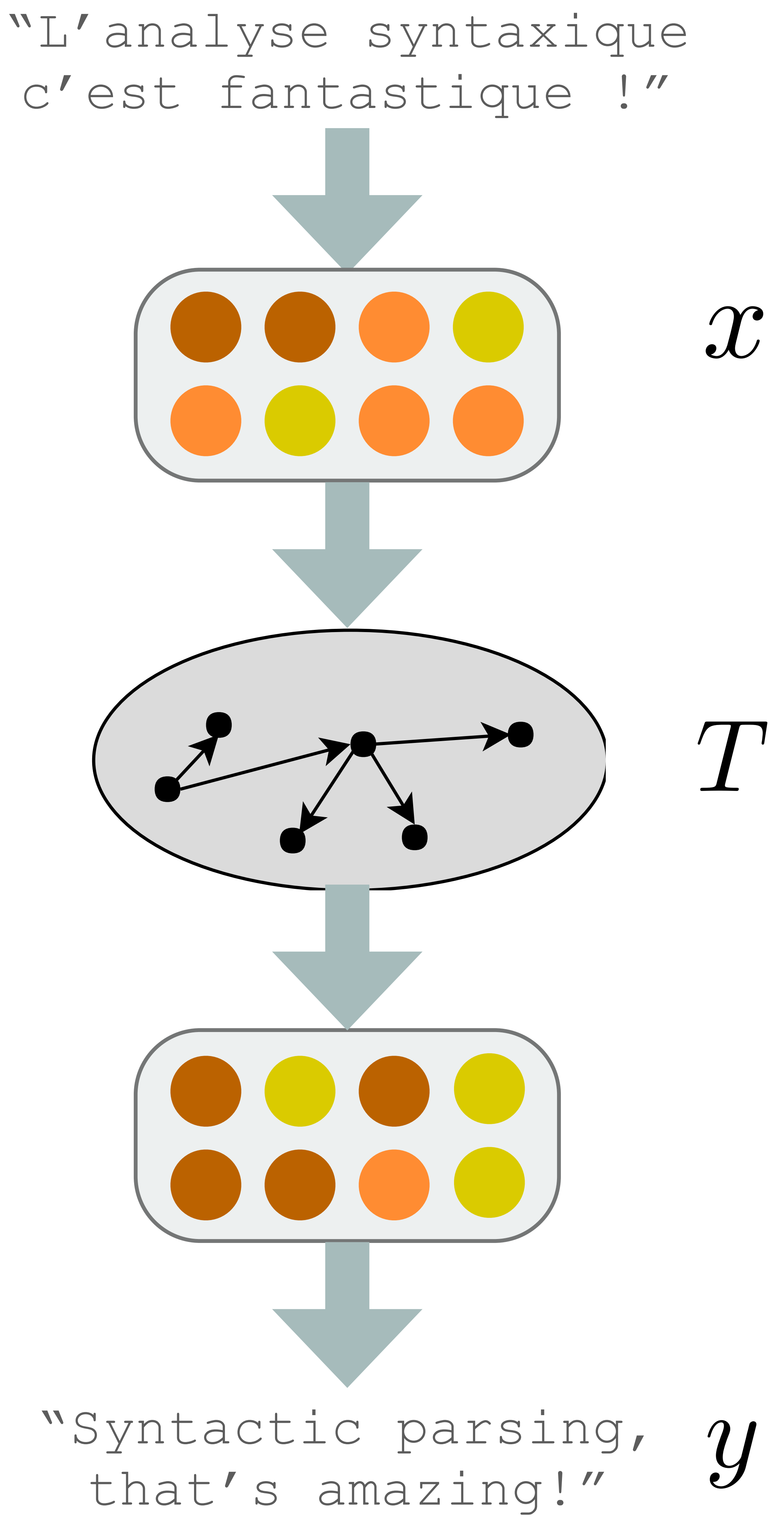
Encoding linguistic structure

- In part 2, we discussed encoding ***detailed*** linguistic knowledge:
  - The structures may not be suitable to downstream applications
  - The quality of linguistic tools (e.g., parsers) may be a limiting factor

Structured latent variable modeling

- What about encoding more general inductive biases? E.g.
  - "text structure can be approximated with trees"
  - "co-reference can be represented as sets of chains"
  - "alignment between two sequences should be bijective / matching"

and induce the structure as part of end-to-end learning



Probabilistic interpretation

$$p(y|x) = \mathbb{E}_{p(T|x)} p(y|x, T)$$

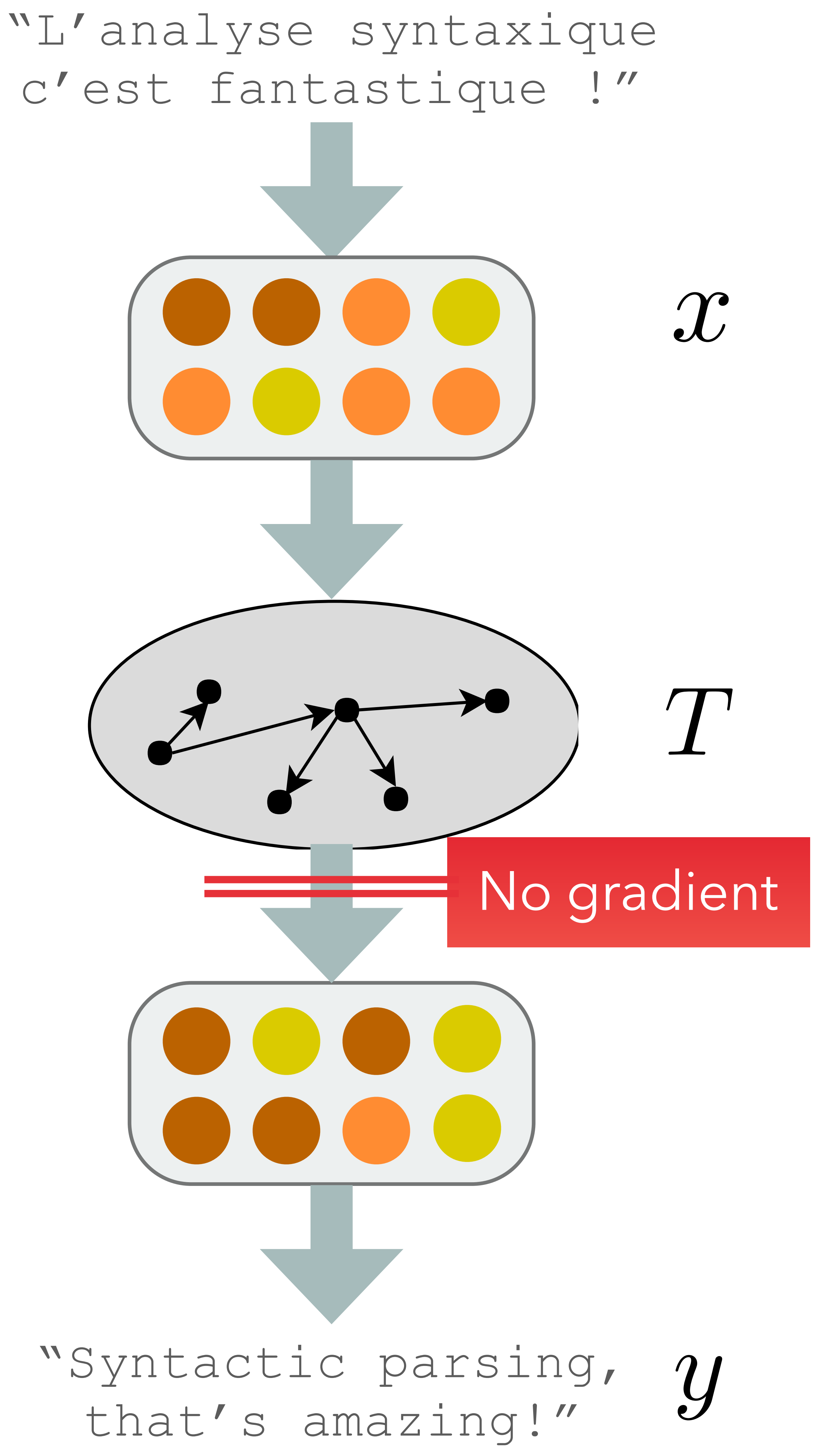
Intractable summation

Picture by Caio Corro

Previous work on structured layers in deep generative models

- REINFORCE: non-differentiable

  (Yogatama et al., ICLR 2017; Williams et al., NAACL 2018, Havrylov et al., NAACL 2019)

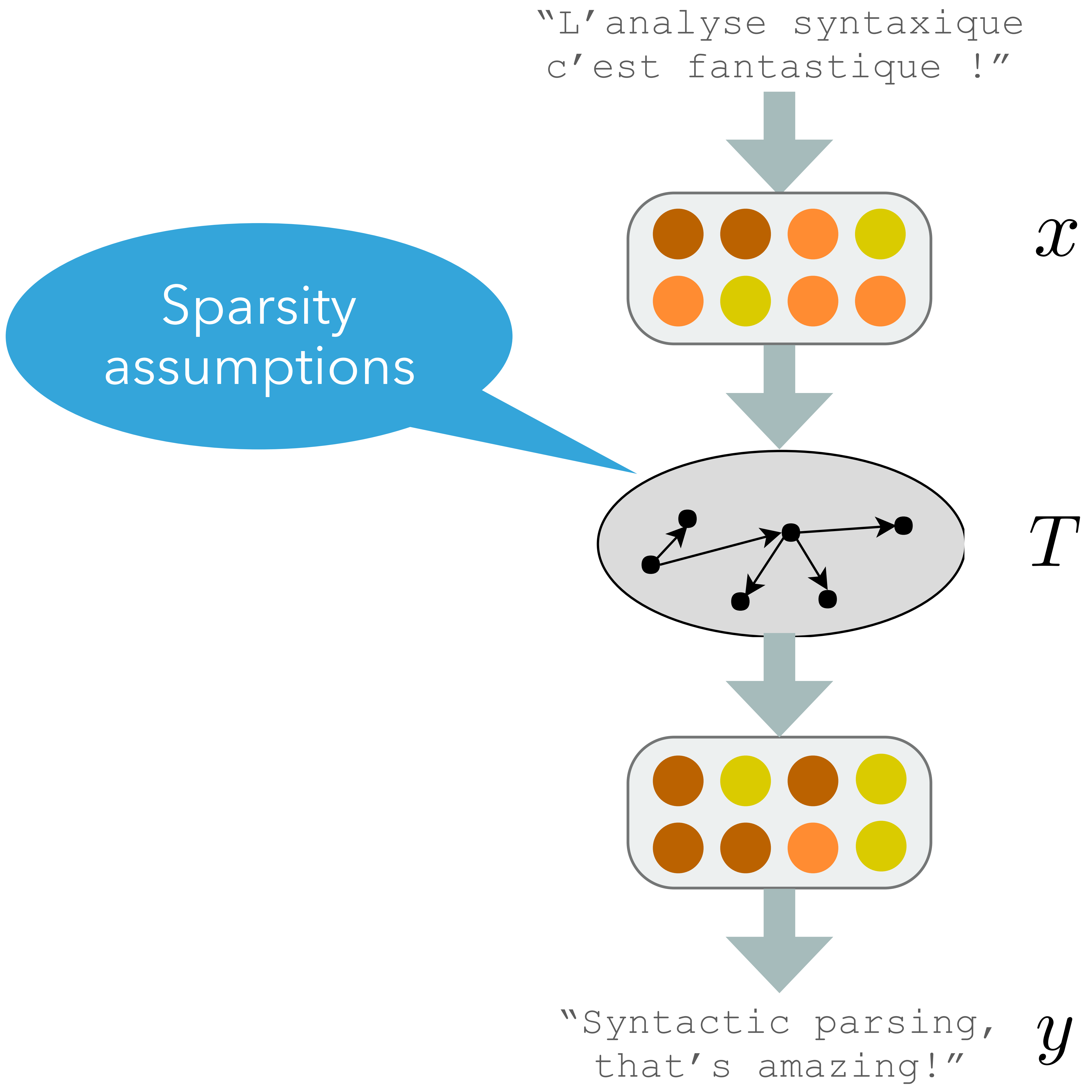


Picture by Caio Corro

Previous work on structured layers in deep generative models

- REINFORCE: non-differentiable
- SparseMAP: exact marginalisation

(Niculae, Martins, Blondel and Cardie, EMNLP 2018)

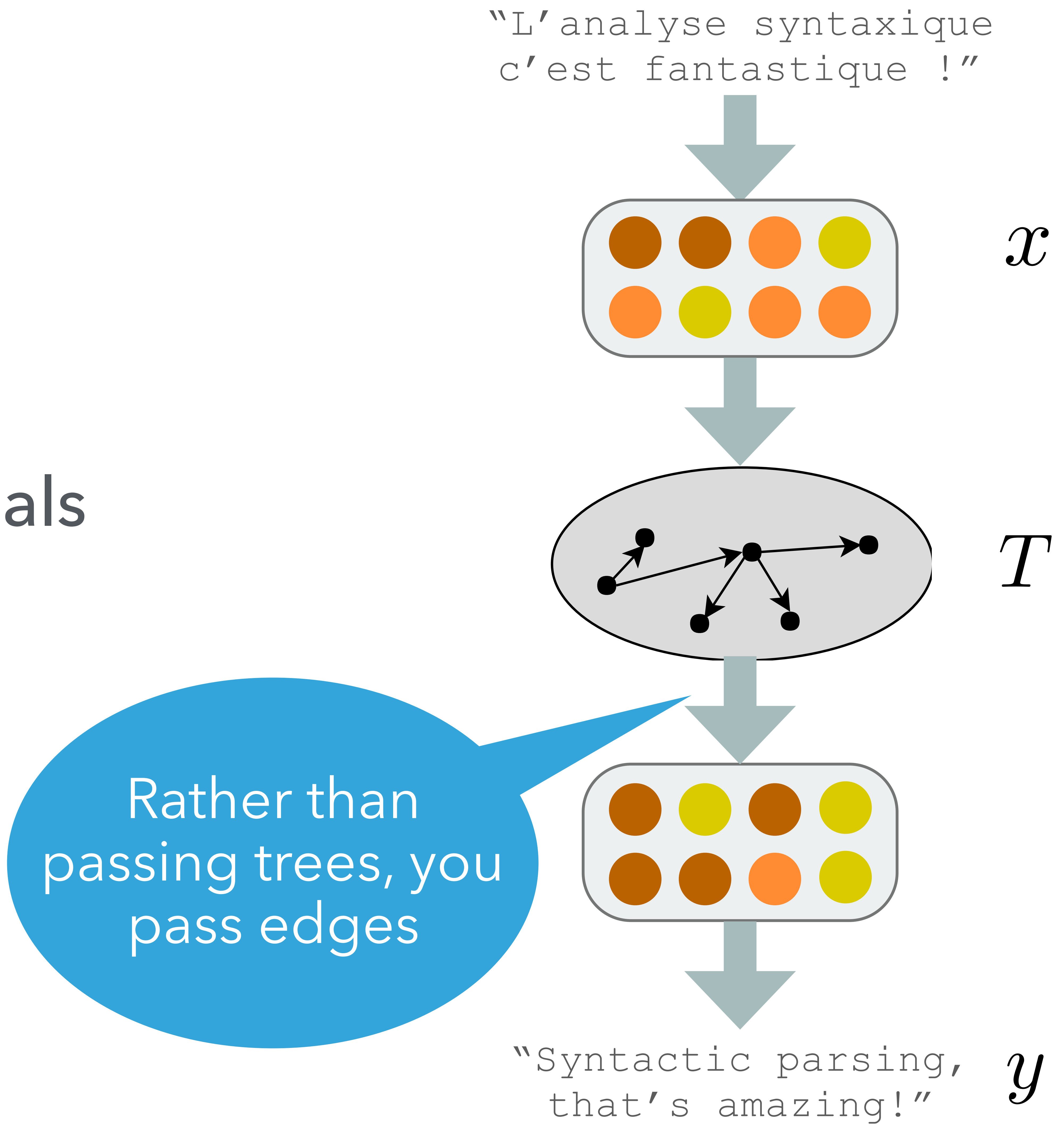


Picture by Caio Corro

Previous work on structured layers in deep generative models

- REINFORCE: non-differentiable
- SparseMAP: exact marginalisation
- Structured Attention: relies on edge marginals

(Kim et al., ICLR 2017; Liu and Lapata, TACL 2017)

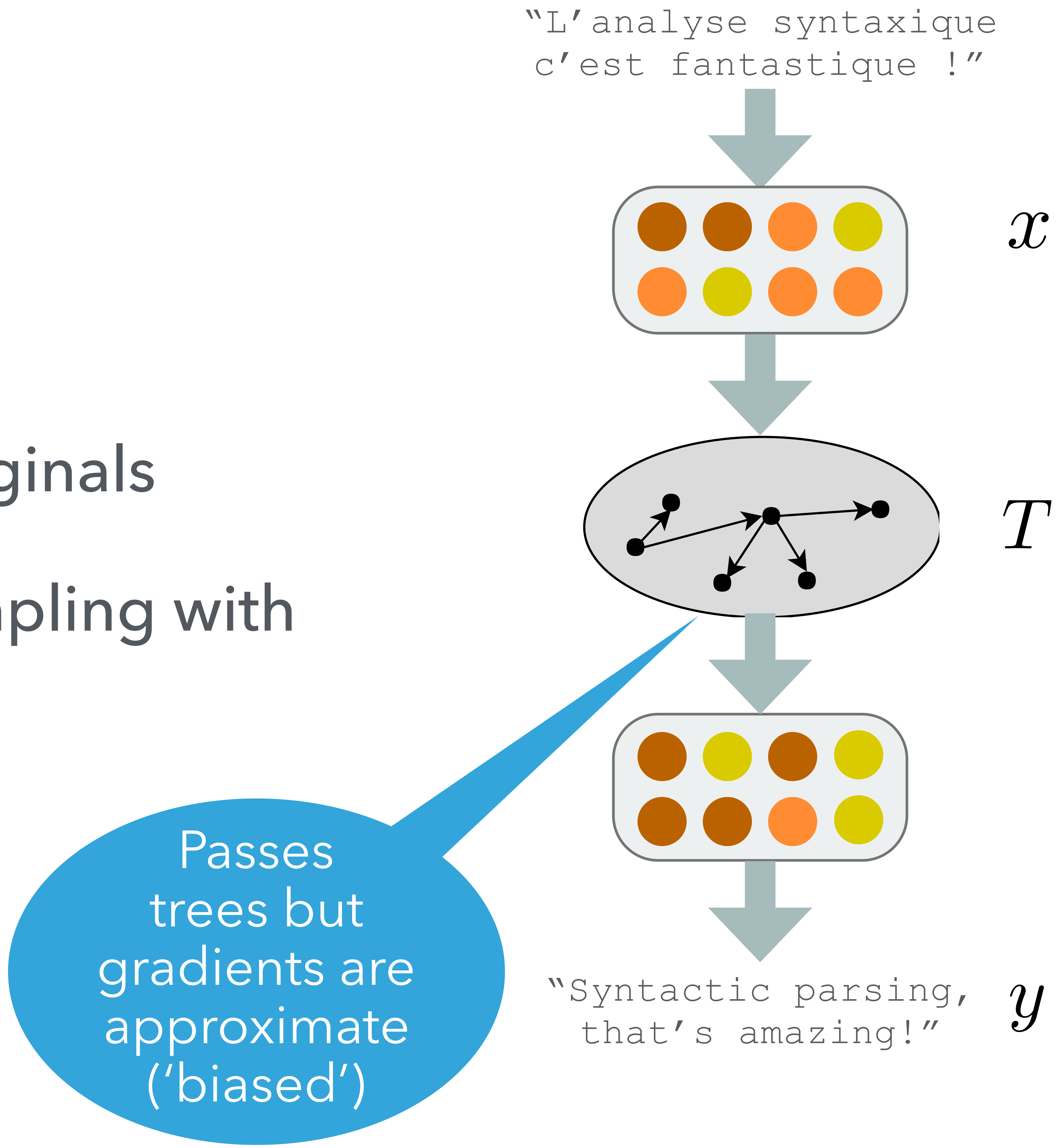


Picture by Caio Corro

Previous work on structured layers in deep generative models

- REINFORCE: non-differentiable
- SparseMAP: exact marginalisation
- Structured Attention: relies on edge marginals
- Perturb-and-Parse: approximate tree sampling with differential dynamic programming

Corro and Titov (ICLR, 2019; ACL 2019)

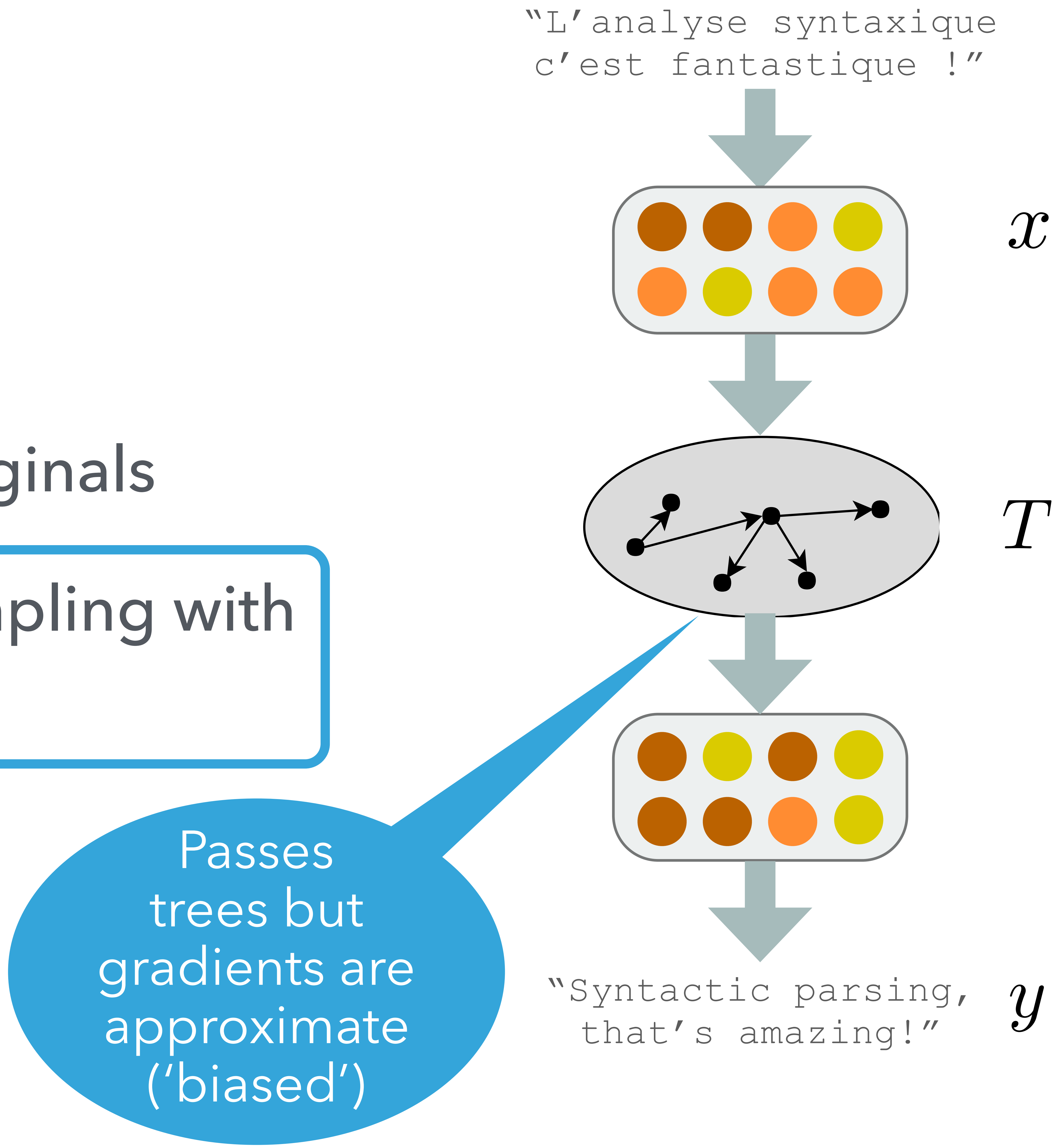


Picture by Caio Corro

Previous work on structured layers in deep generative models

- REINFORCE: non-differentiable
- SparseMAP: exact marginalisation
- Structured Attention: relies on edge marginals
- Perturb-and-Parse: approximate tree sampling with differential dynamic programming

Corro and Titov (ICLR, 2019; ACL 2019)



Picture by Caio Corro

Application: Natural Language Inference / Textual Entailment

Bowman et al. (EMNLP 2015),
Williams et al. (NAACL 2018)

SNLI / MultiNLI datasets

- Premise: A man walking under an umbrella
- Hypothesis: A stranger carrying an umbrella
- Entails, contradicts or neutral?

Corro and Titov (ACL 2019)

Application: Natural Language Inference / Textual Entailment

Corro and Titov (ACL 2019)

Application: Natural Language Inference / Textual Entailment

Corro and Titov (ACL 2019)

Application: Natural Language Inference / Textual Entailment

Ablation tests (MultiNLI)

Baseline	
No intra att.	68.5
Intra att.	67

Latent heads	
1 GCN	69
2 GCN	68.7

Latent trees	
1 GCN	71.9
2 GCN	73.2

Corro and Titov (ACL 2019)

Application: Natural Language Inference / Textual Entailment

Ablation tests (MultiNLI)

Baseline	
No intra att.	68.5
Intra att.	67
Latent heads	
1 GCN	69
2 GCN	68.7
Latent trees	
1 GCN	71.9
2 GCN	73.2

Substantial improvement from P&P and GCNs

Corro and Titov (ACL 2019)

Application: Natural Language Inference / Textual Entailment

Ablation tests (MultiNLI)

Baseline	
No intra att.	68.5
Intra att.	67
Latent heads	
1 GCN	69
2 GCN	68.7
Latent trees	
1 GCN	71.9
2 GCN	73.2

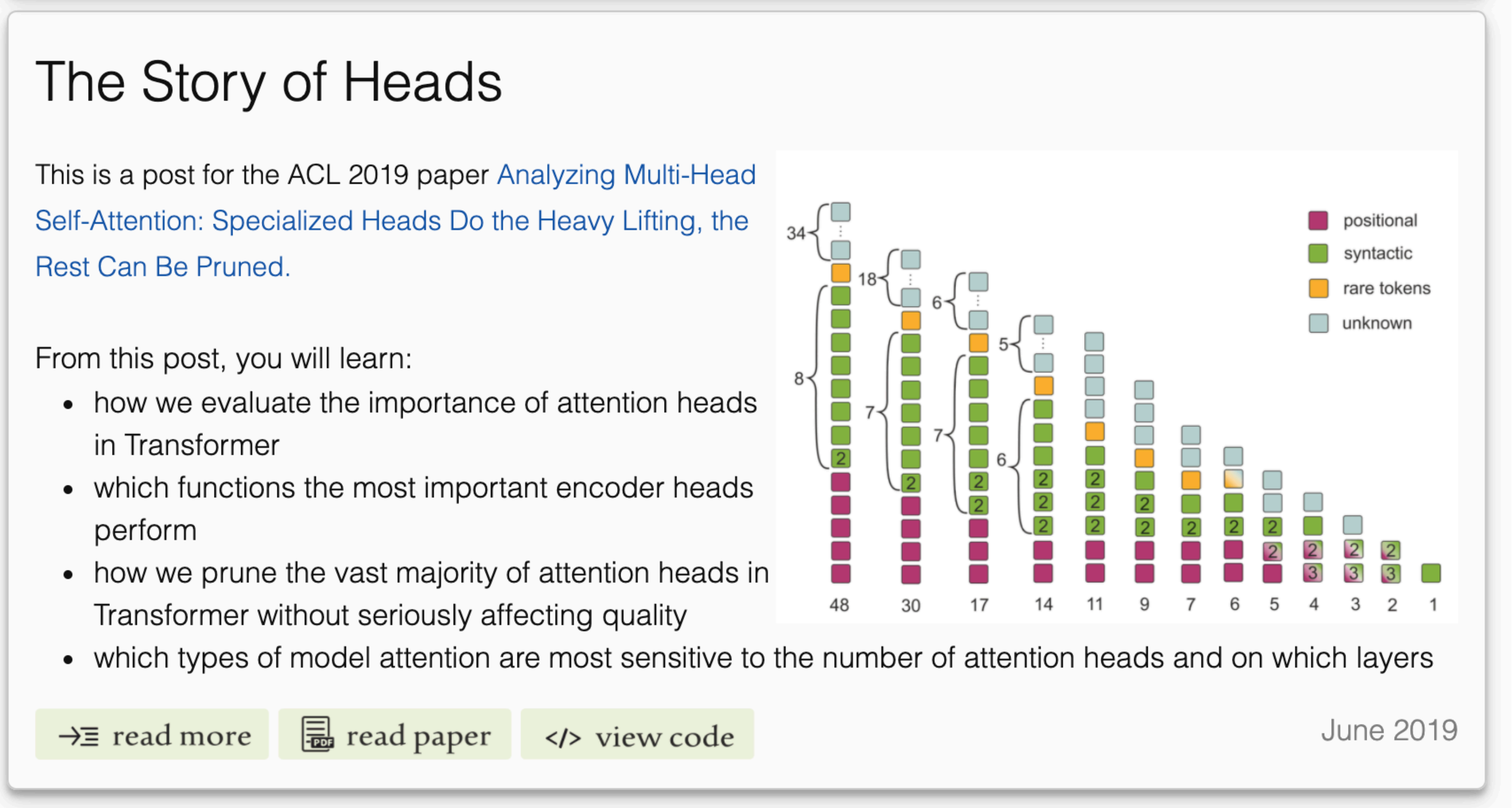
Constraining structures to be projective dependency trees helps

Corro and Titov (ACL 2019)

GNNs over latent structures - summary

- If you have effective linguistic tools, they can provide a useful linguistic bias
- If not, but can make assumptions about the structure, it can still help
- Deep NLP models (e.g., Transformer) already induce graphs which are to certain degree interpretable, e.g.,

Voita, Talbot, Moiseev, Sennrich, and Titov, ACL 2019



The Story of Heads

This is a post for the ACL 2019 paper Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned.

From this post, you will learn:

- how we evaluate the importance of attention heads in Transformer
- which functions the most important encoder heads perform
- how we prune the vast majority of attention heads in Transformer without seriously affecting quality
- which types of model attention are most sensitive to the number of attention heads and on which layers

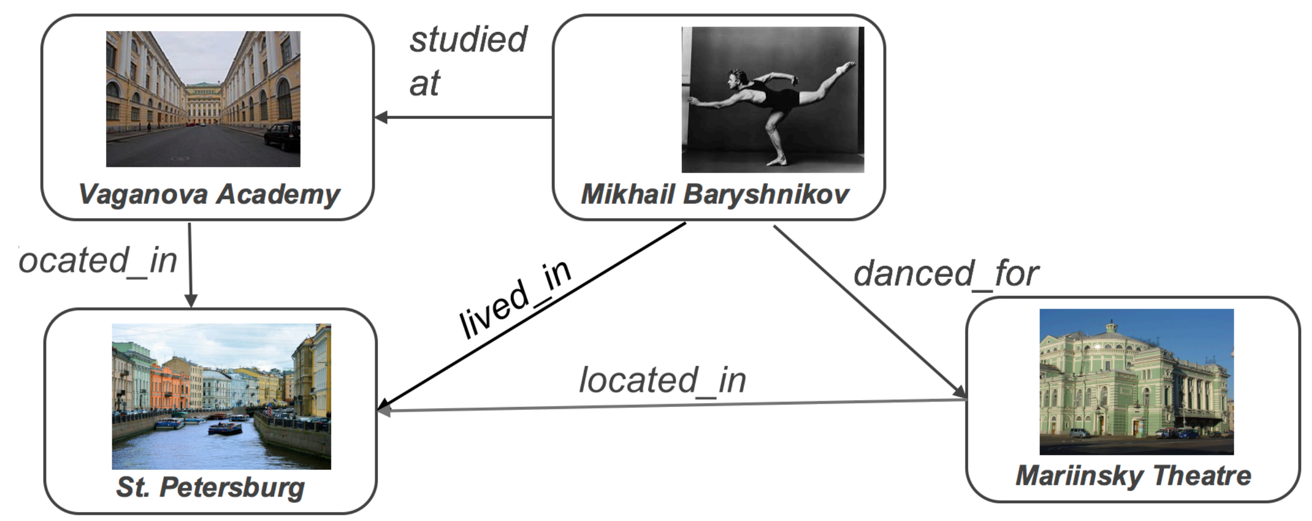
June 2019

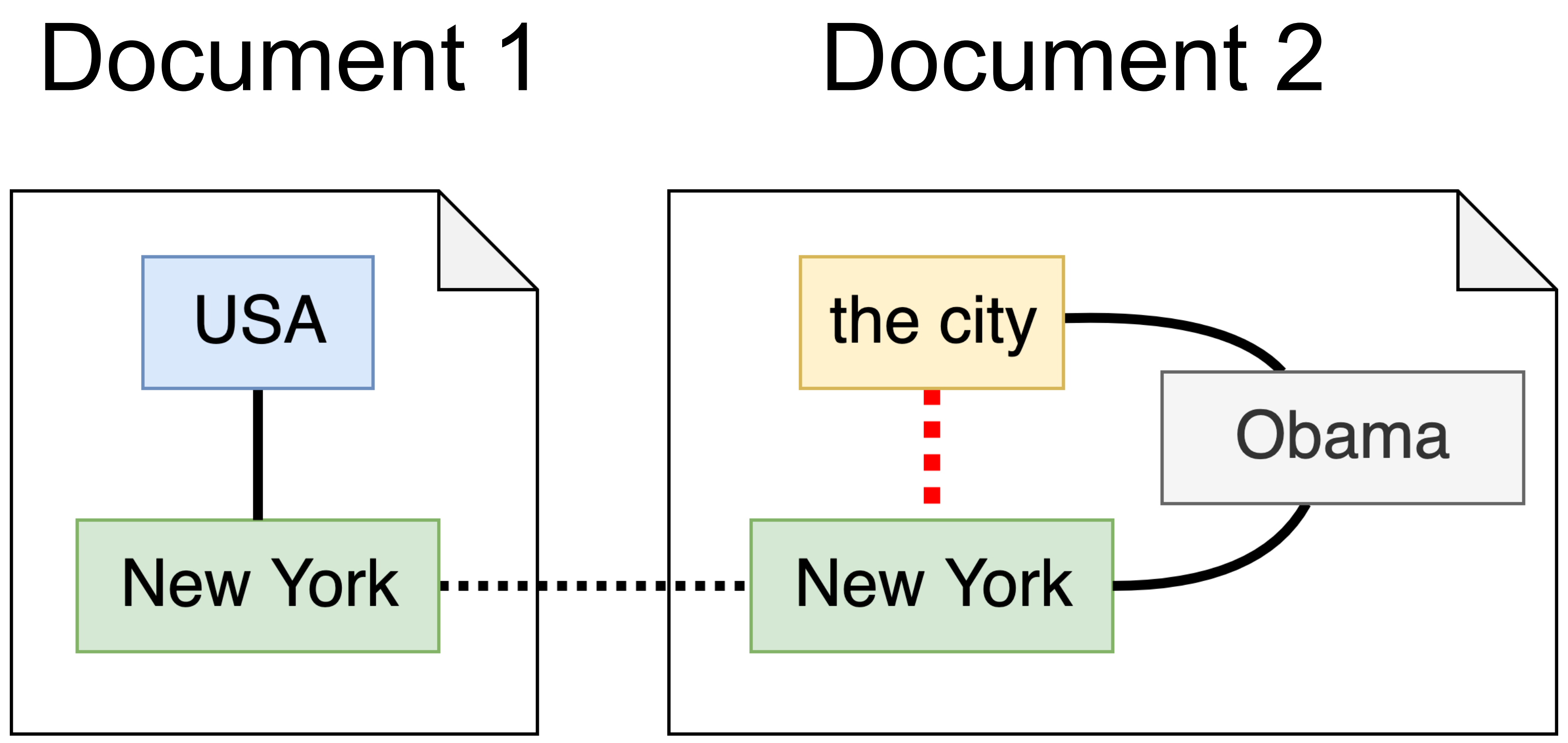
In this section

- GNNs over induced graphs
- Extracting explanations from GNNs

Intepretability of GNNs

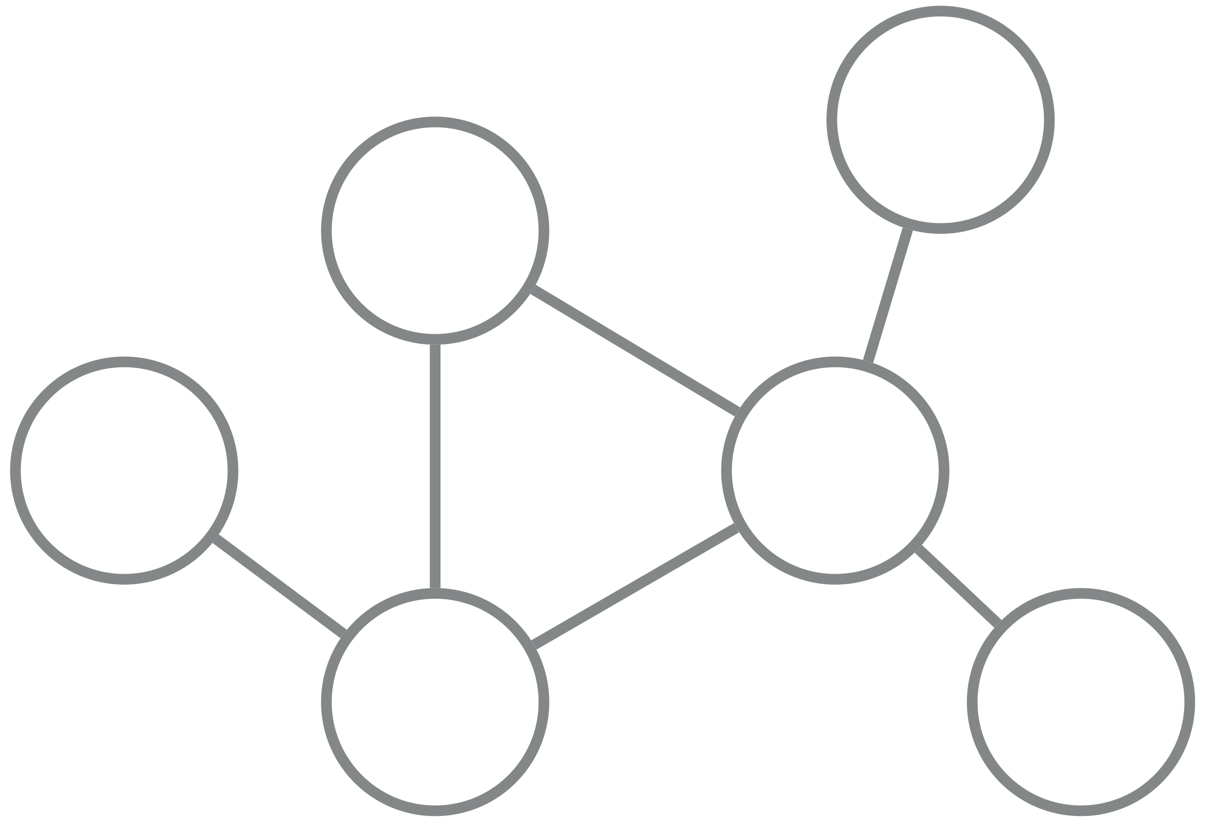
- Often applied to very large graphs (e.g., knowledge bases or linked document collections)



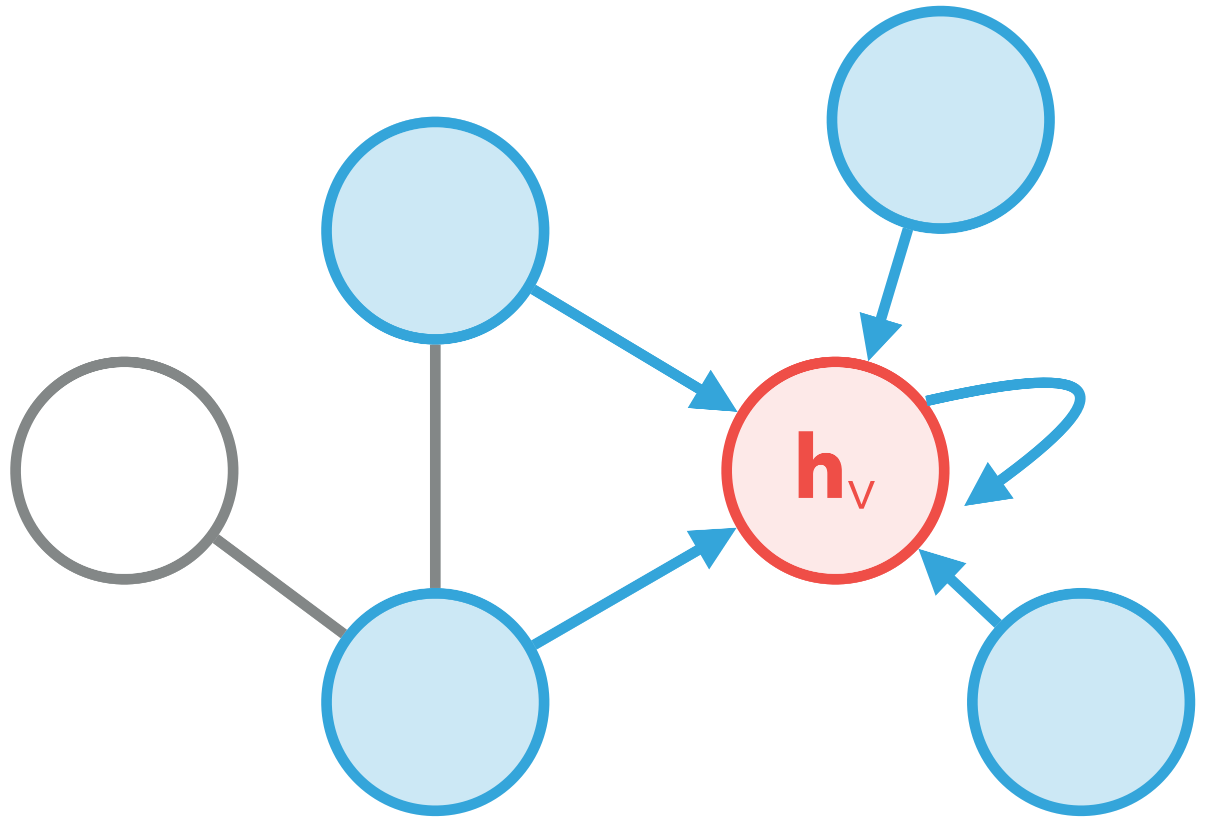


- No effective methods to answer:
  - Which paths a model relies on?
  - Which parts of the graphs are not needed?
  - …

Graph Neural Networks (GNNs)



Undirected graph



Update of red node

In layer k:

$$\mathbf{h}_v^{(k+1)} = \mathrm{ReLU}\Big(\sum_{u \in \mathrm{neighbors}(v)} \mathbf{m}_{u,v}^{(k)} \Big)$$

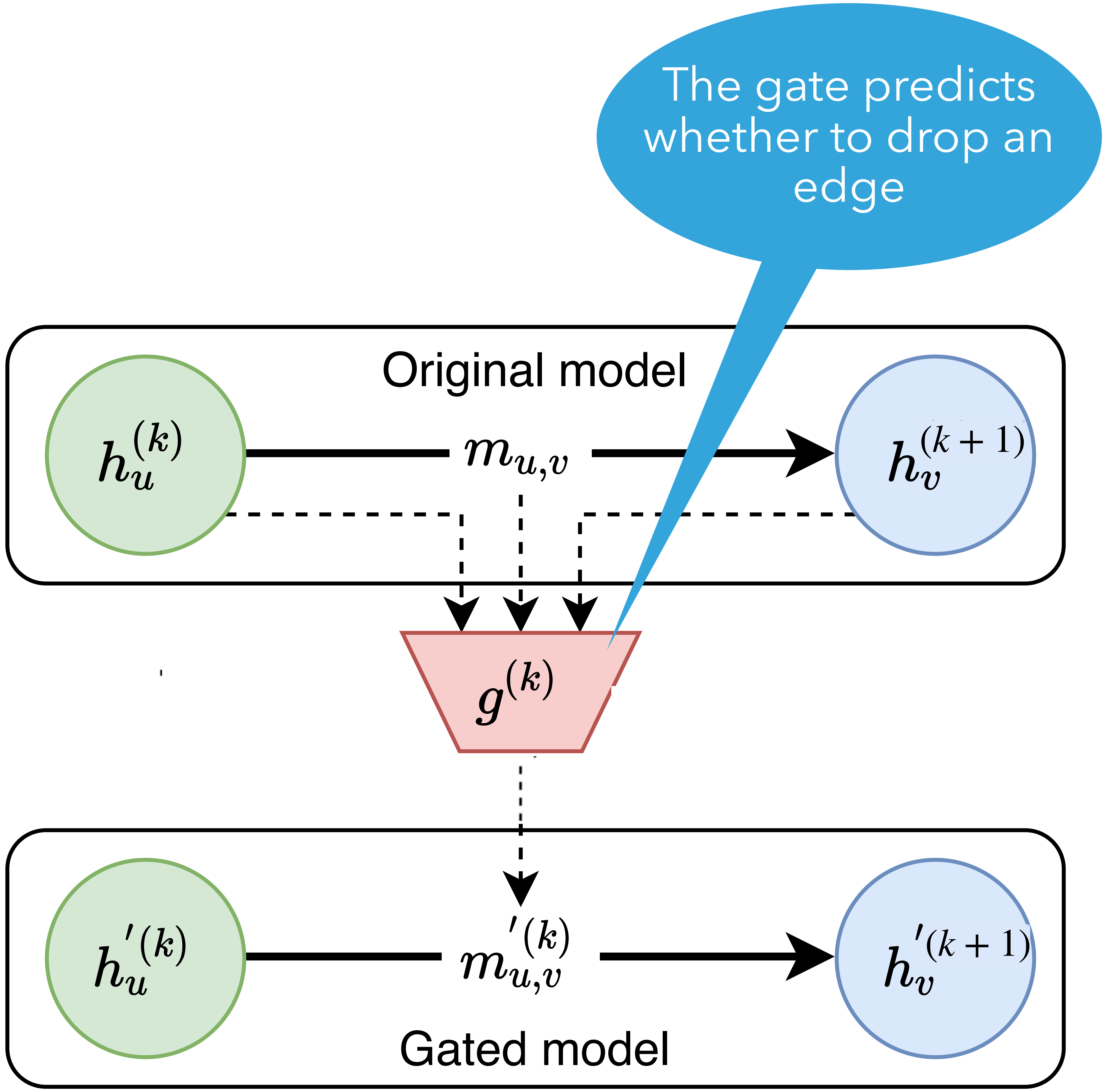
Closely related to Transformers

Applied to all nodes and in multiple layers

Extensions: handle labels, add attention

Masking Message in GNNs

- Execute GNN on original graph
- Predict which edges to keep
- Execute GNN on the pruned graph

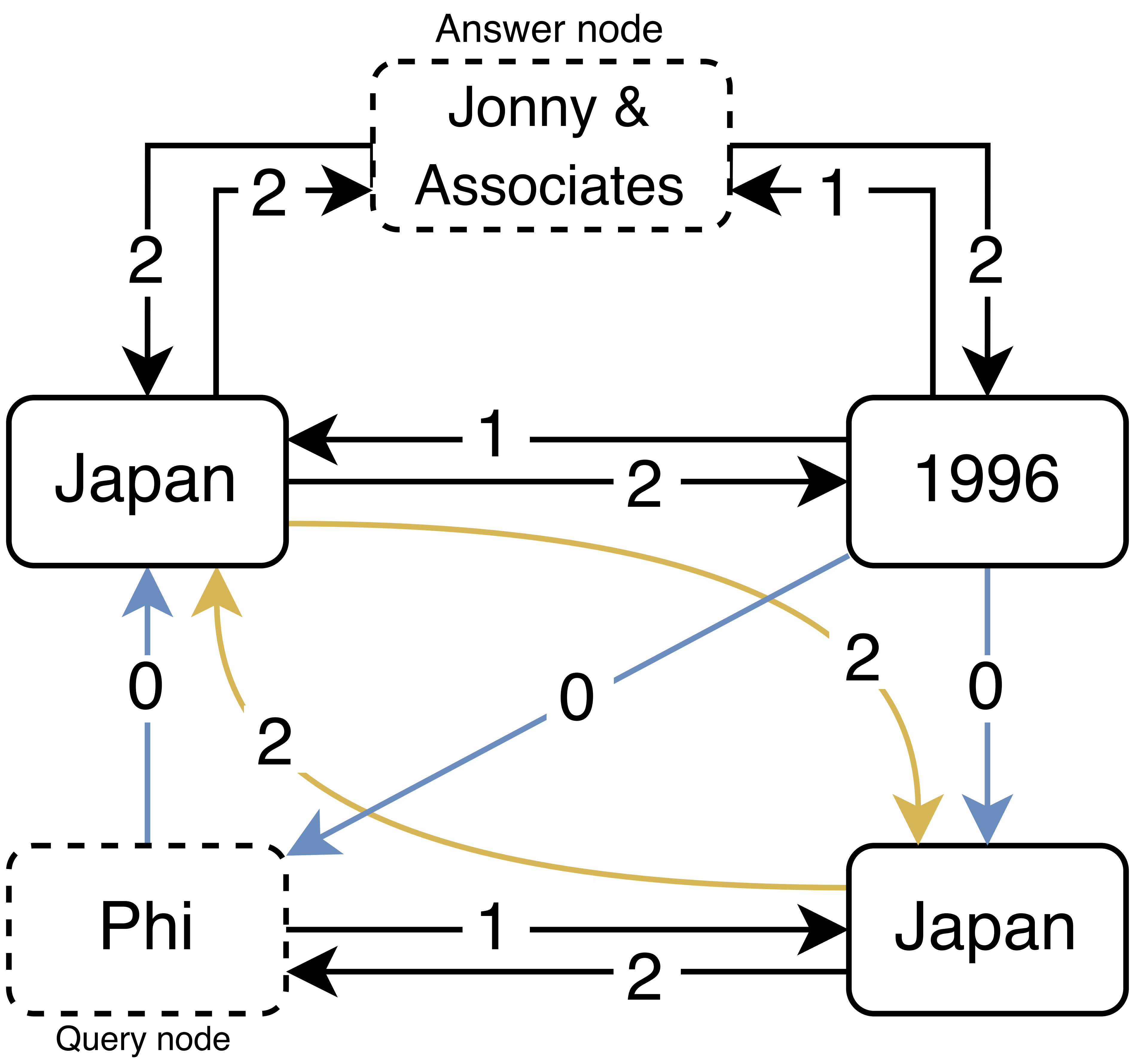


- Trained to agree with the original model, while masking as many edges as possible

Work in progress

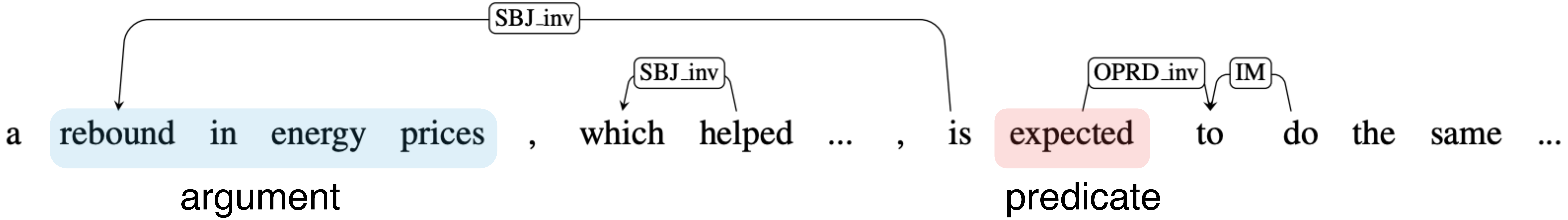
Analysing GNN-based QA-based models (Wikihop)

- Which edge in is used in which layer
- We can also uncover how information flows from nodes in the query to the answer (or alternatives)



Work in progress

Analysing GNN-based syntactically-driven models



- Discovery of predicate-argument structure (~ information extraction)
- It turns out the extractor relies only on a tiny sub-part of the syntactic tree!

Accuracy of the sparsified computation is the same as of the original model

Work in progress

Conclusions

- GNNs can be used to incorporate various types of inductive biases
- They can be made interpretable
- There also lots of open problems and opportunities to do interesting research!

European Research Council
Nederlandse Organisatie voor Wetenschappelijk Onderzoek
Yandex
Google Research Award
SAP