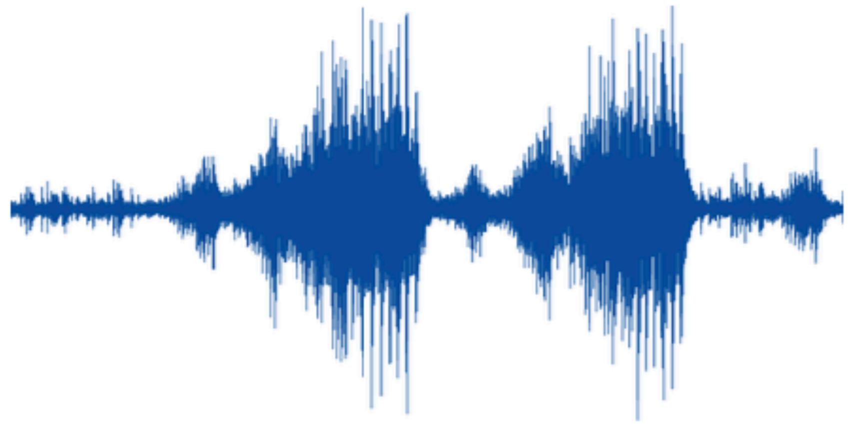
Topics in AI (CPSC 532S):
Multimodal Learning with Vision, Language and Sound

Lecture 19: Graph Neural Networks (cont)

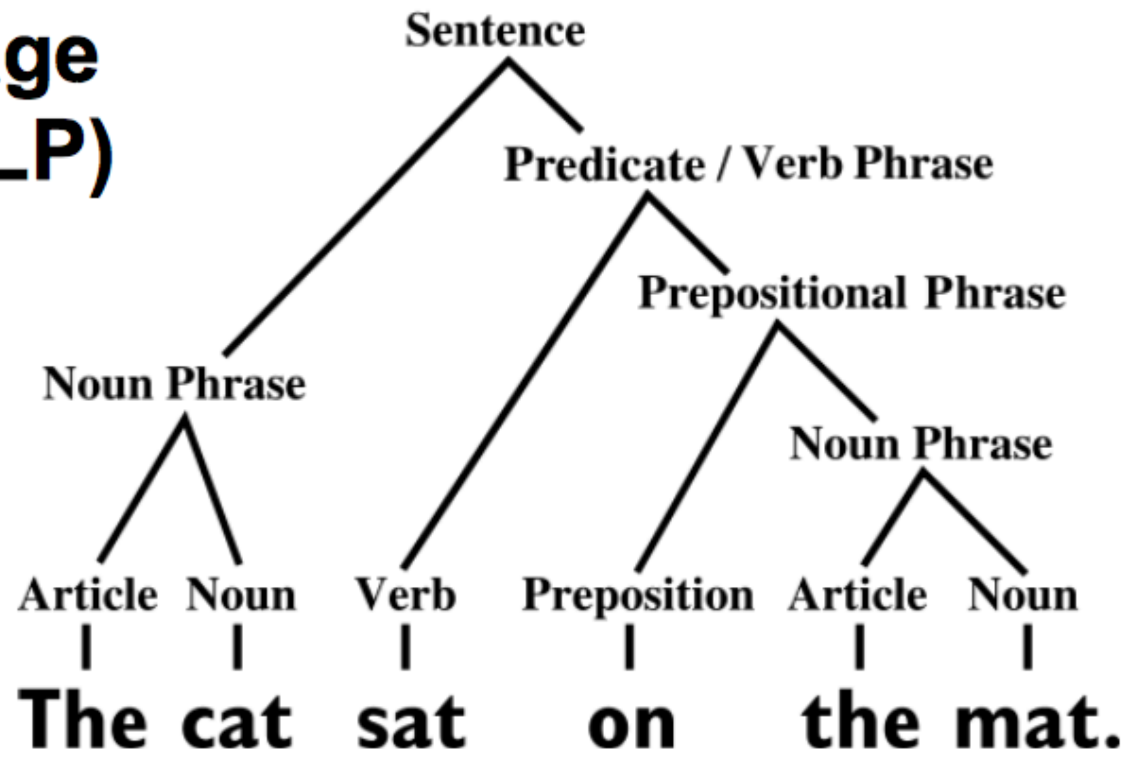
Traditional **Neural Networks**

IMAGENET

Speech data

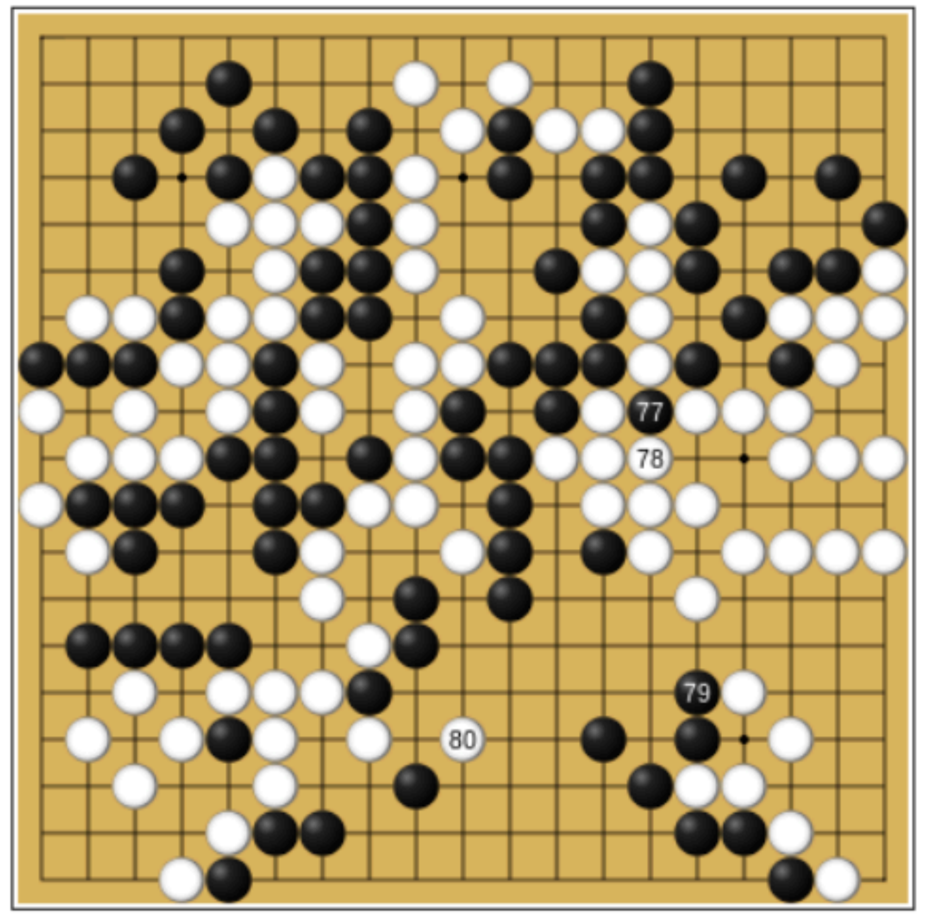


Natural language processing (NLP)



...

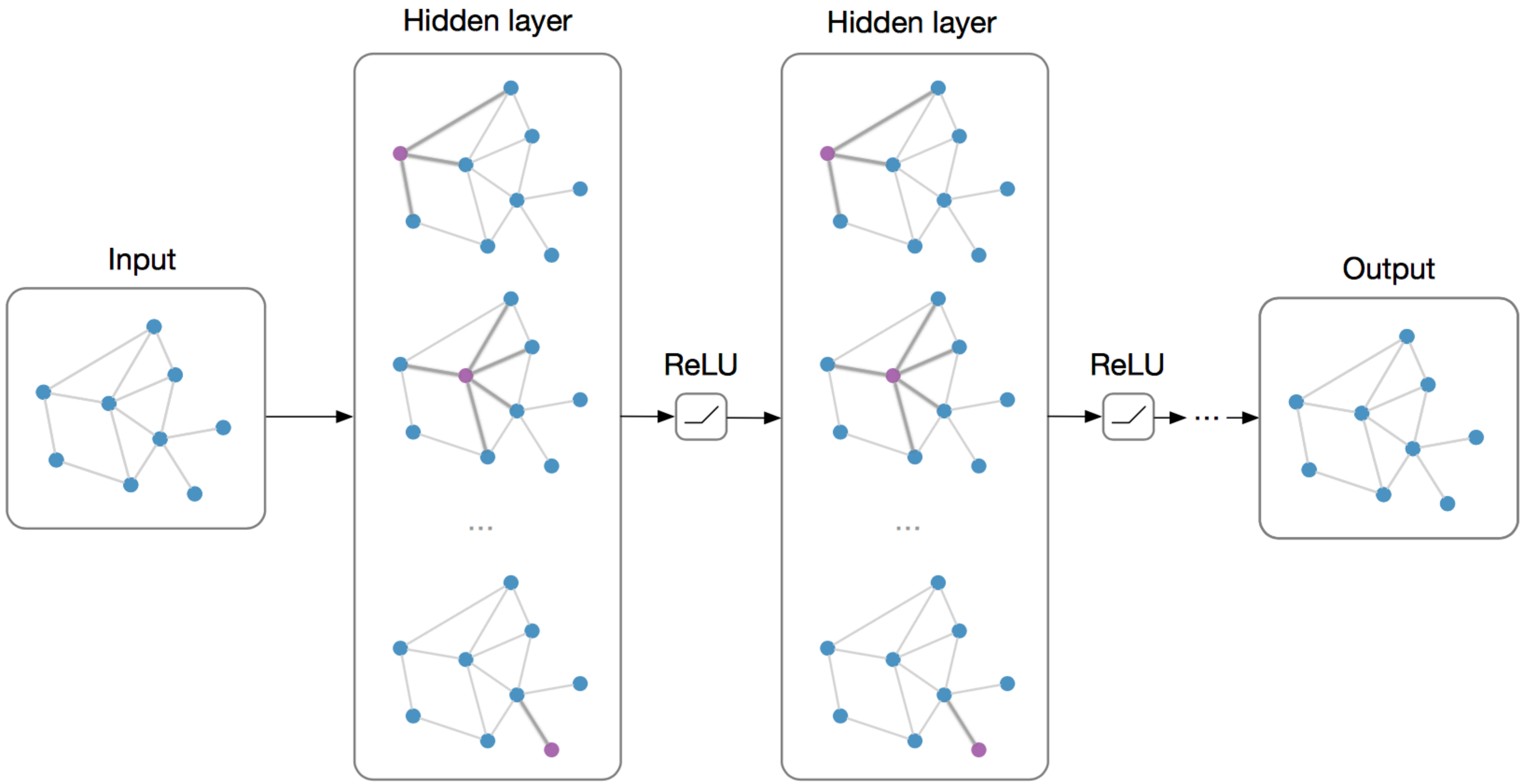
Grid games



Deep neural nets that exploit:

- translation equivariance (weight sharing)
- hierarchical compositionality

Graph Neural Networks (GNNs)



Main Idea: Pass massages between pairs of nodes and agglomerate

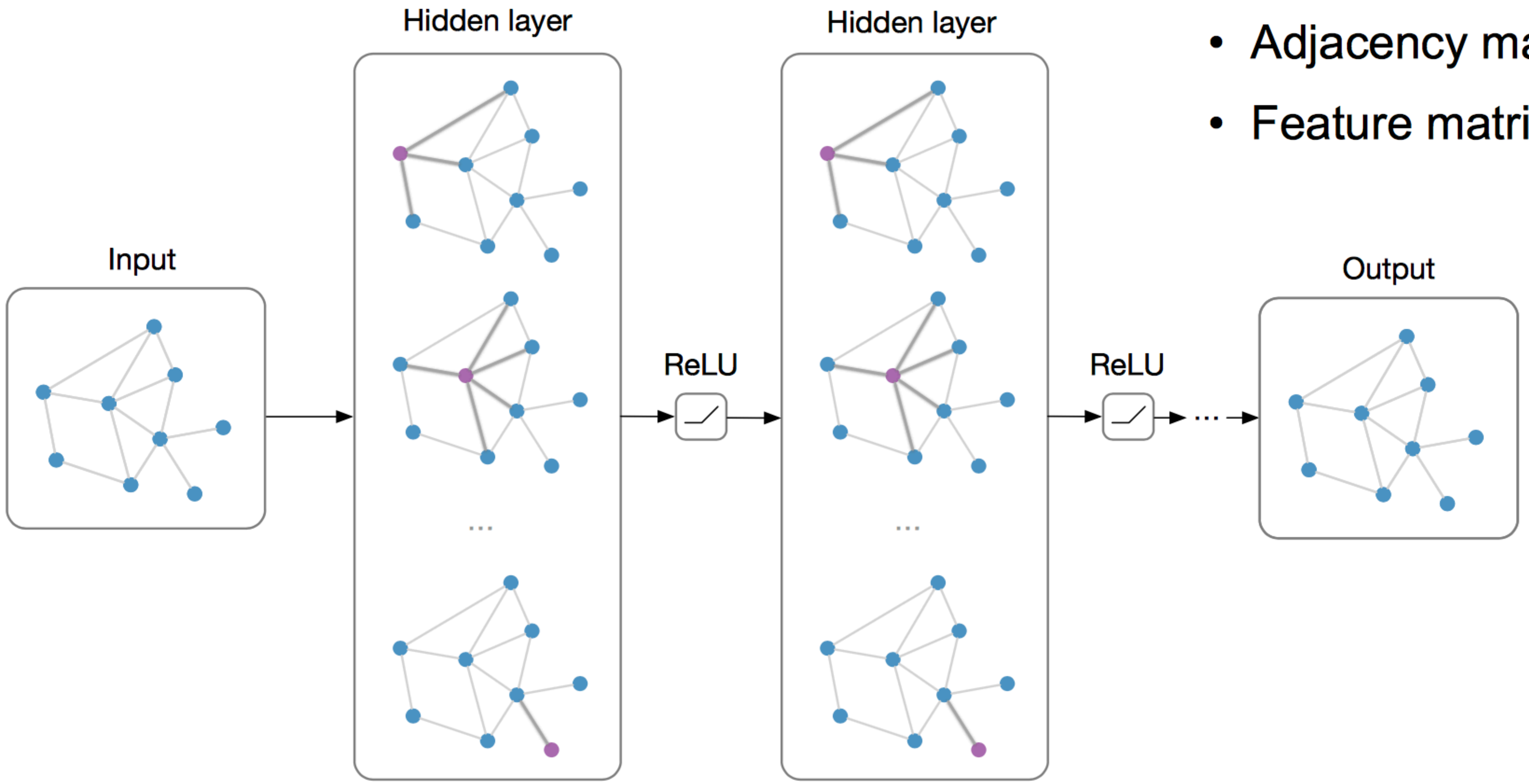
Alternative Interpretation: Pass massages between nodes to refine node (and possibly edge) representations

* slide from Thomas Kipf, **University of Amsterdam**

Graph Neural Networks (GNNs)

Notation: $\mathcal{G} = (\mathbf{A}, \mathbf{X})$

- Adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$
- Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times F}$



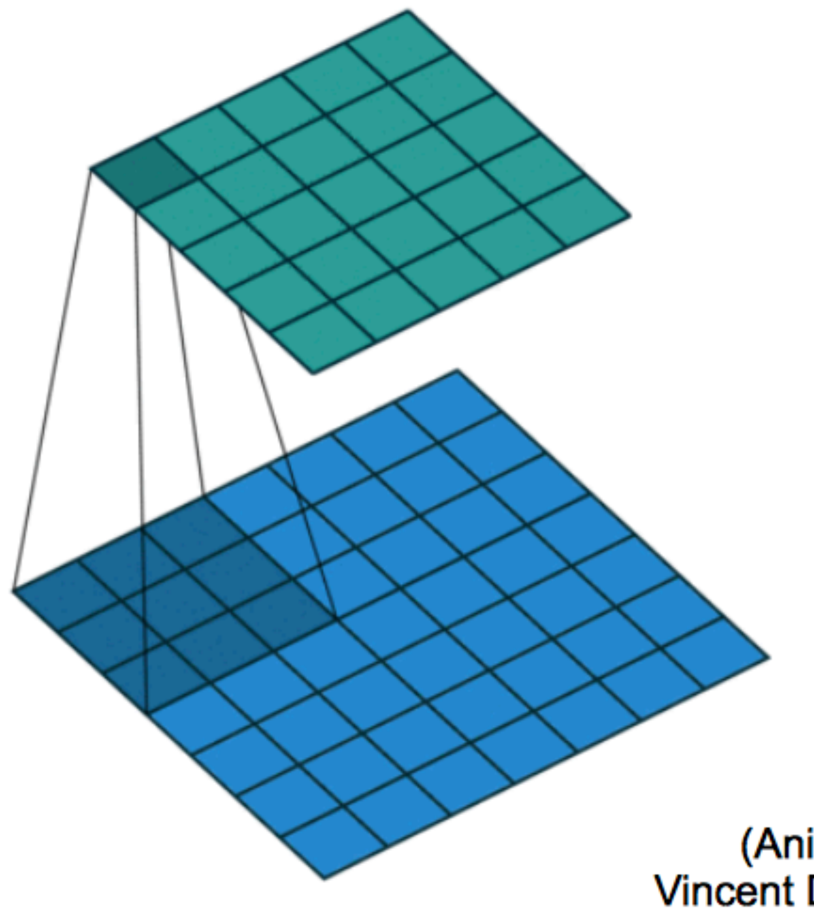
Main Idea: Pass massages between pairs of nodes and agglomerate

Alternative Interpretation: Pass massages between nodes to refine node (and possibly edge) representations

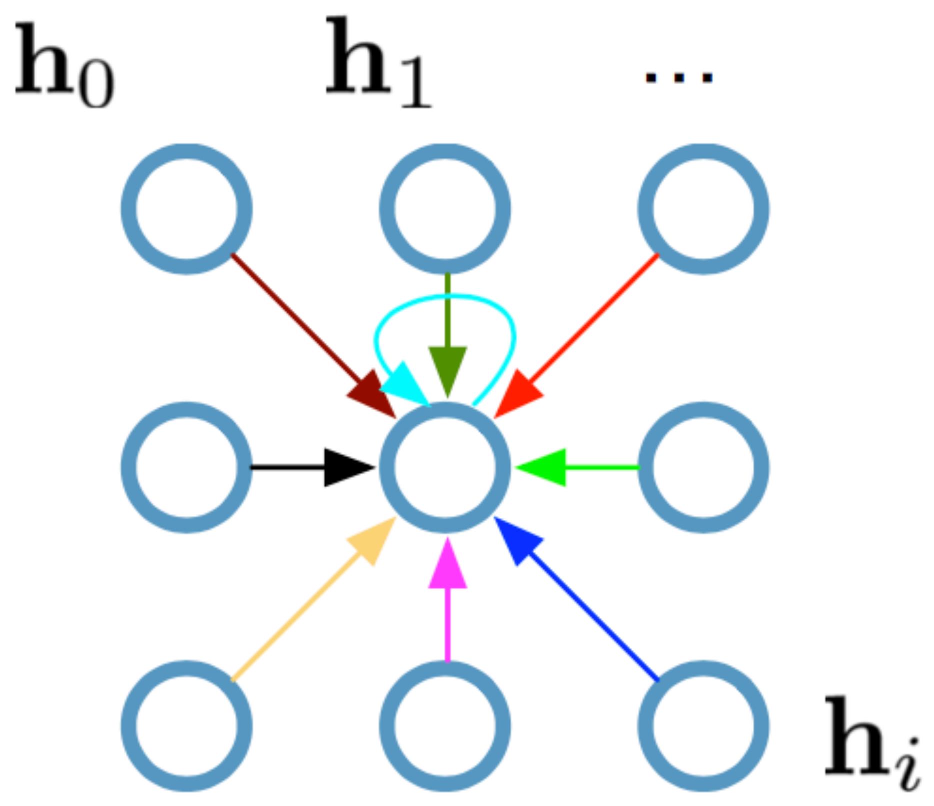
* slide from Thomas Kipf, **University of Amsterdam**

Recap: Convolutional Neural Networks (CNNs) on Grids

Single CNN layer with 3x3 filter:

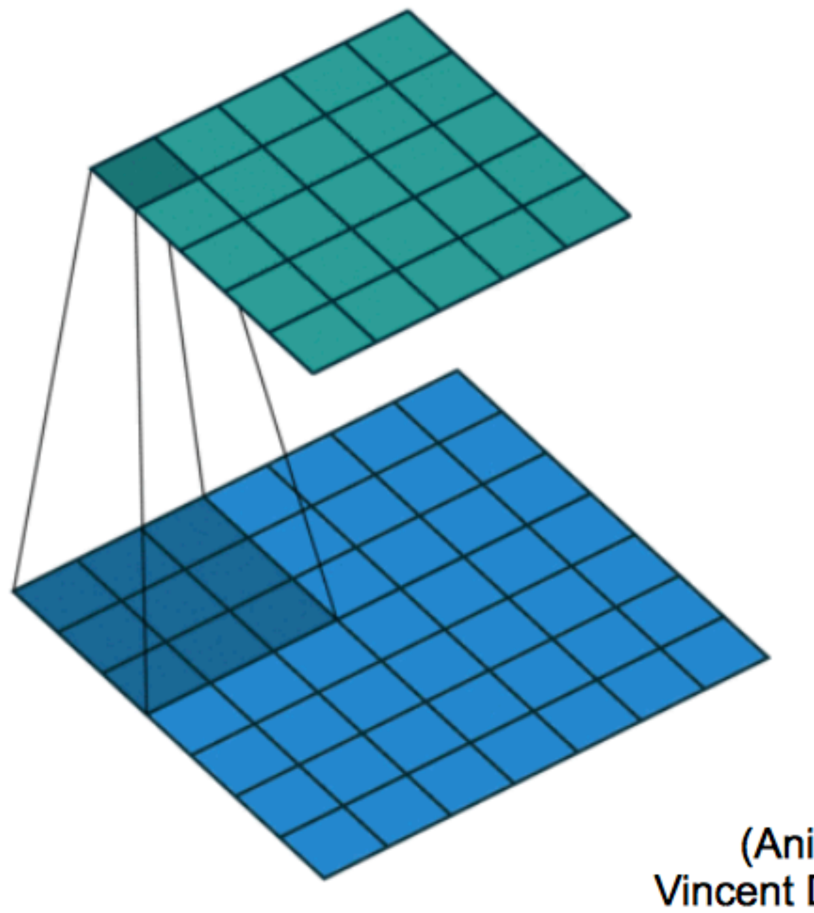


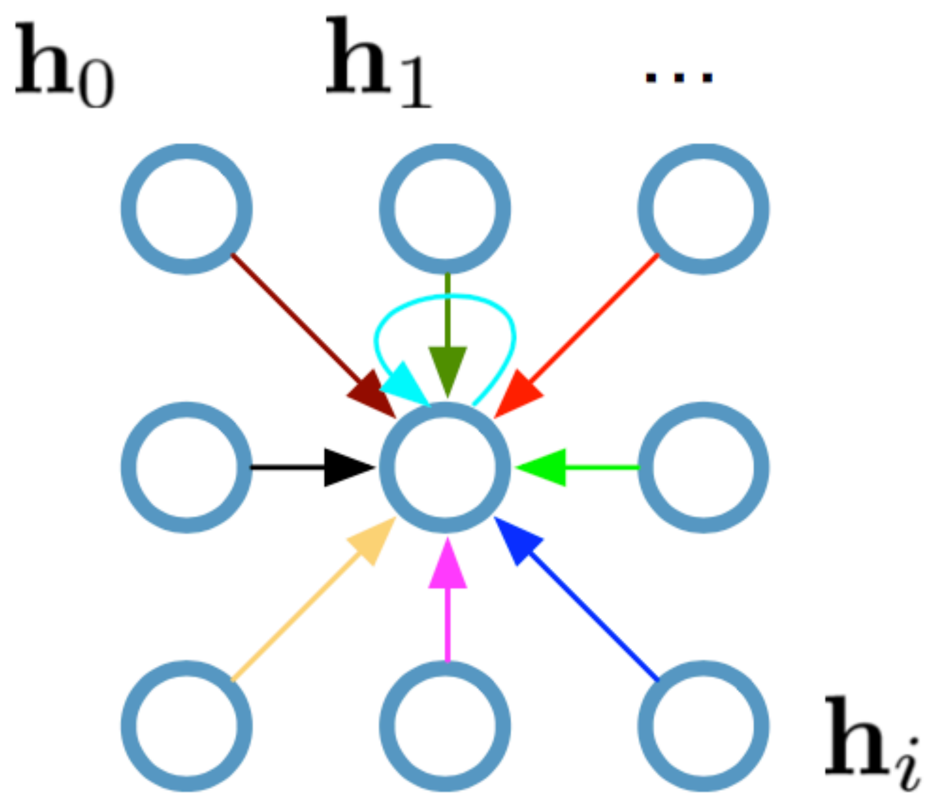
(Animation by Vincent Dumoulin)



Recap: Convolutional Neural Networks (CNNs) on Grids

Single CNN layer with 3x3 filter:

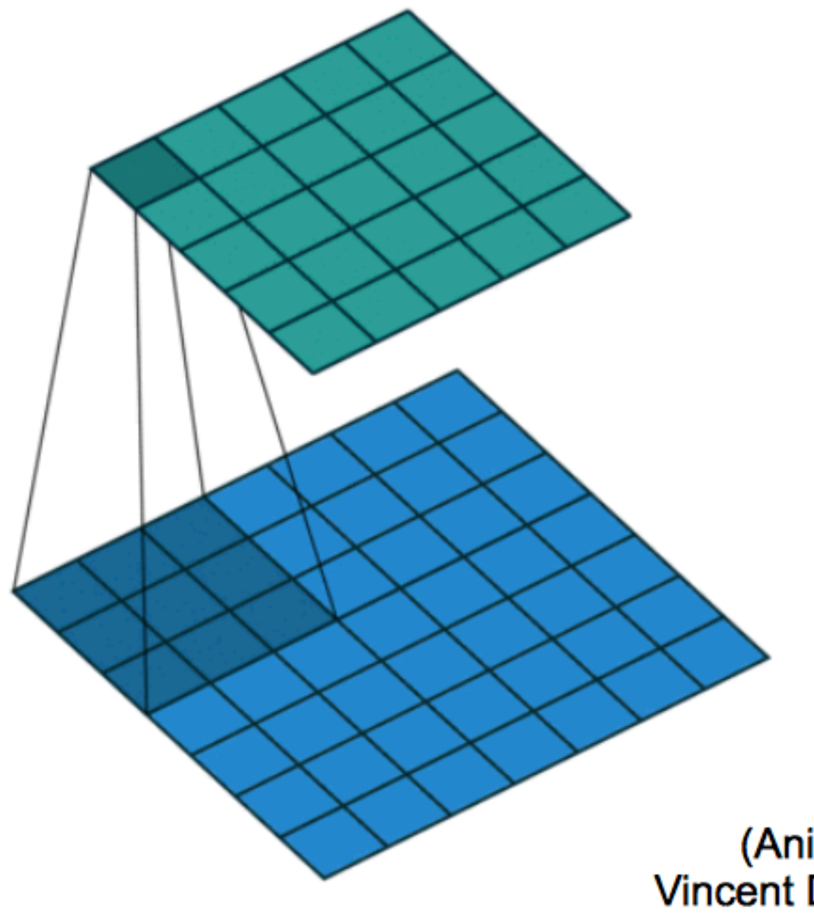

(Animation by Vincent Dumoulin)



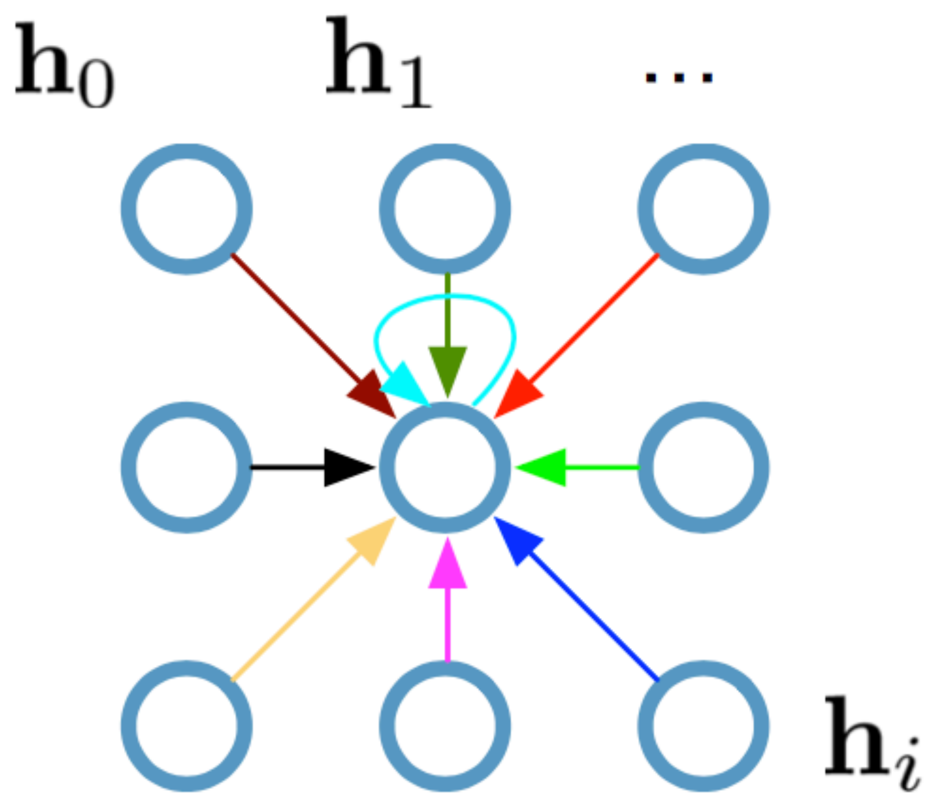
$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

Recap: Convolutional Neural Networks (CNNs) on Grids

Single CNN layer with 3x3 filter:



(Animation by Vincent Dumoulin)



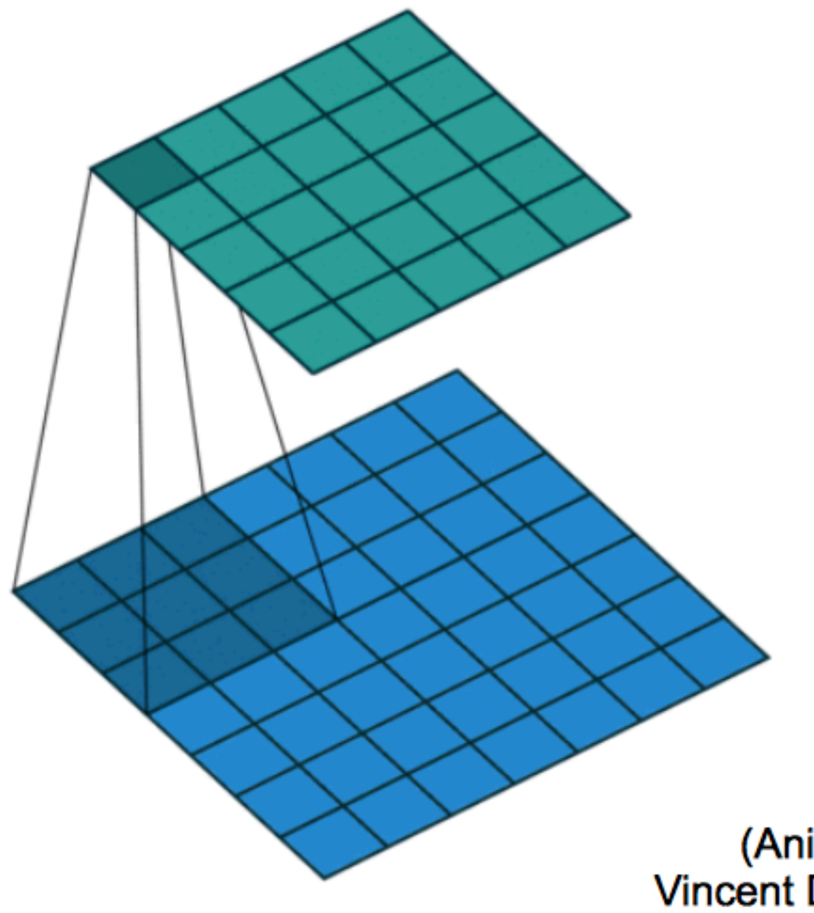
Update for a single pixel:

- Transform messages individually $\mathbf{W}_i\mathbf{h}_i$
- Add everything up $\sum_i \mathbf{W}_i\mathbf{h}_i$

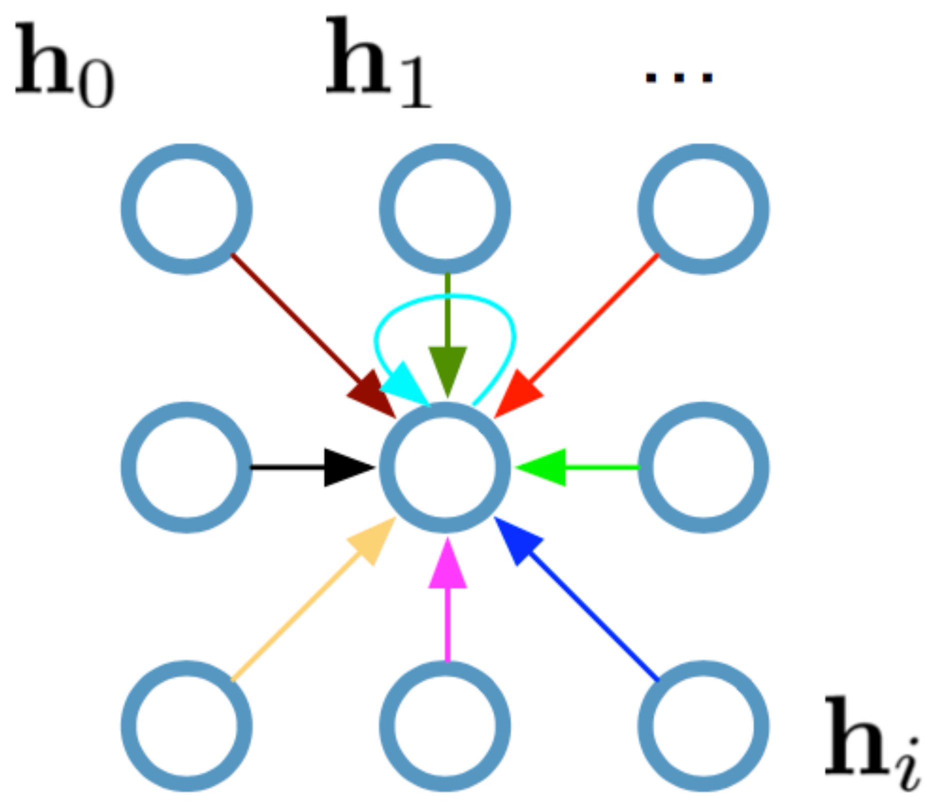
$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

Recap: Convolutional Neural Networks (CNNs) on Grids

Single CNN layer with 3x3 filter:



(Animation by Vincent Dumoulin)



Update for a single pixel:

- Transform messages individually $\mathbf{W}_i \mathbf{h}_i$
- Add everything up $\sum_i \mathbf{W}_i \mathbf{h}_i$

$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

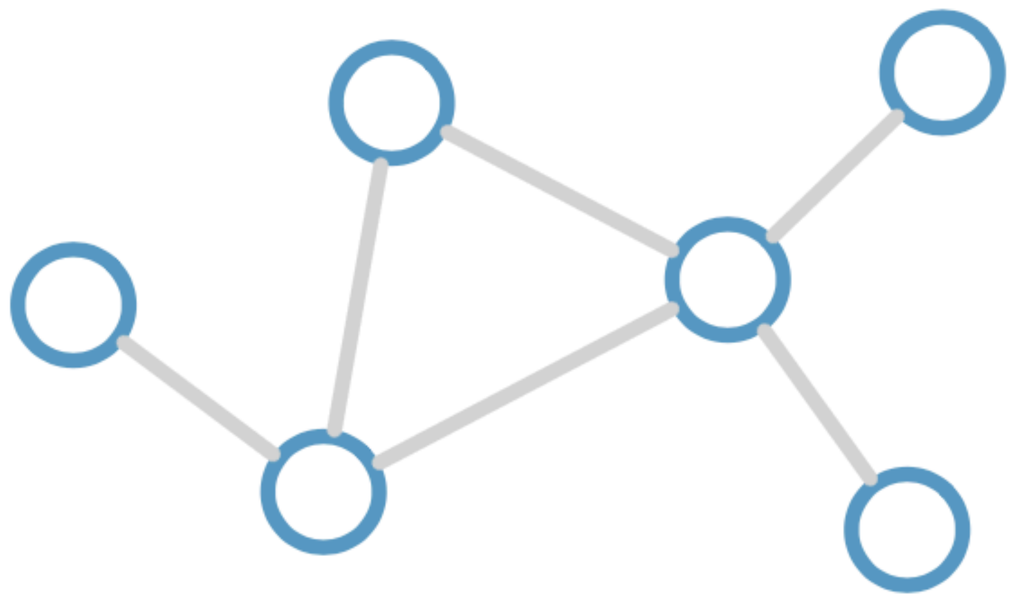
Full update:

$$\mathbf{h}_4^{(l+1)} = \sigma\left(\mathbf{W}_0^{(l)}\mathbf{h}_0^{(l)} + \mathbf{W}_1^{(l)}\mathbf{h}_1^{(l)} + \cdots + \mathbf{W}_8^{(l)}\mathbf{h}_8^{(l)}\right)$$

Graph Convolutional Networks (GCNs)

Kipf & Welling (ICLR 2017), related previous works by Duvenaud et al. (NIPS 2015) and Li et al. (ICLR 2016)

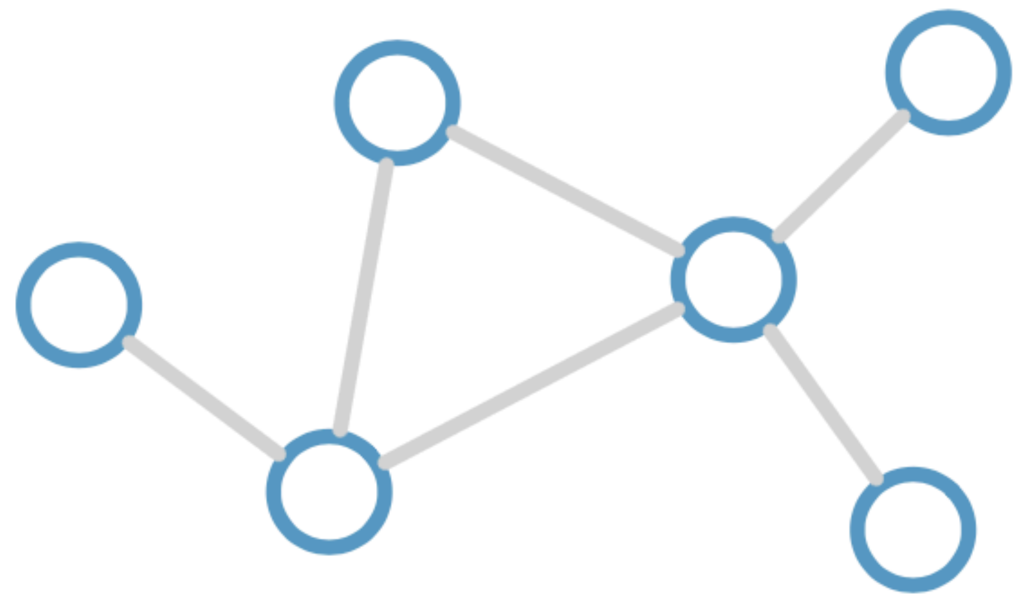
Consider this undirected graph:



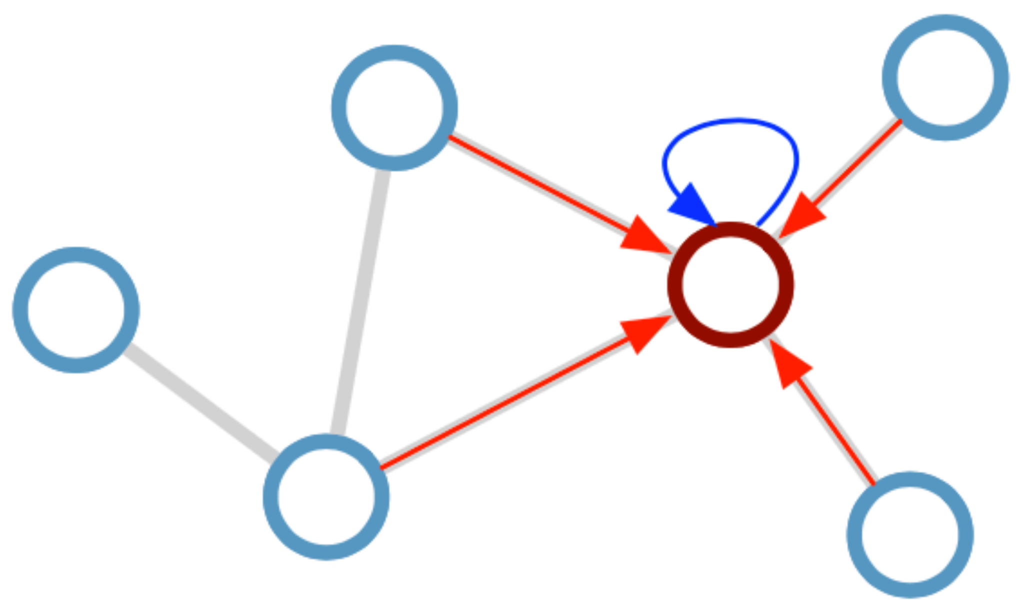
Graph Convolutional Networks (GCNs)

Kipf & Welling (ICLR 2017), related previous works by Duvenaud et al. (NIPS 2015) and Li et al. (ICLR 2016)

Consider this undirected graph:



Calculate update for node in red:

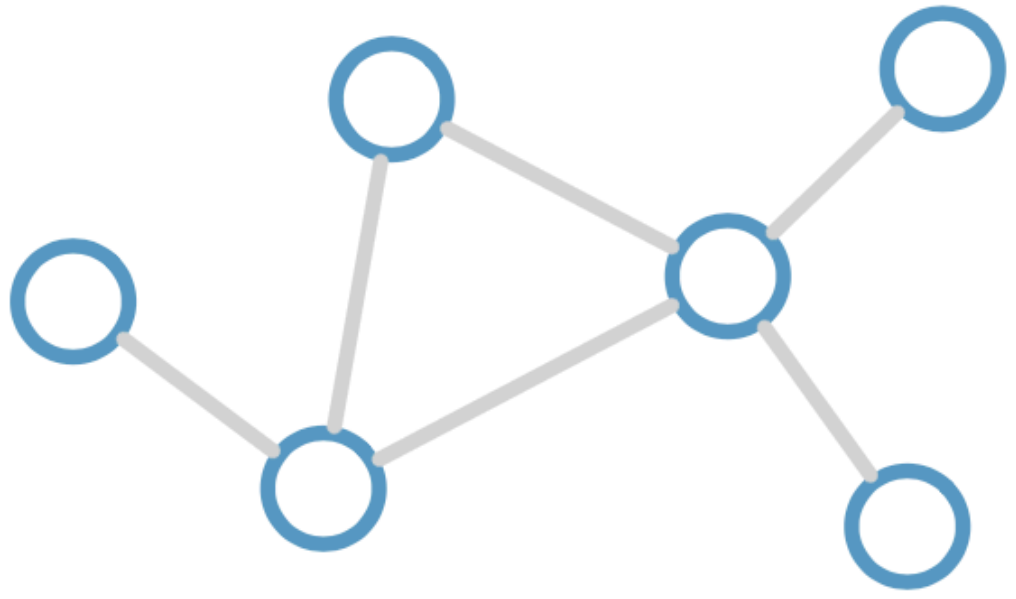


* slide from Thomas Kipf, **University of Amsterdam**

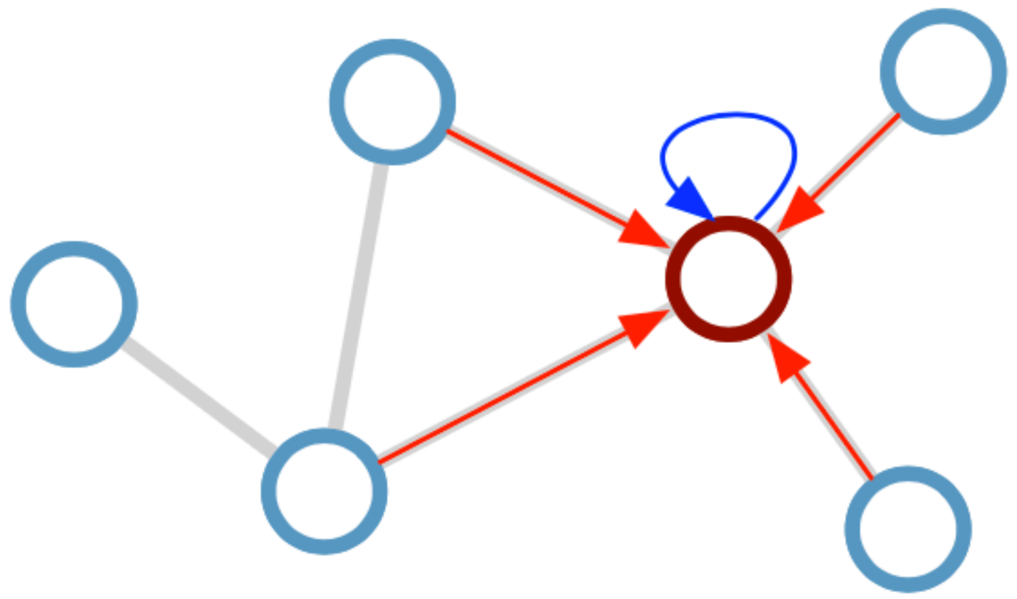
Graph Convolutional Networks (GCNs)

Kipf & Welling (ICLR 2017), related previous works by Duvenaud et al. (NIPS 2015) and Li et al. (ICLR 2016)

Consider this undirected graph:



Calculate update for node in red:



Update rule:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

Scalability: subsample messages [Hamilton et al., NIPS 2017]

$\mathcal{N}_i$: neighbor indices

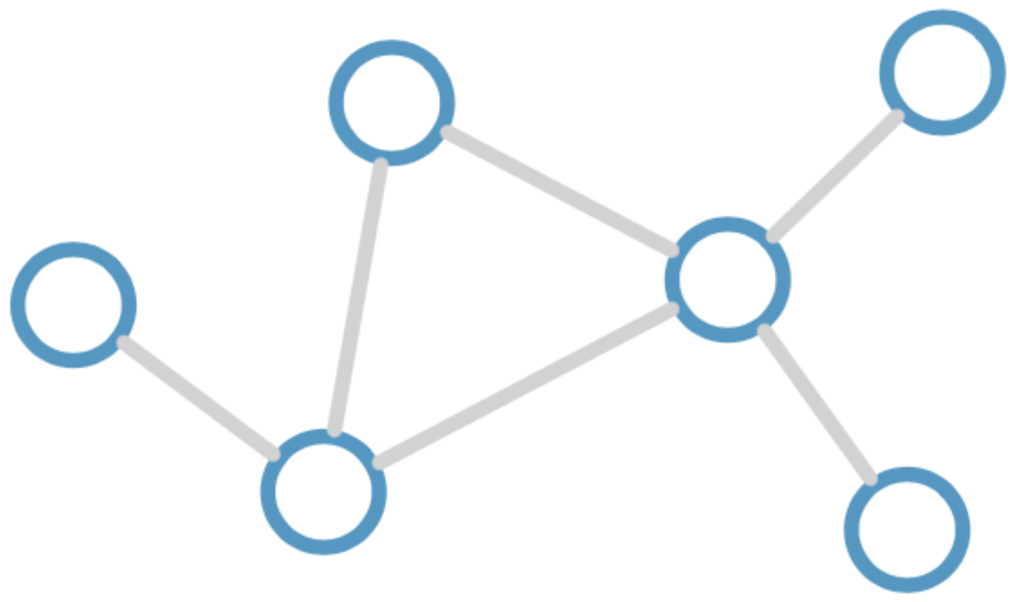
c_{ij} : norm. constant (fixed/trainable)

* slide from Thomas Kipf, **University of Amsterdam**

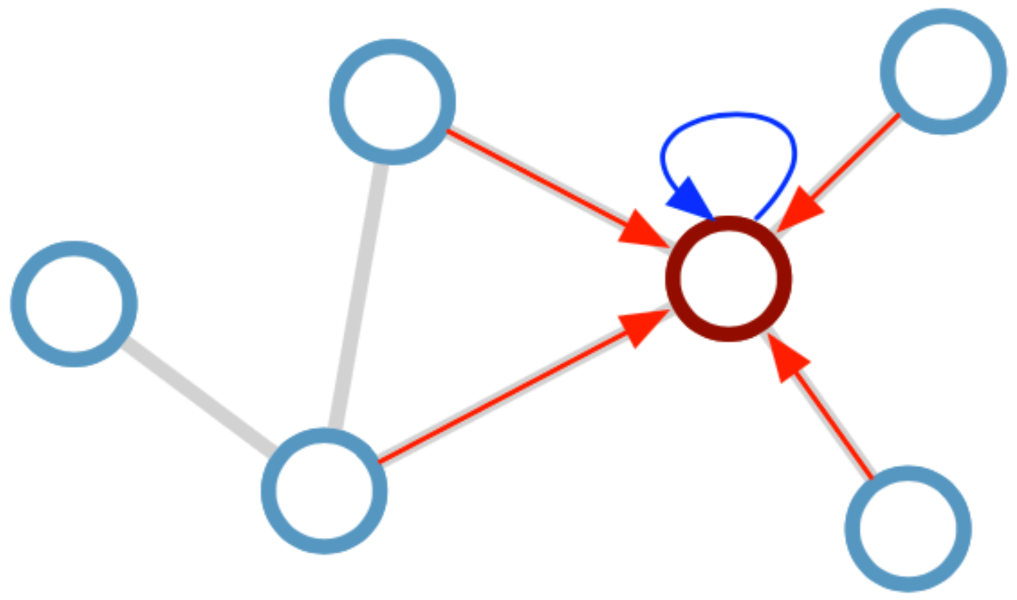
Graph Convolutional Networks (GCNs)

Kipf & Welling (ICLR 2017), related previous works by Duvenaud et al. (NIPS 2015) and Li et al. (ICLR 2016)

Consider this undirected graph:



Calculate update for node in red:



Desirable properties:

- Weight sharing over all locations
- Invariance to permutations
- Linear complexity O(E)
- Applicable both in transductive and inductive settings

Update rule:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

Scalability: subsample messages [Hamilton et al., NIPS 2017]

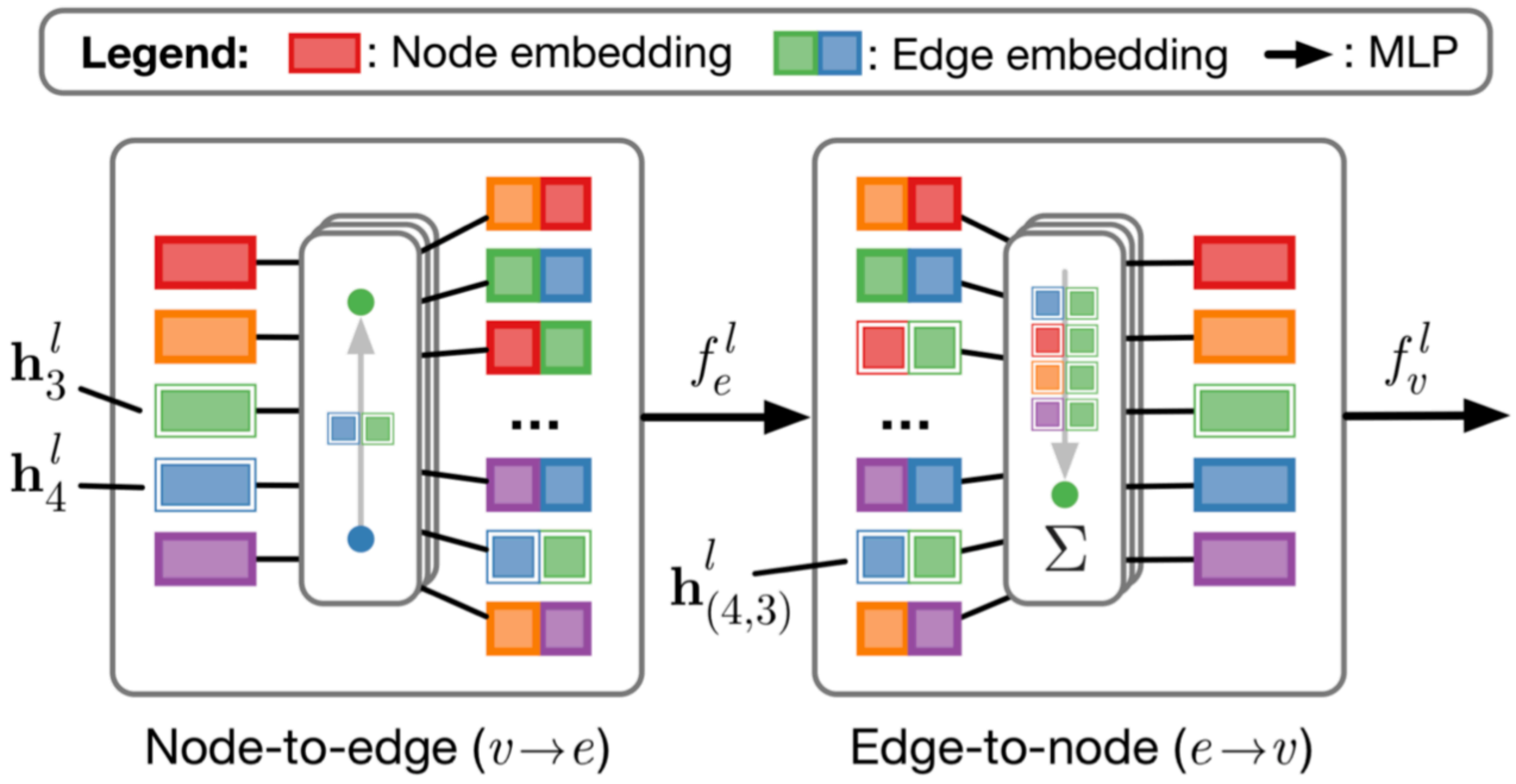
$\mathcal{N}_i$: neighbor indices

c_{ij} : norm. constant (fixed/trainable)

* slide from Thomas Kipf, **University of Amsterdam**

GNNs with **Edge** Embeddings

Battaglia et al. (NIPS 2016), Gilmer et al. (ICML 2017), Kipf et al. (ICML 2018)



Formally:

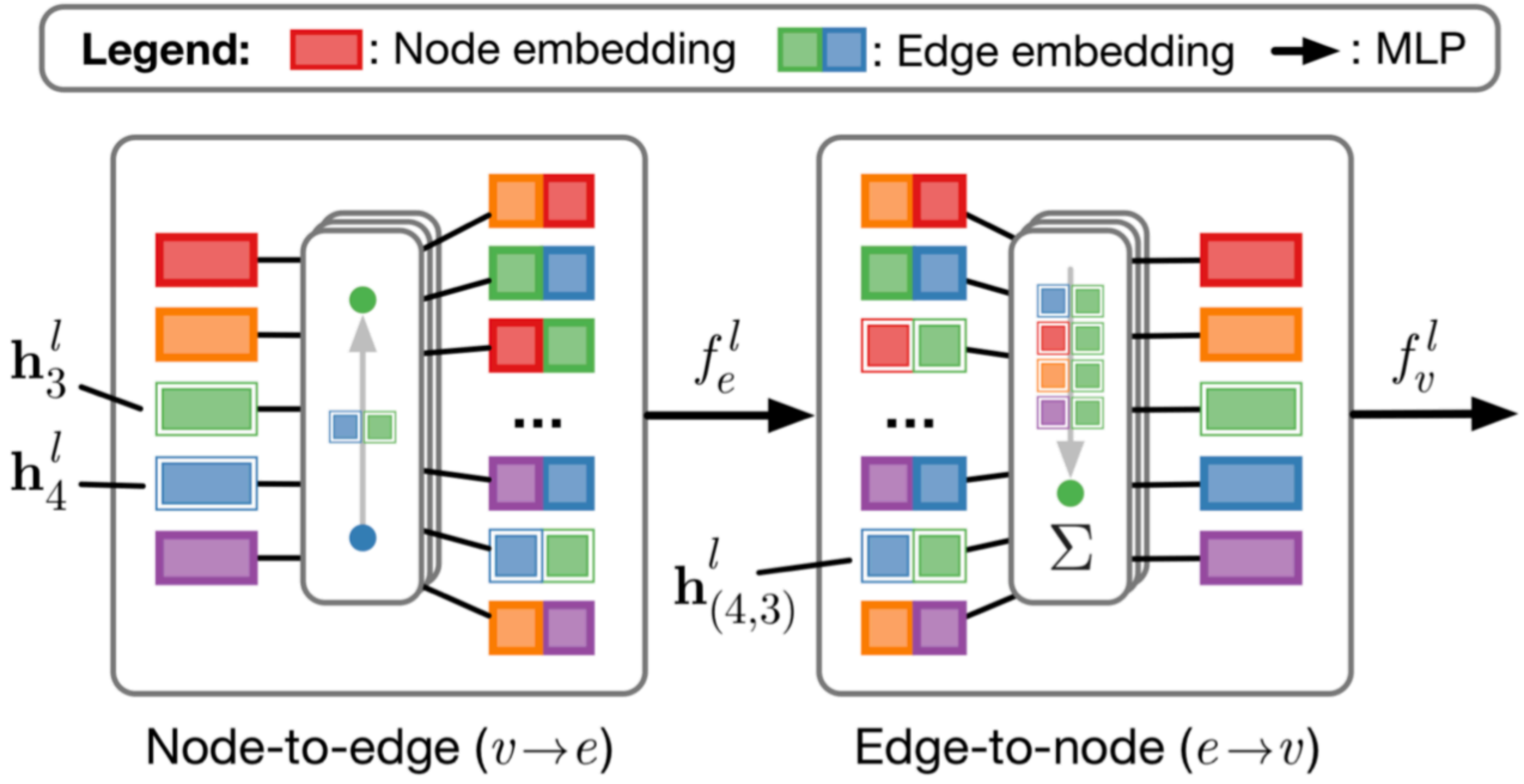
$$v \to e: \quad \mathbf{h}_{(i,j)}^l = f_e^l([\mathbf{h}_i^l, \mathbf{h}_j^l, \mathbf{x}_{(i,j)}])$$

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

* slide from Thomas Kipf, **University of Amsterdam**

GNNs with **Edge** Embeddings

Battaglia et al. (NIPS 2016), Gilmer et al. (ICML 2017), Kipf et al. (ICML 2018)



Pros:

- Supports edge features
- More expressive than GCN
- As general as it gets (?)
- Supports sparse matrix ops

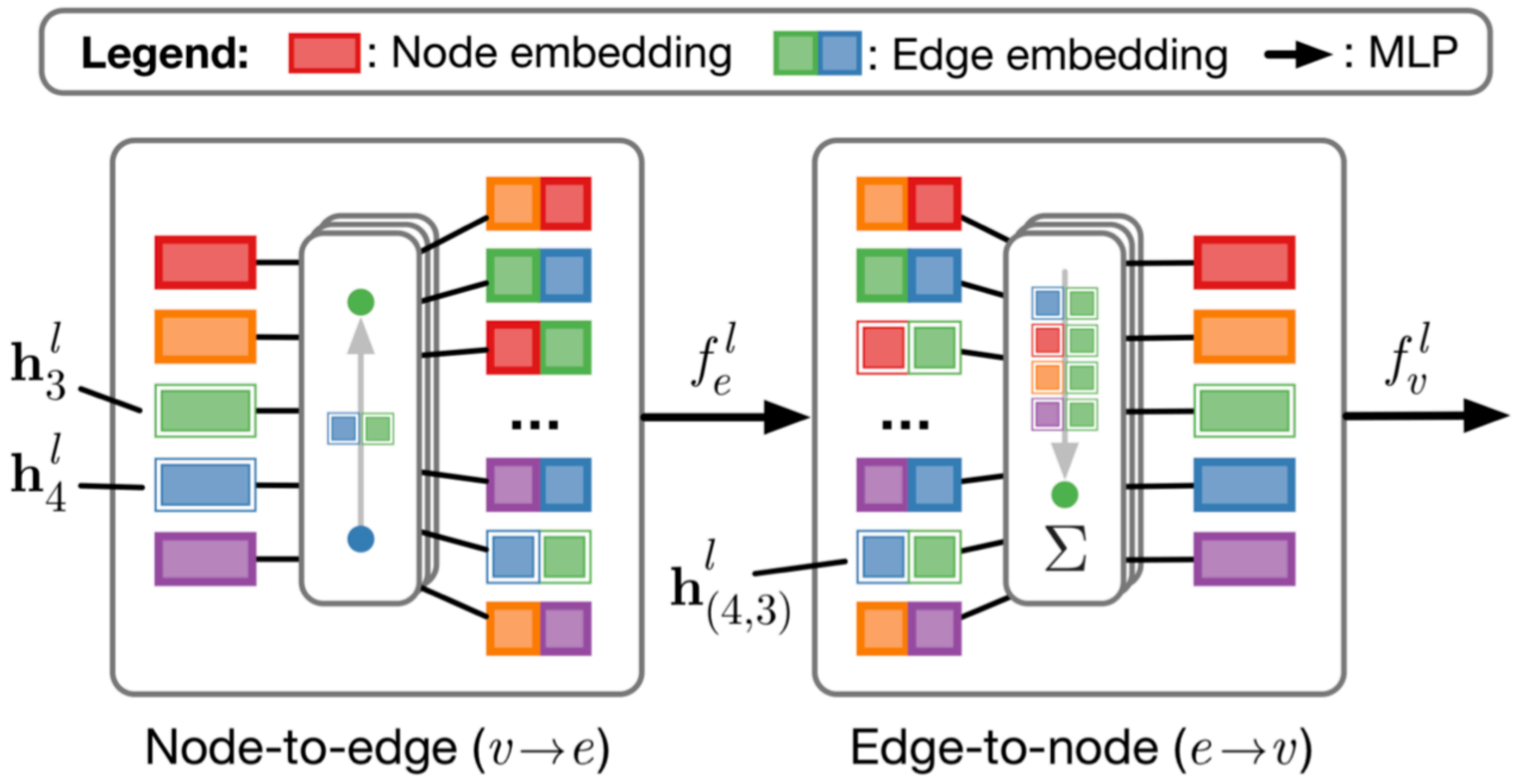
Formally:

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

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

GNNs with **Edge** Embeddings

Battaglia et al. (NIPS 2016), Gilmer et al. (ICML 2017), Kipf et al. (ICML 2018)



Formally:

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

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

Pros:

- Supports edge features
- More expressive than GCN
- As general as it gets (?)
- Supports sparse matrix ops

Cons:

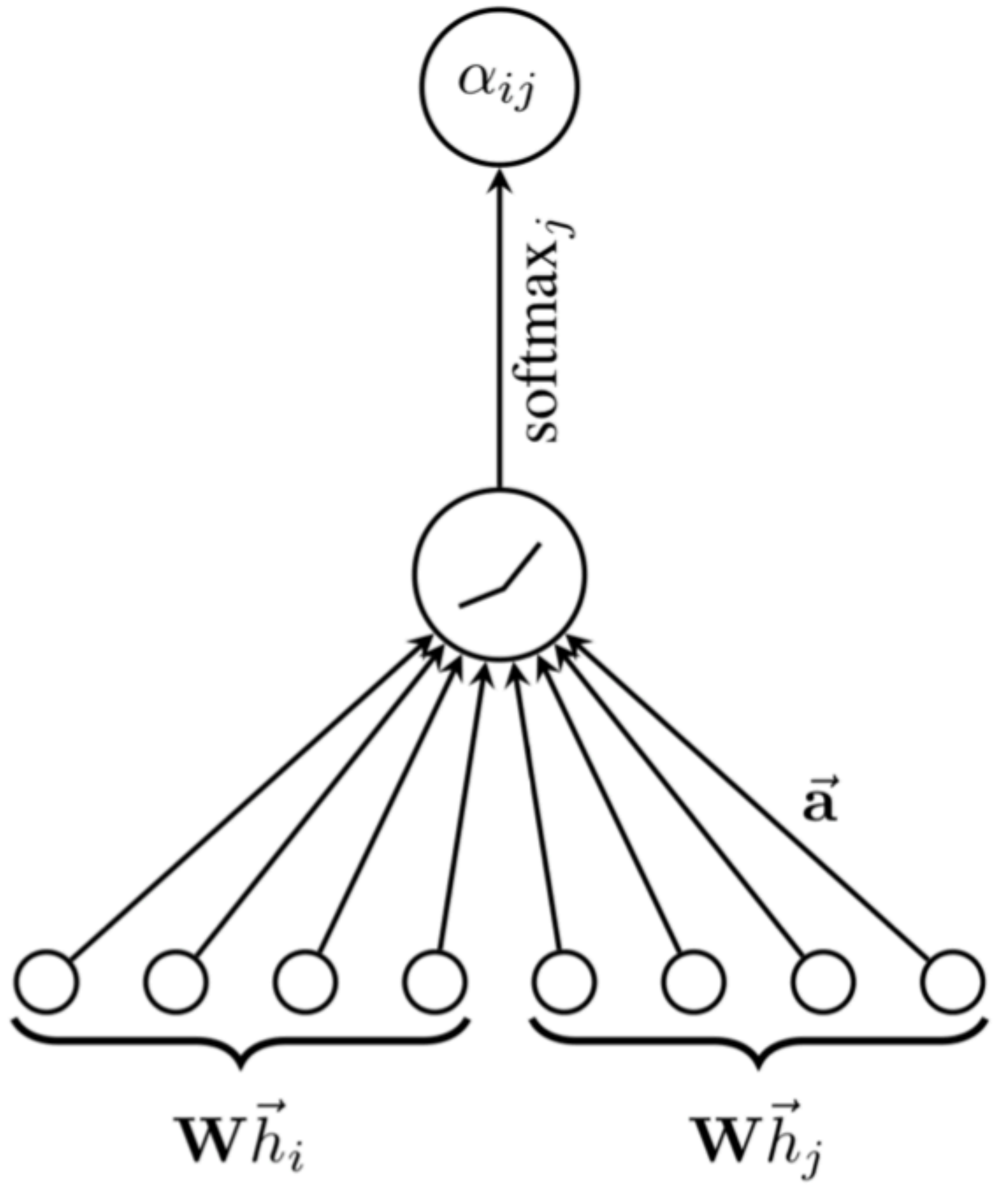
- Need to store intermediate edge-based activations
- Difficult to implement with subsampling

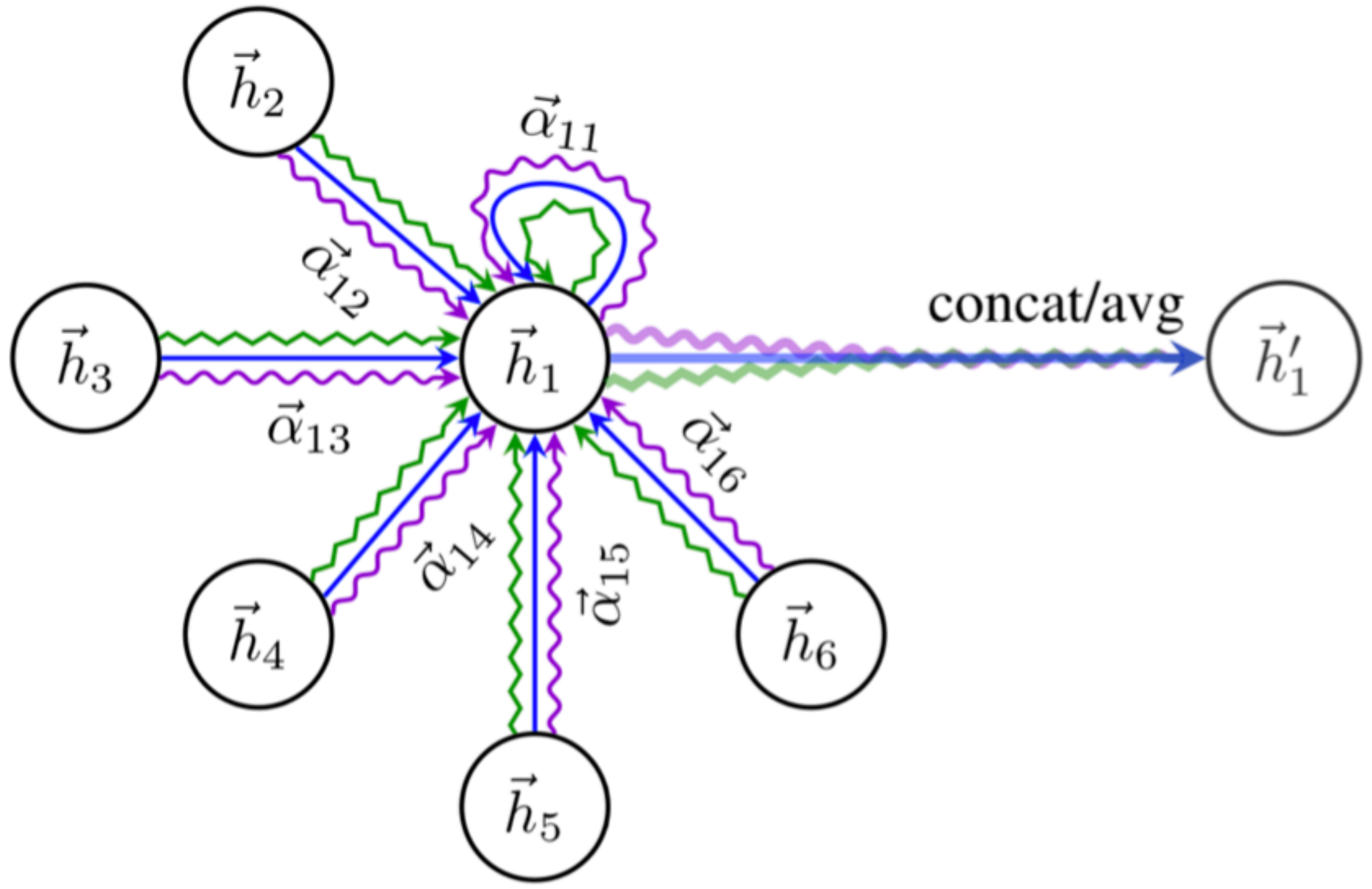
➡ In practice limited to small graphs

* slide from Thomas Kipf, **University of Amsterdam**

Graph Neural Networks (GNNs) with **Attention**

Monti et al. (CVPR 2017), Hoshen (NIPS 2017), Veličković et al. (ICLR 2018)





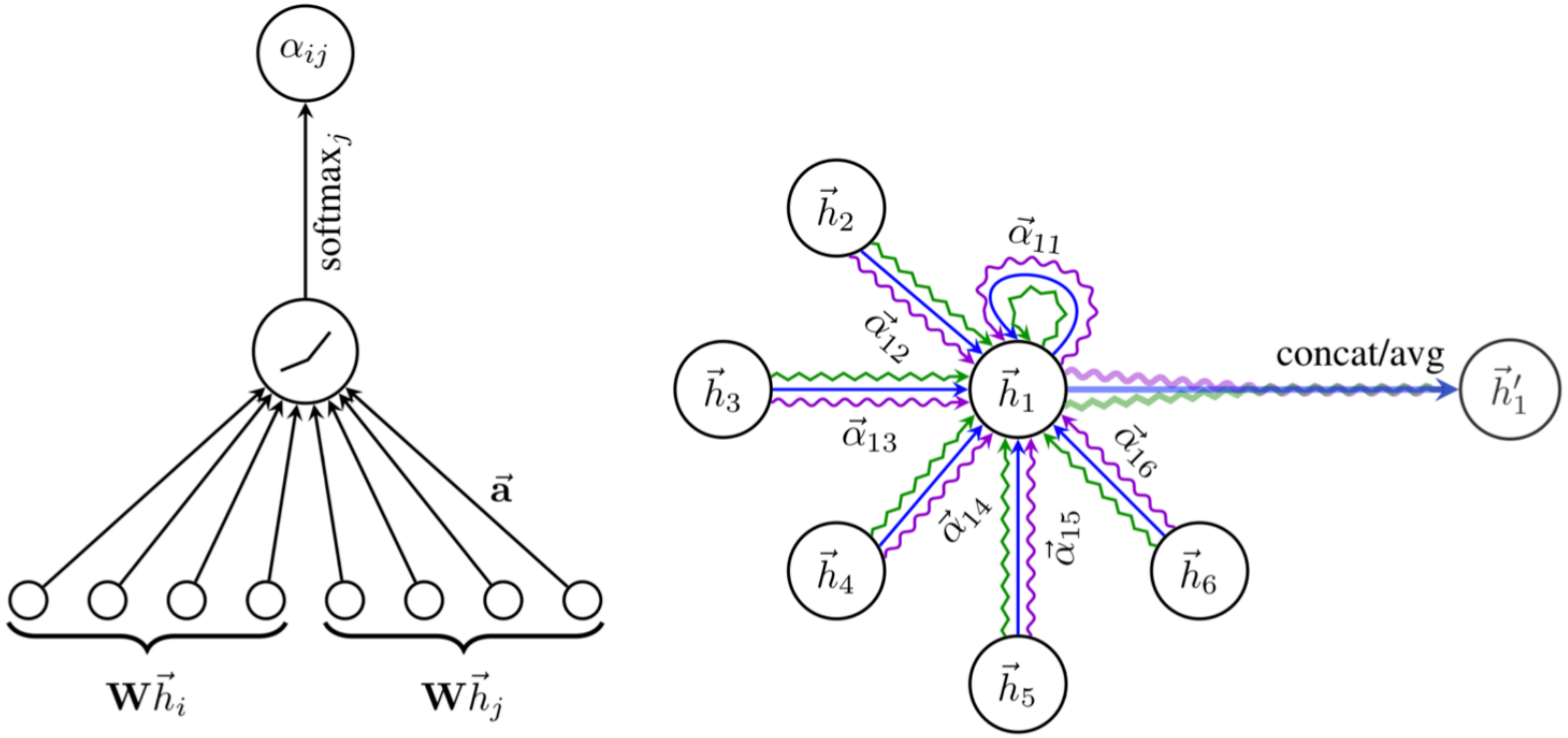
[Figure from Veličković et al. (ICLR 2018)]

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k\mathbf{W}^k\vec{h}_j\right)$$

* slide from Thomas Kipf, **University of Amsterdam**

Graph Neural Networks (GNNs) with **Attention**

Monti et al. (CVPR 2017), Hoshen (NIPS 2017), Veličković et al. (ICLR 2018)



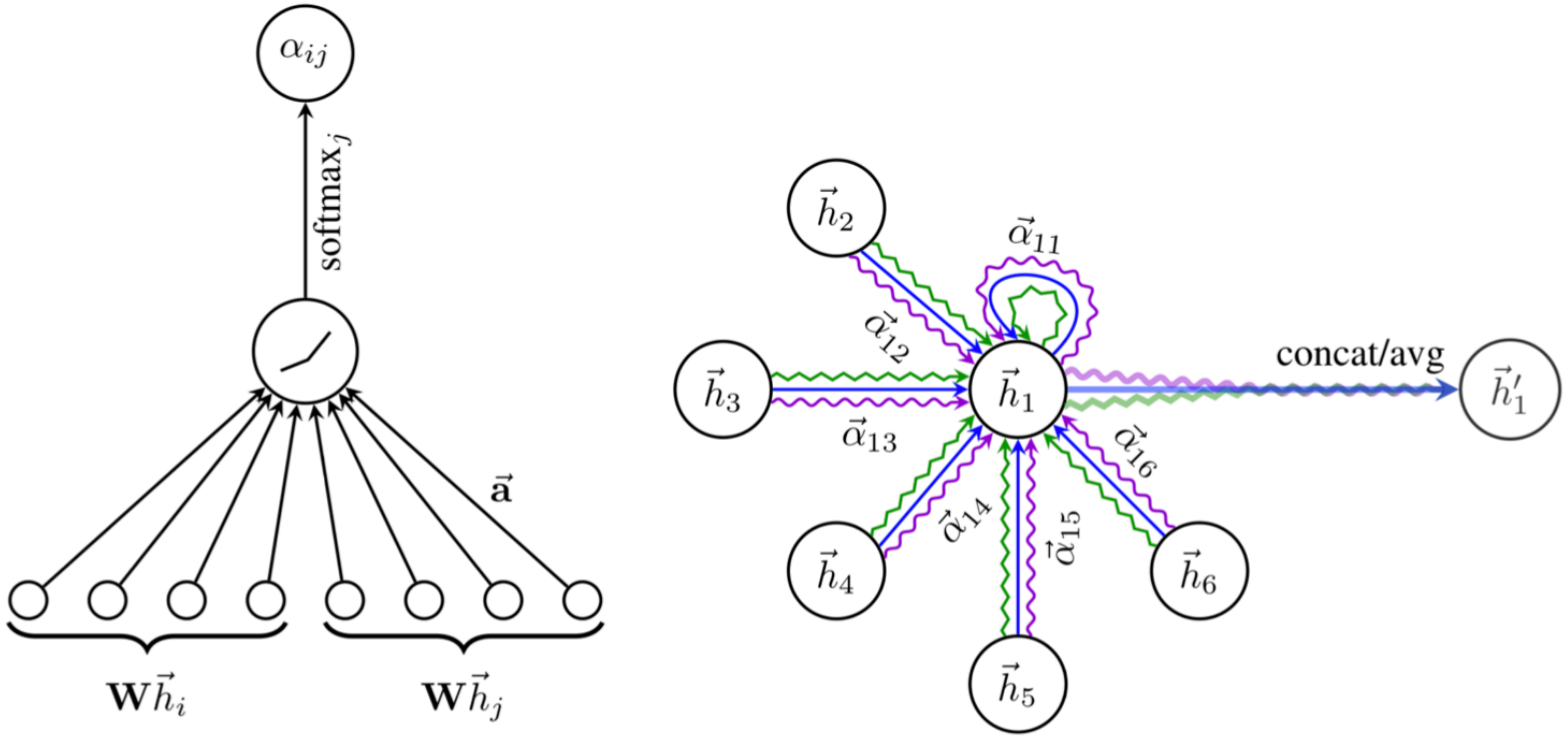
[Figure from Veličković et al. (ICLR 2018)]

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k\mathbf{W}^k\vec{h}_j\right) \qquad \alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i\|\mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i\|\mathbf{W}\vec{h}_k]\right)\right)}$$

* slide from Thomas Kipf, **University of Amsterdam**

Graph Neural Networks (GNNs) with **Attention**

Monti et al. (CVPR 2017), Hoshen (NIPS 2017), Veličković et al. (ICLR 2018)



[Figure from Veličković et al. (ICLR 2018)]

Pros:

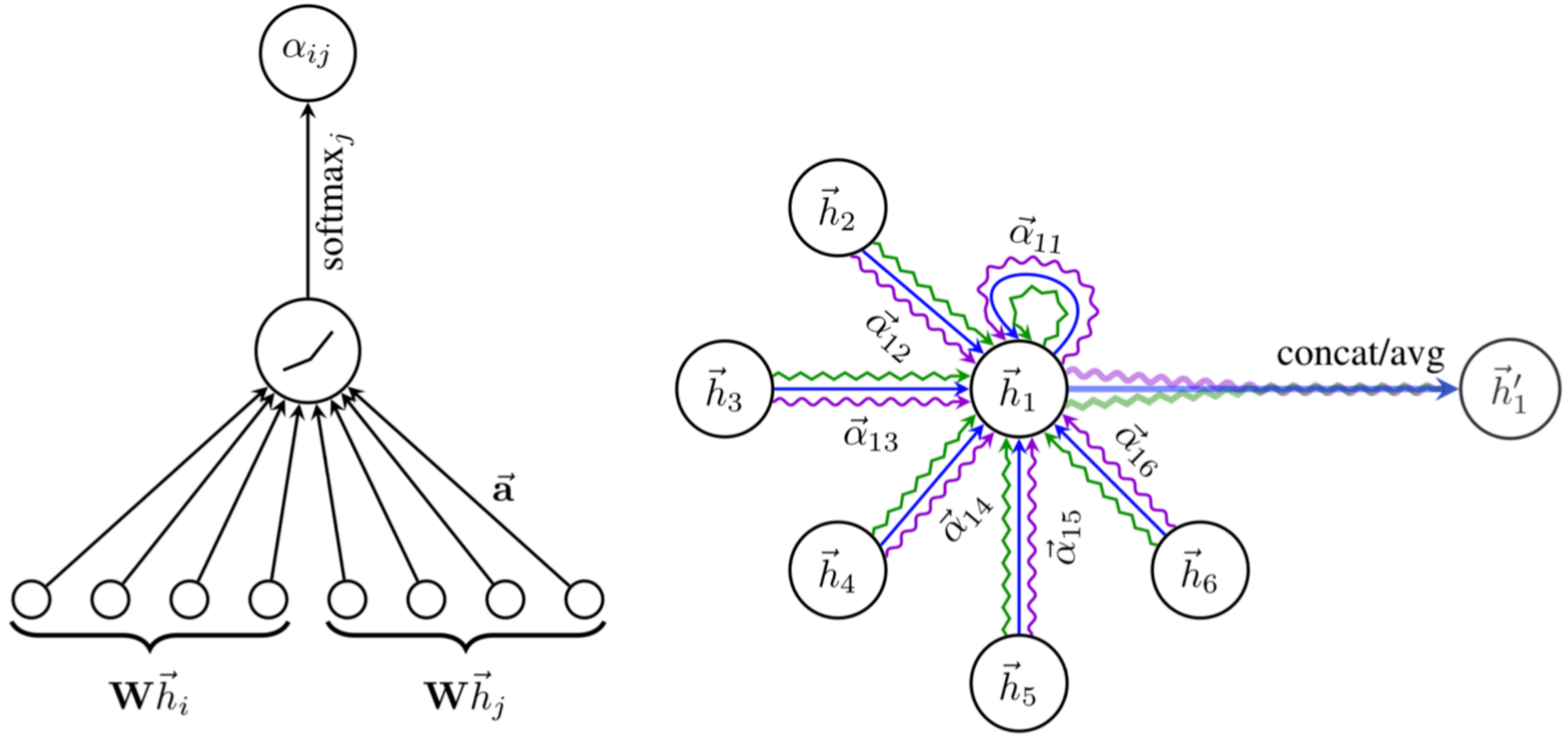
- No need to store intermediate edge-based activation vectors (when using dot-product attn.)
- Slower than GCNs but faster than GNNs with edge embeddings

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k\mathbf{W}^k\vec{h}_j\right) \qquad \alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i\|\mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i\|\mathbf{W}\vec{h}_k]\right)\right)}$$

* slide from Thomas Kipf, **University of Amsterdam**

Graph Neural Networks (GNNs) with **Attention**

Monti et al. (CVPR 2017), Hoshen (NIPS 2017), Veličković et al. (ICLR 2018)



[Figure from Veličković et al. (ICLR 2018)]

Pros:

- No need to store intermediate edge-based activation vectors (when using dot-product attn.)
- Slower than GCNs but faster than GNNs with edge embeddings

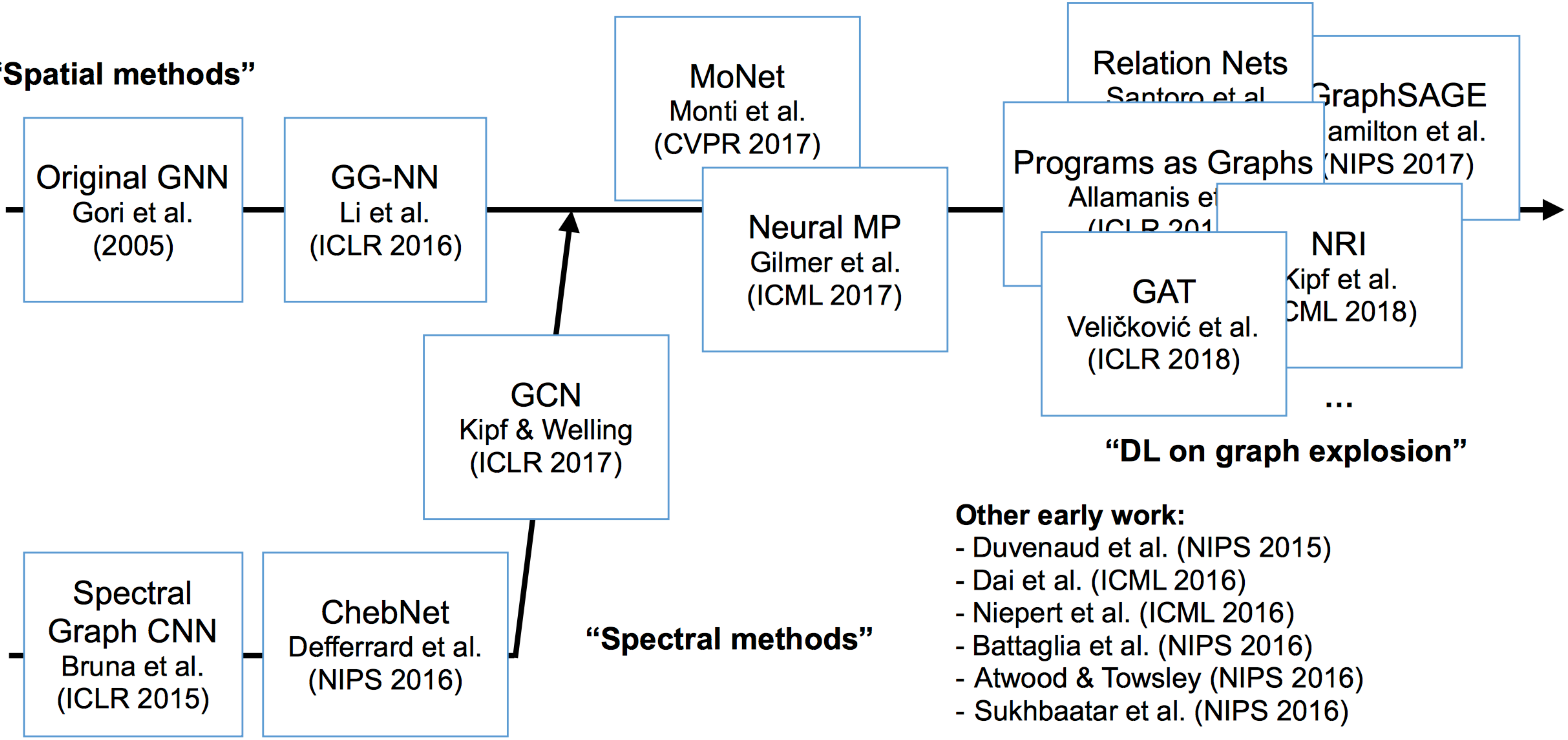
Cons:

- (Most likely) less expressive than GNNs with edge embeddings
- Can be more difficult to optimize

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k\mathbf{W}^k\vec{h}_j\right) \qquad \alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i\|\mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i\|\mathbf{W}\vec{h}_k]\right)\right)}$$

* slide from Thomas Kipf, **University of Amsterdam**

A **Brief History** of Graph Neural Nets



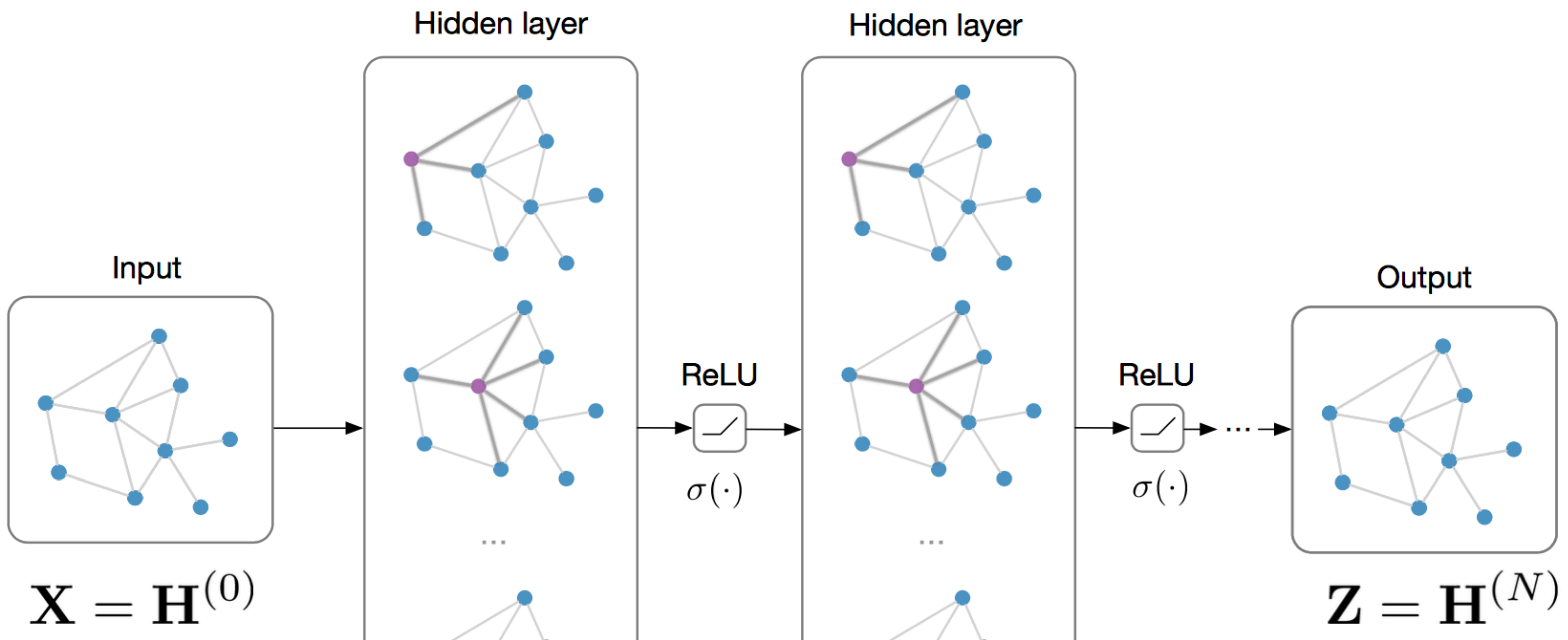
Other early work:
- Duvenaud et al. (NIPS 2015)
- Dai et al. (ICML 2016)
- Niepert et al. (ICML 2016)
- Battaglia et al. (NIPS 2016)
- Atwood & Towsley (NIPS 2016)
- Sukhbaatar et al. (NIPS 2016)

(slide inspired by Alexander Gaunt's talk on GNNs)

How do we use GNN / GCN for real problems?

Classification and **Link Prediction** with GNNs / GCNs

Input: Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times E}$, preprocessed adjacency matrix $\hat{\mathbf{A}}$

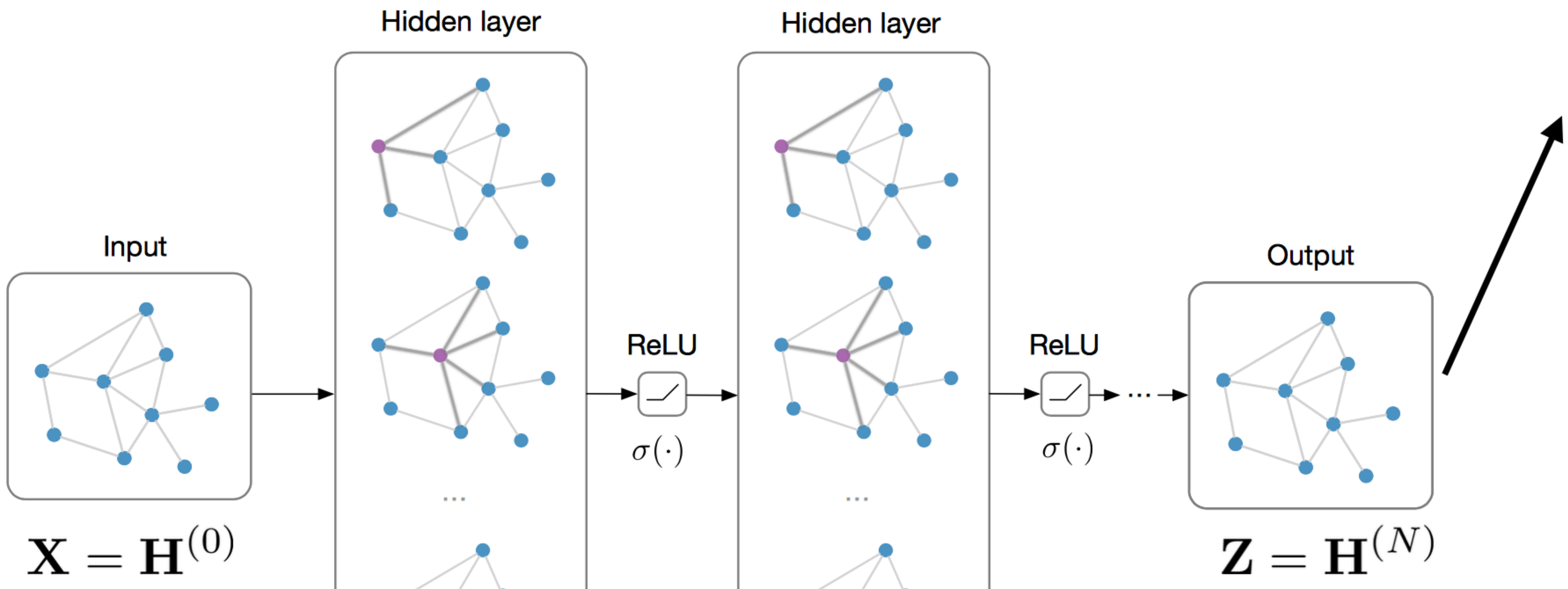


$$\mathbf{H}^{(l+1)} = \sigma\left(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right)$$

* slide from Thomas Kipf, **University of Amsterdam**

Classification and **Link Prediction** with GNNs / GCNs

Input: Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times E}$, preprocessed adjacency matrix $\hat{\mathbf{A}}$



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

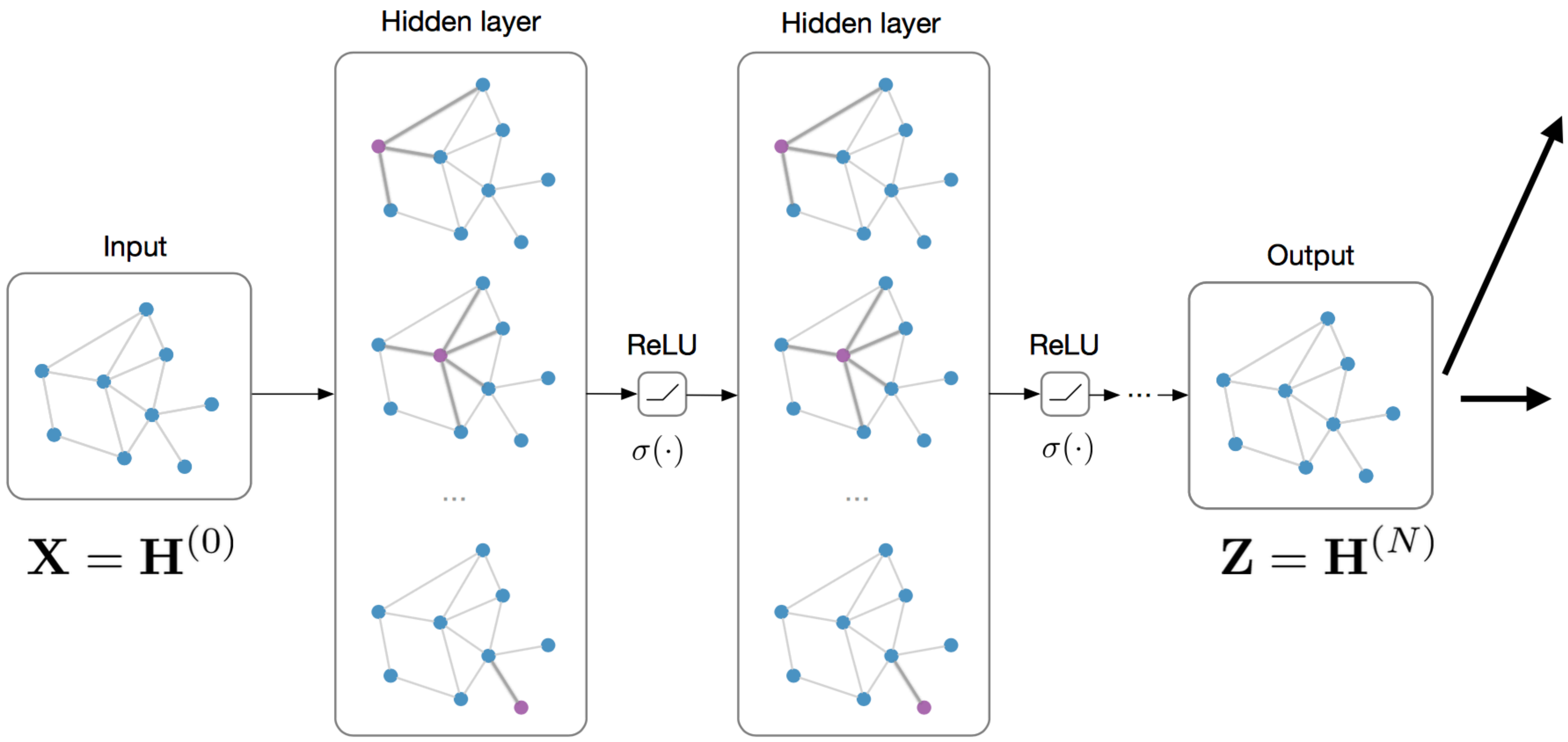
Node classification:

$$\mathrm{softmax}(\mathbf{z_n})$$

e.g. Kipf & Welling (ICLR 2017)

Classification and **Link Prediction** with GNNs / GCNs

Input: Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times E}$, preprocessed adjacency matrix $\hat{\mathbf{A}}$



$\mathbf{X} = \mathbf{H}^{(0)}$

$\mathbf{Z} = \mathbf{H}^{(N)}$

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

Node classification:

$\mathrm{softmax}(\mathbf{z_n})$

e.g. Kipf & Welling (ICLR 2017)

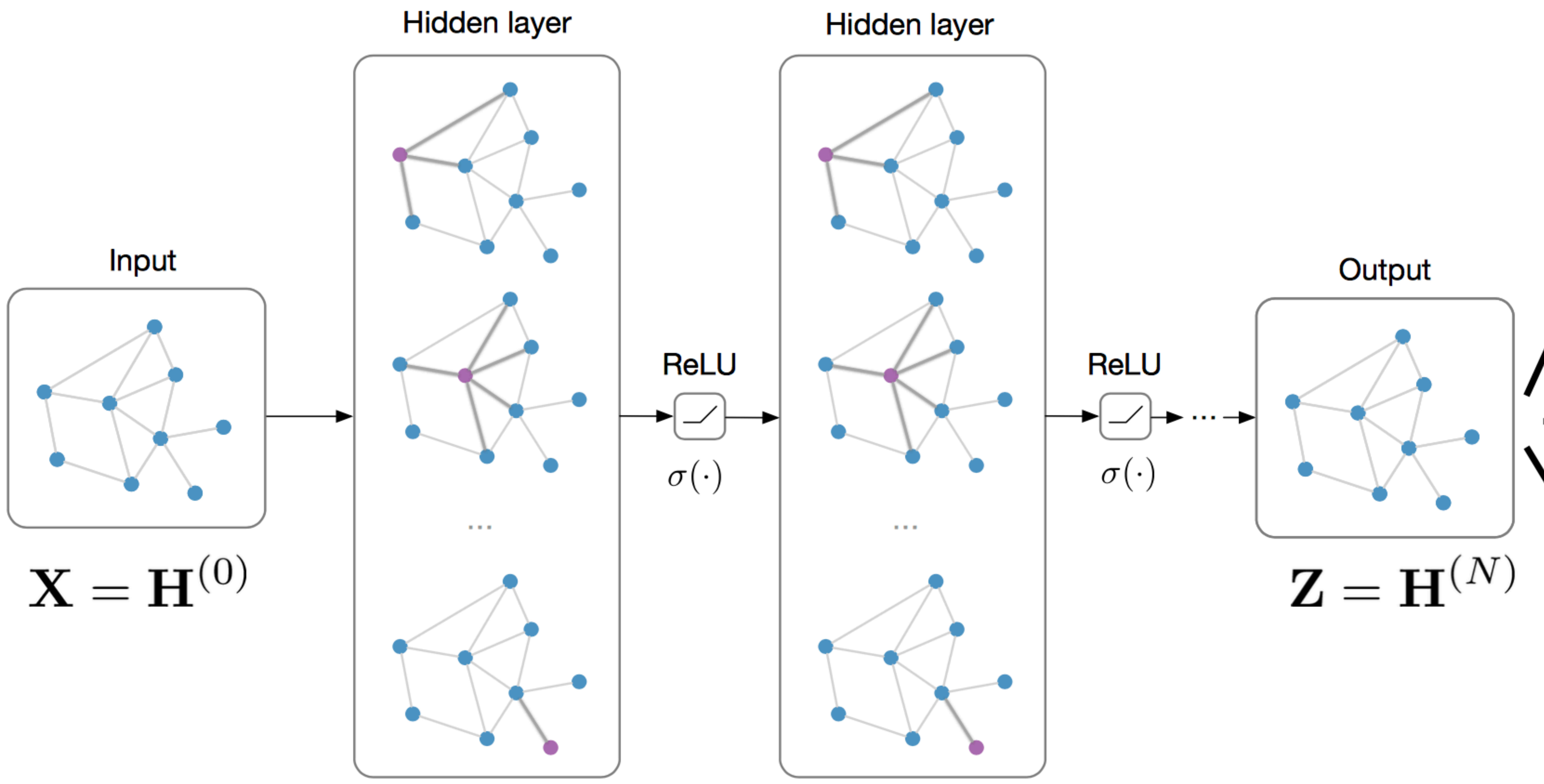
Graph classification:

$\mathrm{softmax}(\sum_n \mathbf{z_n})$

e.g. Duvenaud et al. (NIPS 2015)

Classification and **Link Prediction** with GNNs / GCNs

Input: Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times E}$, preprocessed adjacency matrix $\hat{\mathbf{A}}$



$$\mathbf{X} = \mathbf{H}^{(0)}$$

$$\mathbf{Z} = \mathbf{H}^{(N)}$$

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

Node classification:

$$\mathrm{softmax}(\mathbf{z_n})$$

e.g. Kipf & Welling (ICLR 2017)

Graph classification:

$$\mathrm{softmax}(\textstyle\sum_n \mathbf{z_n})$$

e.g. Duvenaud et al. (NIPS 2015)

Link prediction:

$$p(A_{ij}) = \sigma(\mathbf{z_i^T z_j})$$

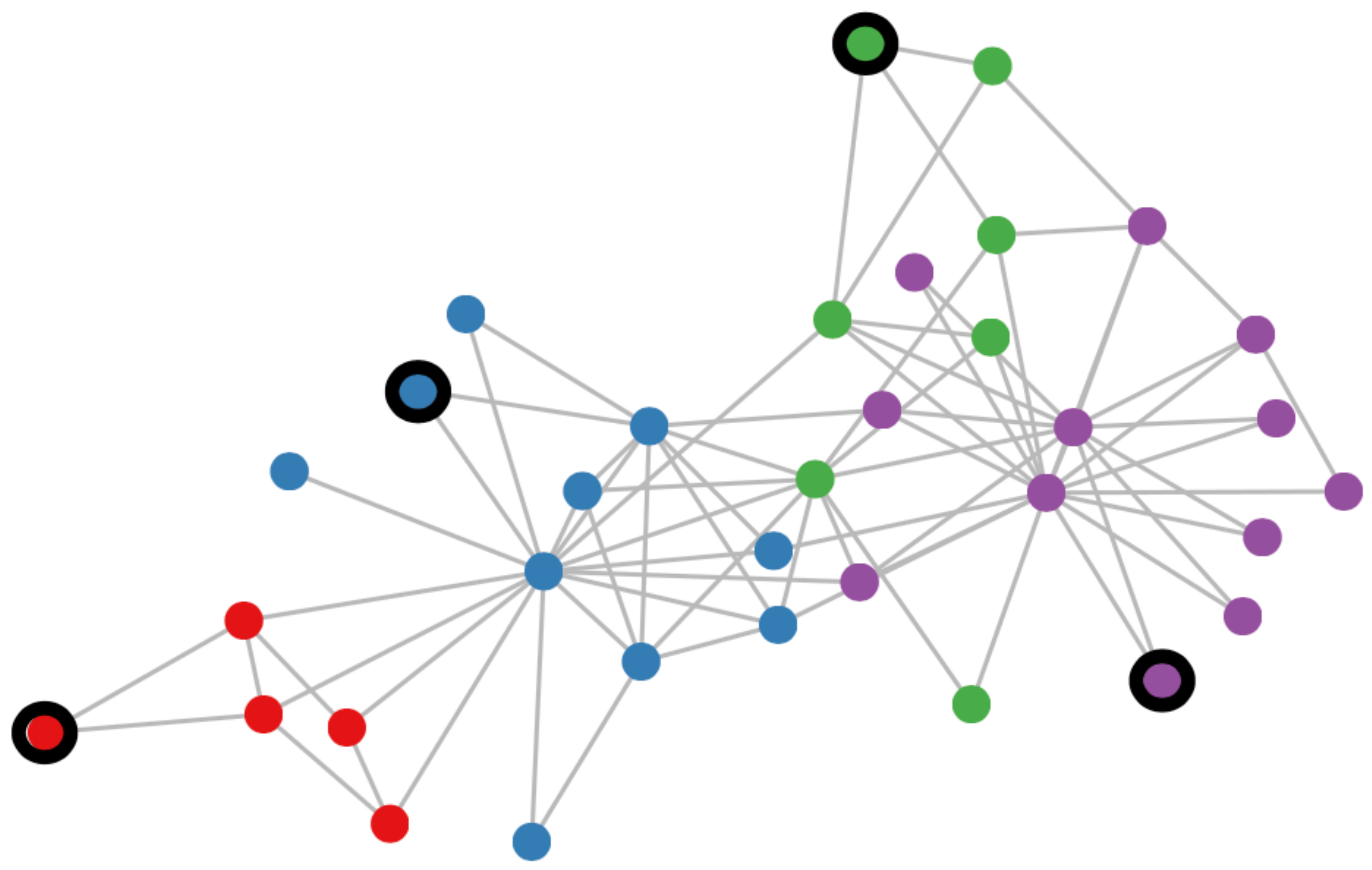
Kipf & Welling (NIPS BDL 2016)
"Graph Auto-Encoders"

Semi-supervised Classification on Graphs

Setting:

Some nodes are labeled (black circle)
All other nodes are unlabeled

Task:
Predict node label of unlabeled nodes



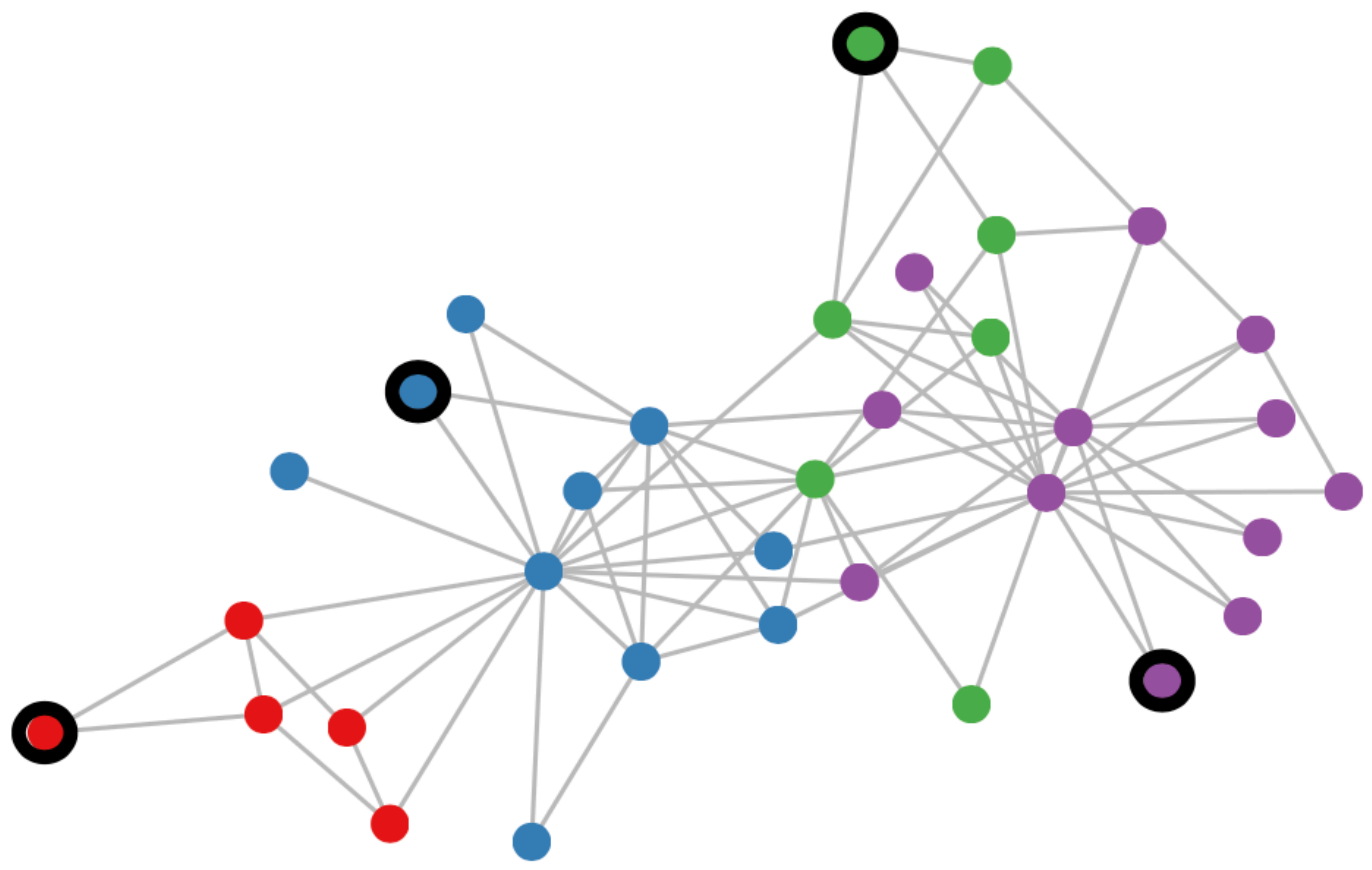
* slide from Thomas Kipf, **University of Amsterdam**

Semi-supervised Classification on Graphs

Setting:

Some nodes are labeled (black circle)
All other nodes are unlabeled

Task:
Predict node label of unlabeled nodes



Evaluate loss on labeled nodes only:

$$\mathcal{L} = -\sum_{l \in \mathcal{Y}_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$

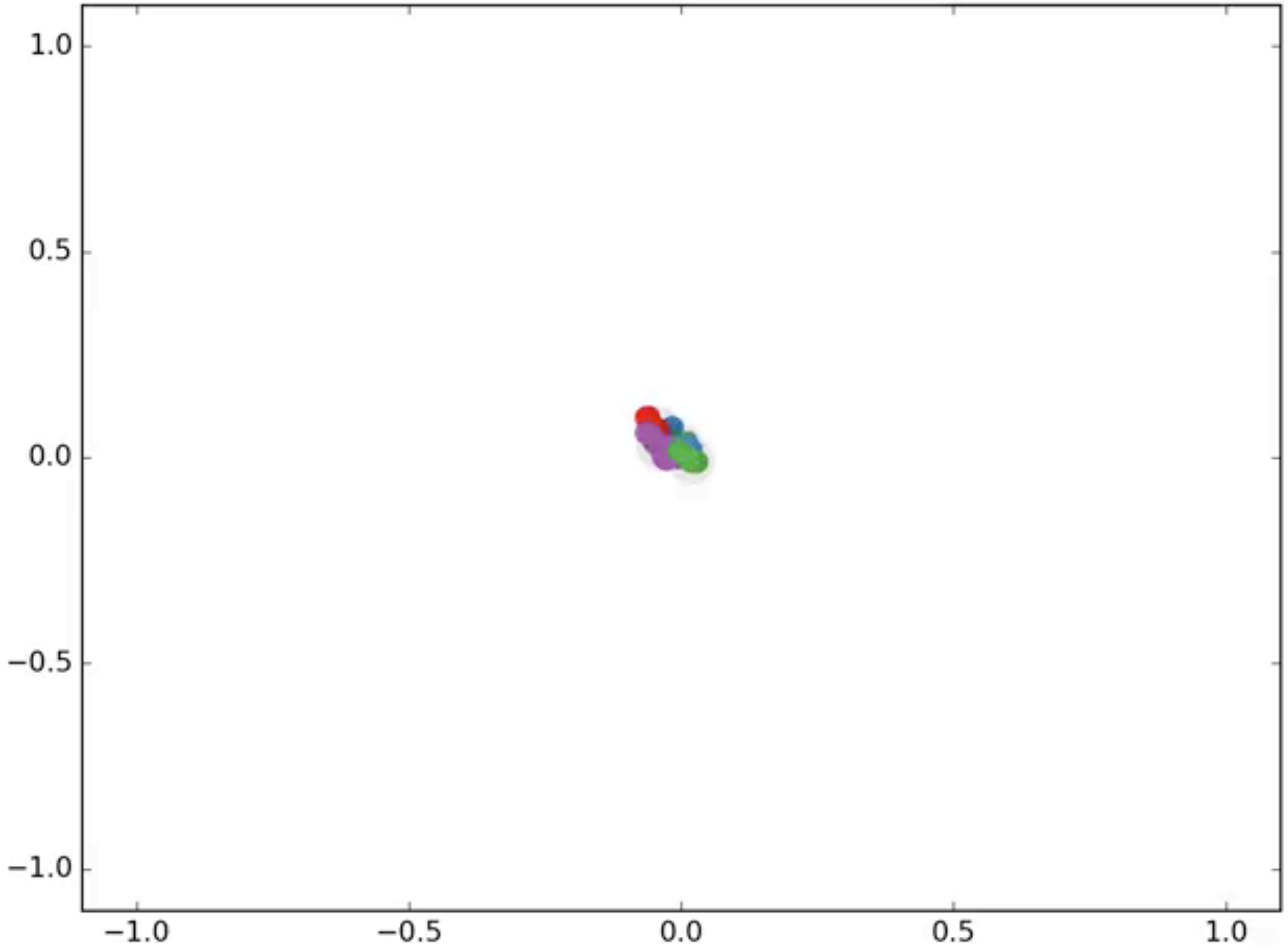
$\mathcal{Y}_L$ set of labeled node indices

$\mathbf{Y}$ label matrix

$\mathbf{Z}$ GCN output (after softmax)

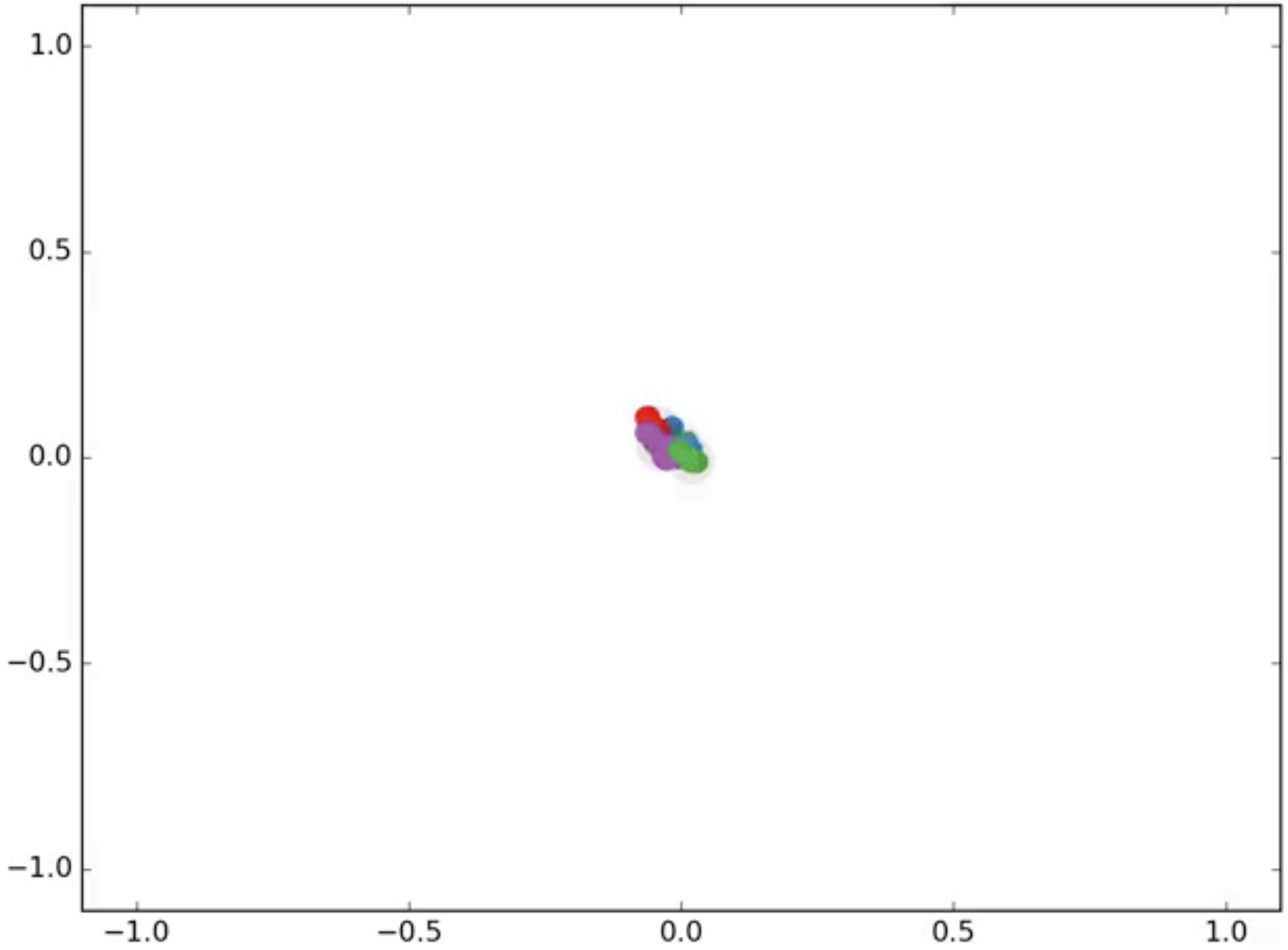
* slide from Thomas Kipf, **University of Amsterdam**

Semi-supervised Classification on Graphs



* slide from Thomas Kipf, **University of Amsterdam**

Semi-supervised Classification on Graphs

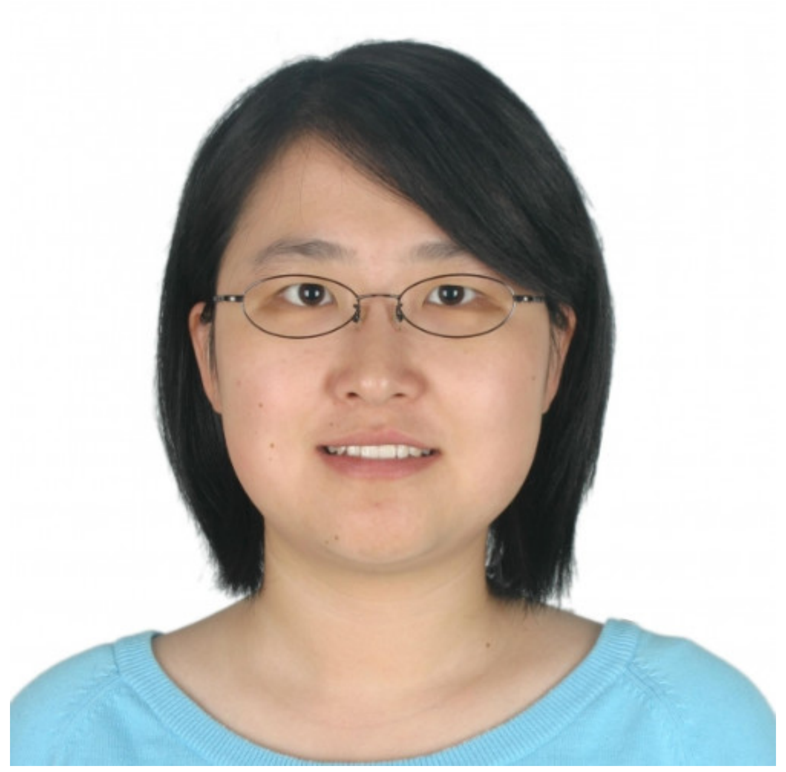


* slide from Thomas Kipf, **University of Amsterdam**

Graph Neural Nets (GNNs) are strict Generalizations of Traditional Neural Nets

(CNNs / RNNs can be implemented using GNNs / GCNs, but this is inefficient)

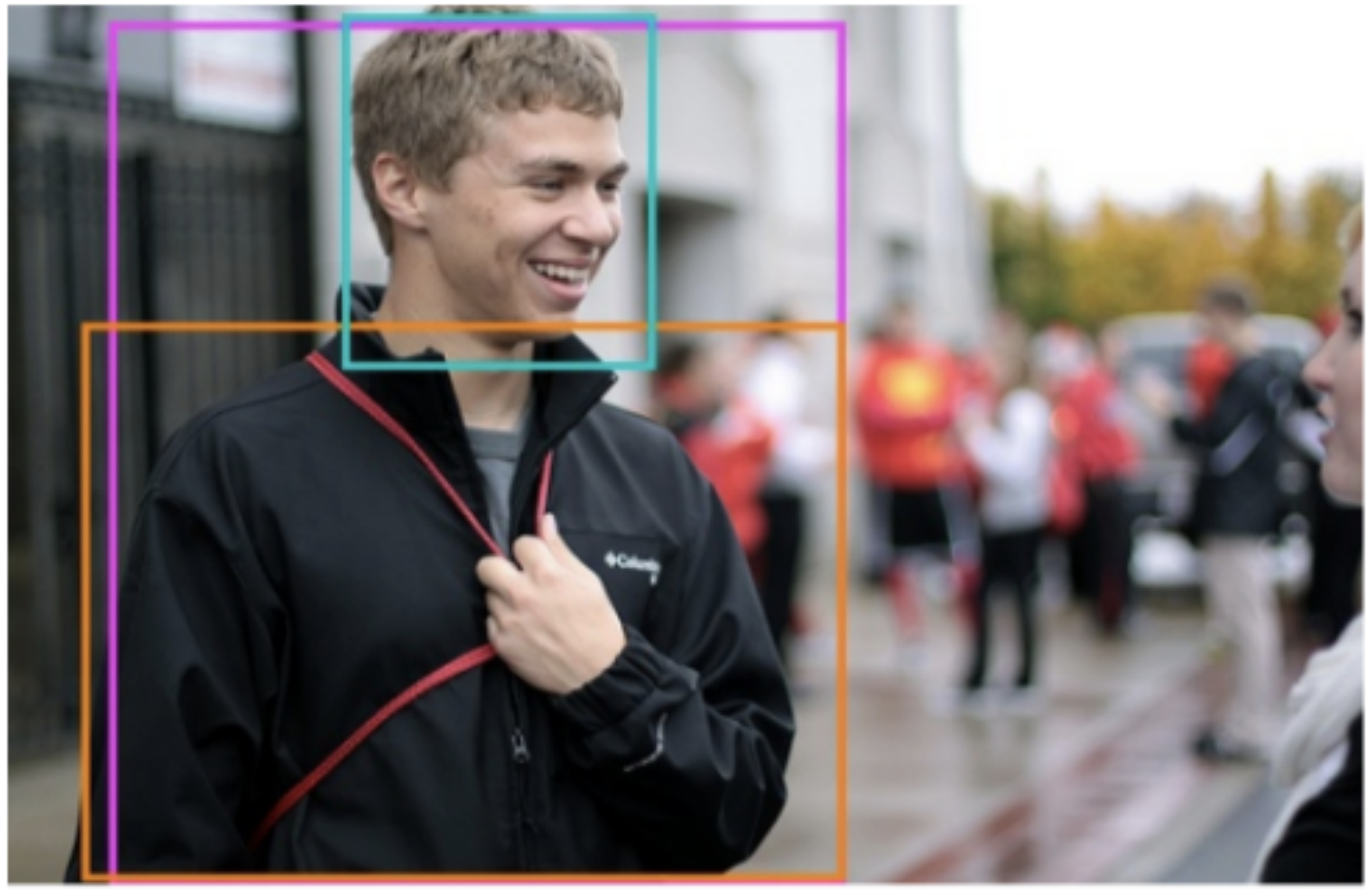
G^{3}raphGround: Graph-based Language Grounding



Mohit Bajaj Lanjun Wang Leonid Sigal

Image Grounding: Beyond Object Detection

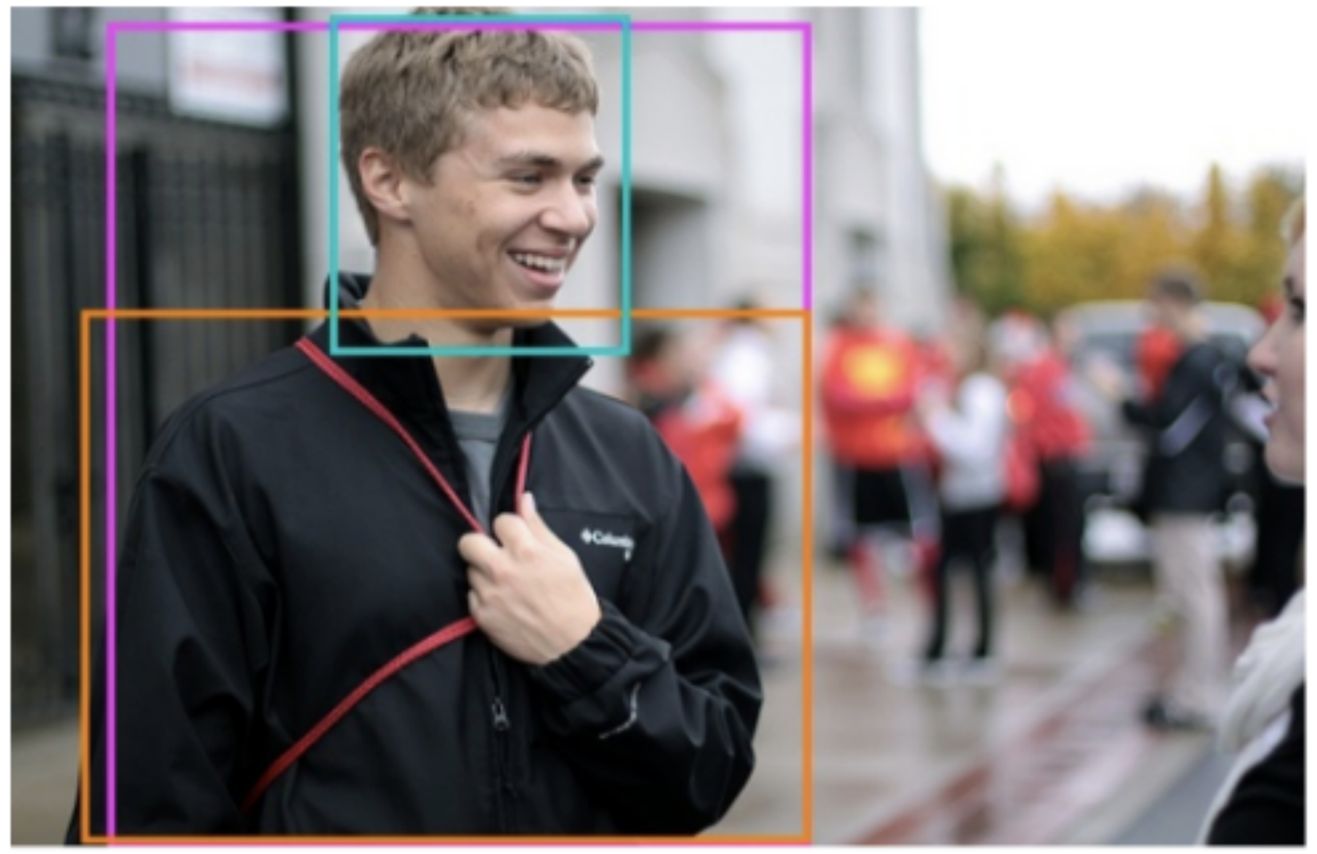
Given the **image** and one or more **natural language phrases**, locate regions that correspond to those phrases.



A man wearing a black-jacket has a smile on his face.

Image Grounding: Beyond Object Detection

Given the **image** and one or more **natural language phrases**, locate regions that correspond to those phrases.



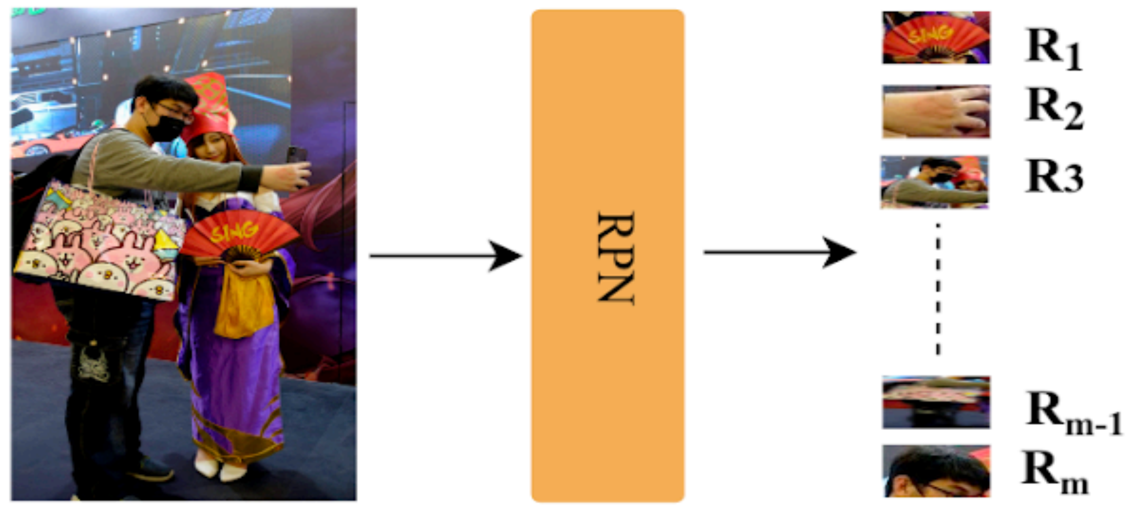
A man wearing a black-jacket has a smile on his face.

Fundamental task for **image / video understanding**

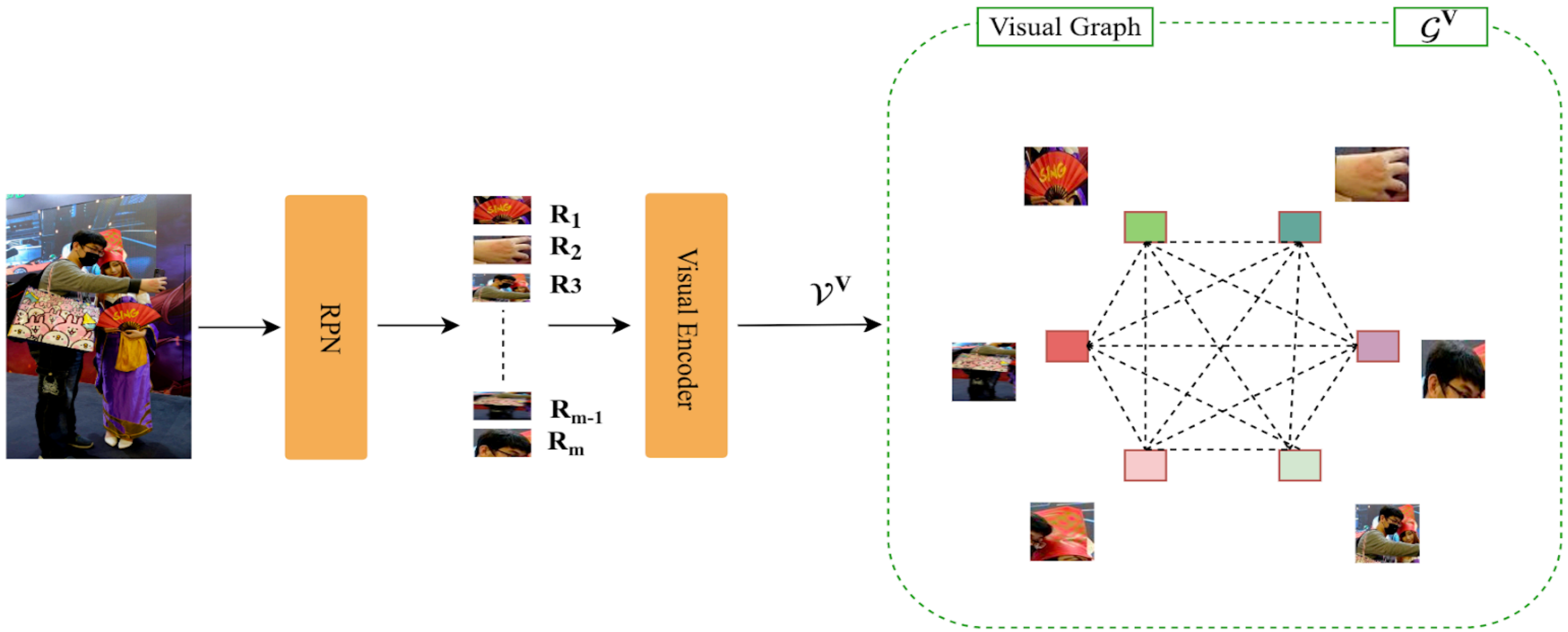
— Helps improve performance on other tasks (e.g., image captioning, VQA)

Proposed **Architecture**

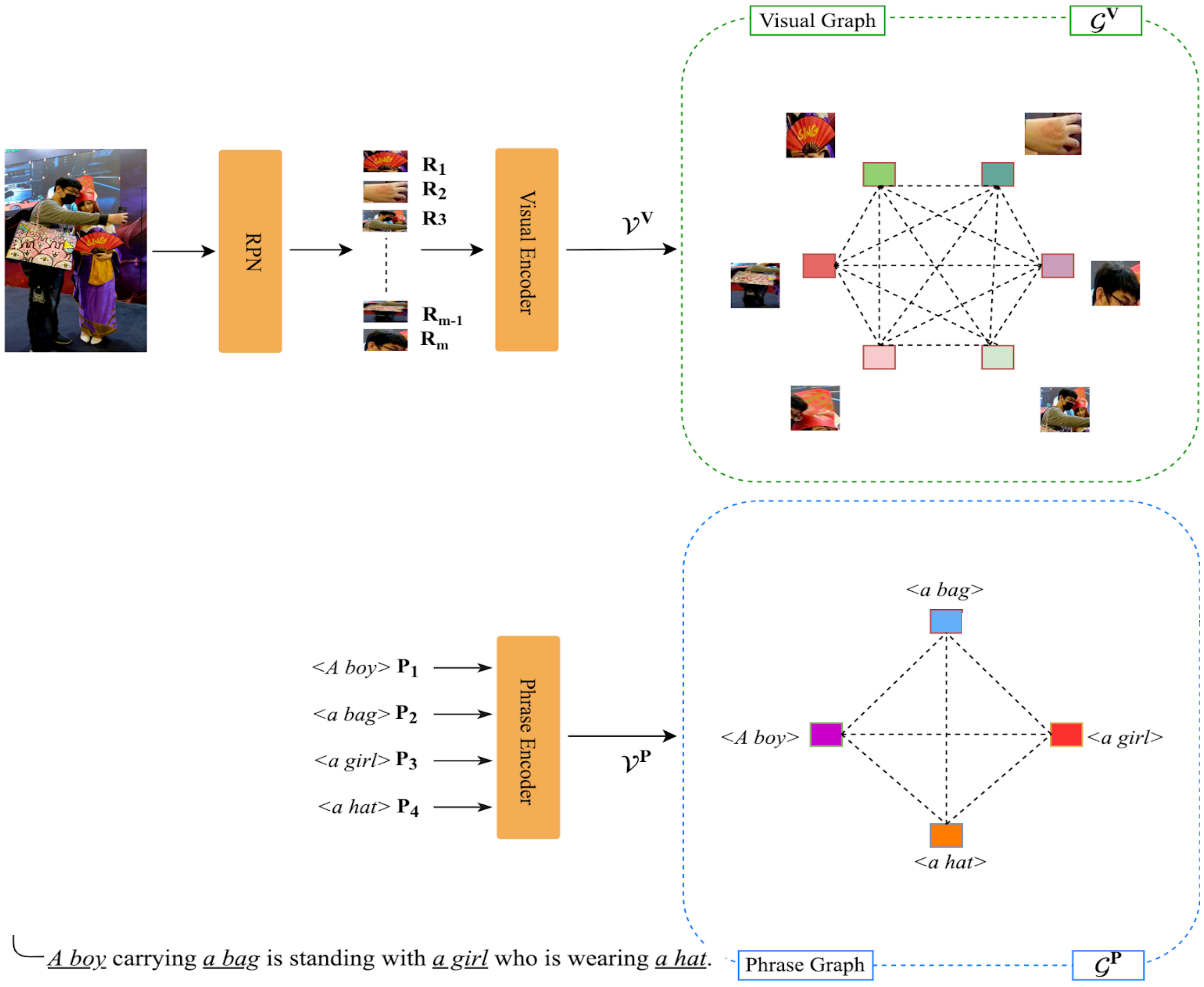
Proposed **Architecture**



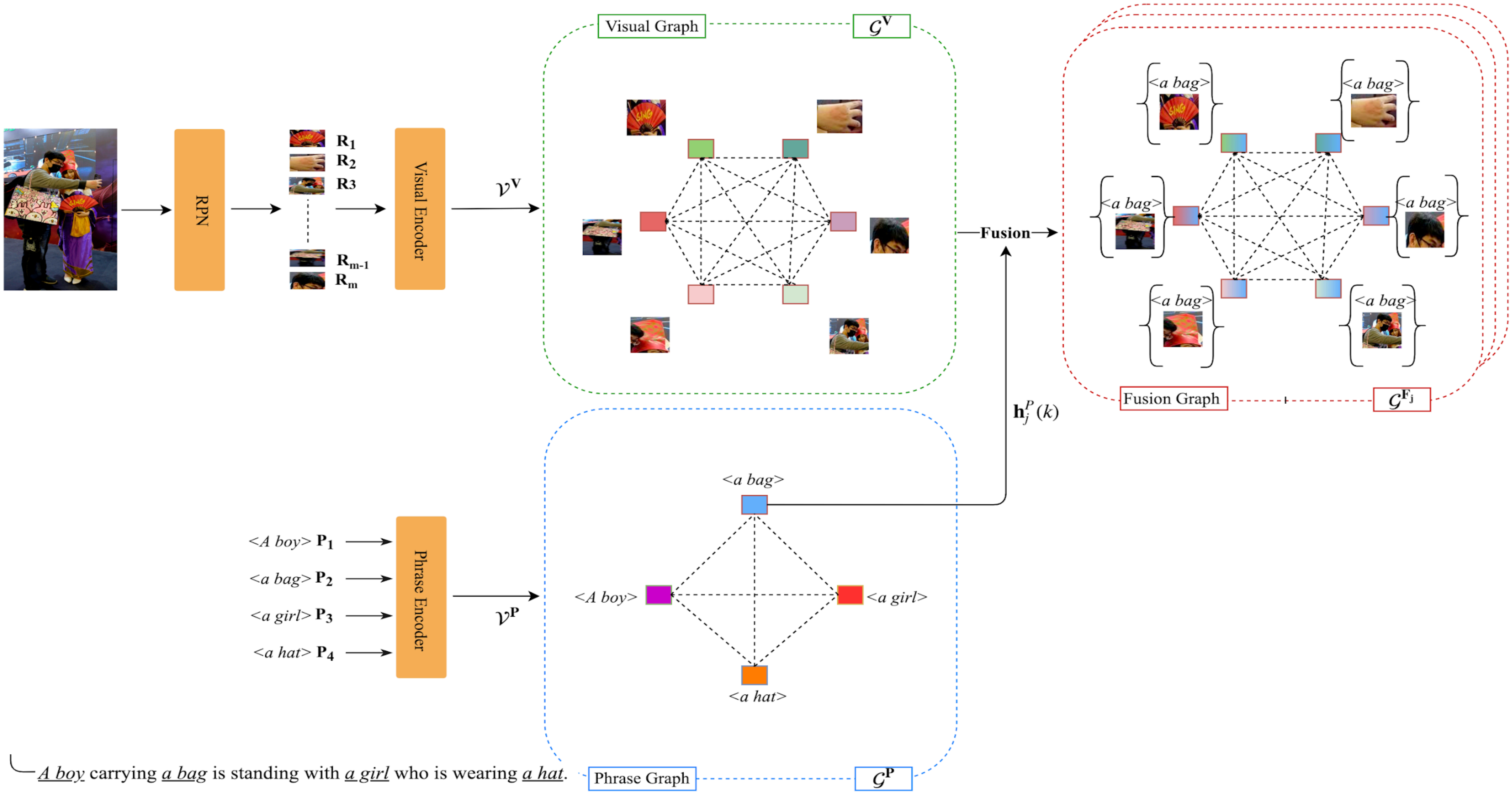
Proposed **Architecture**



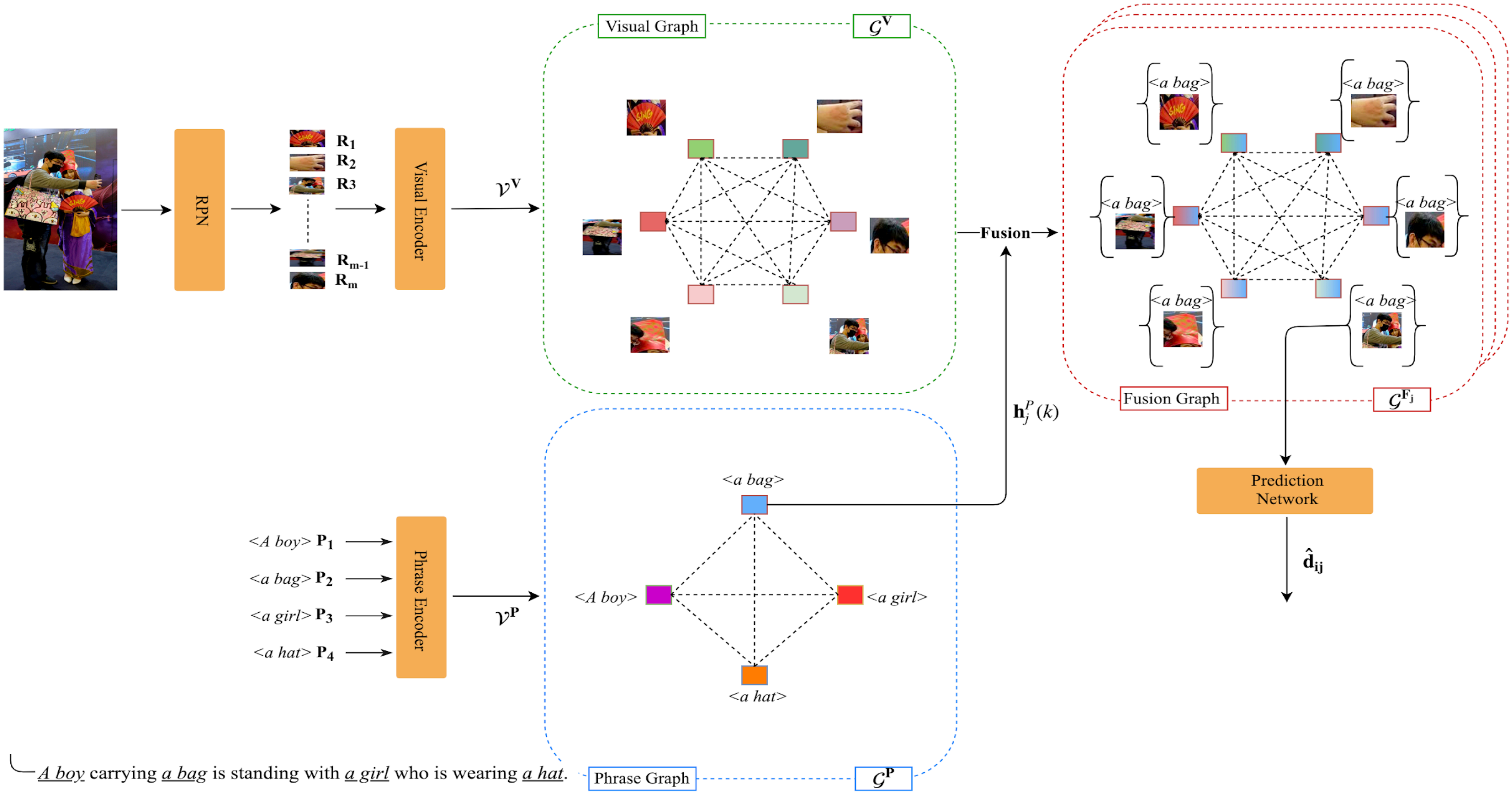
Proposed **Architecture**



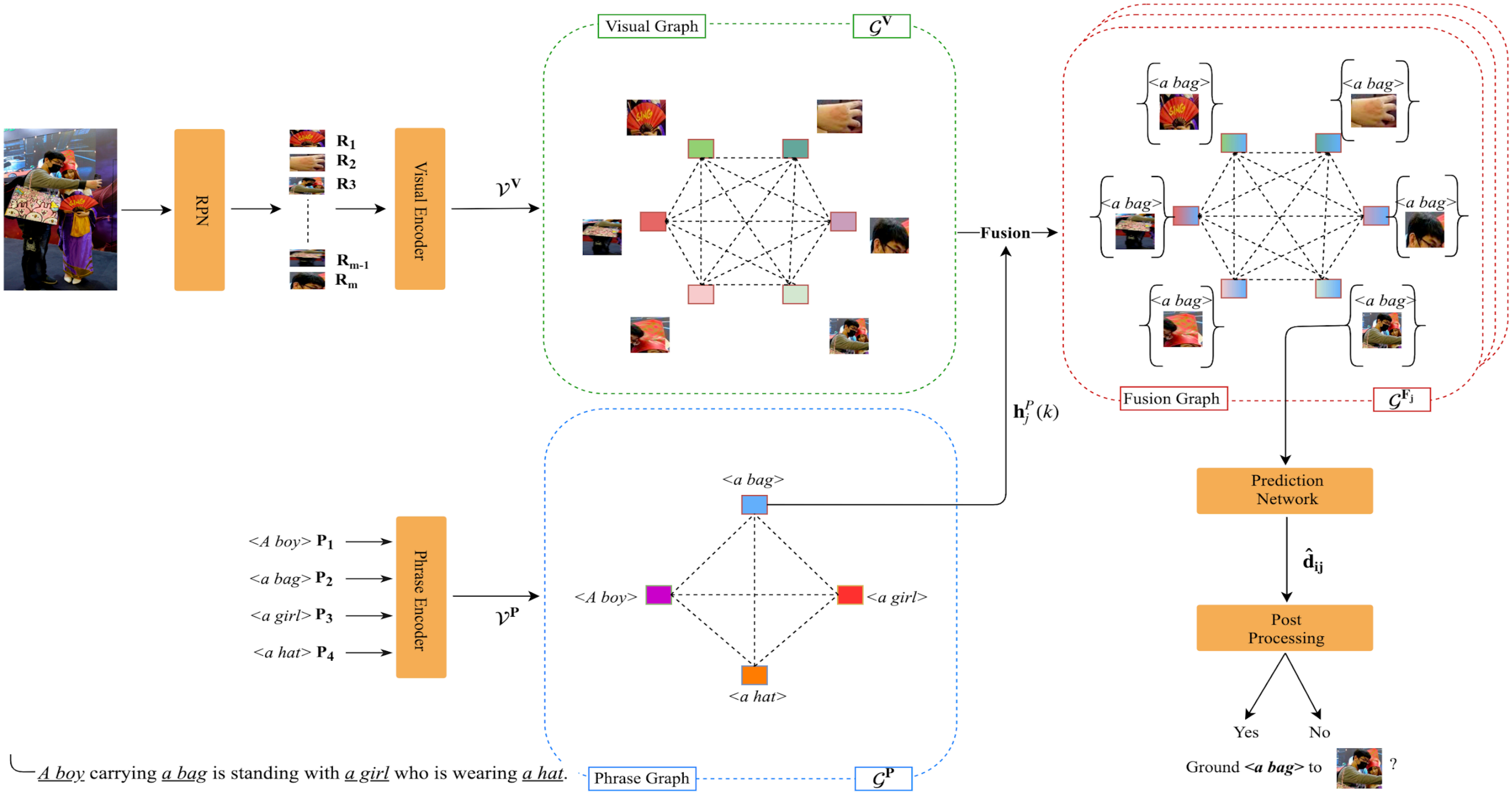
Proposed **Architecture**



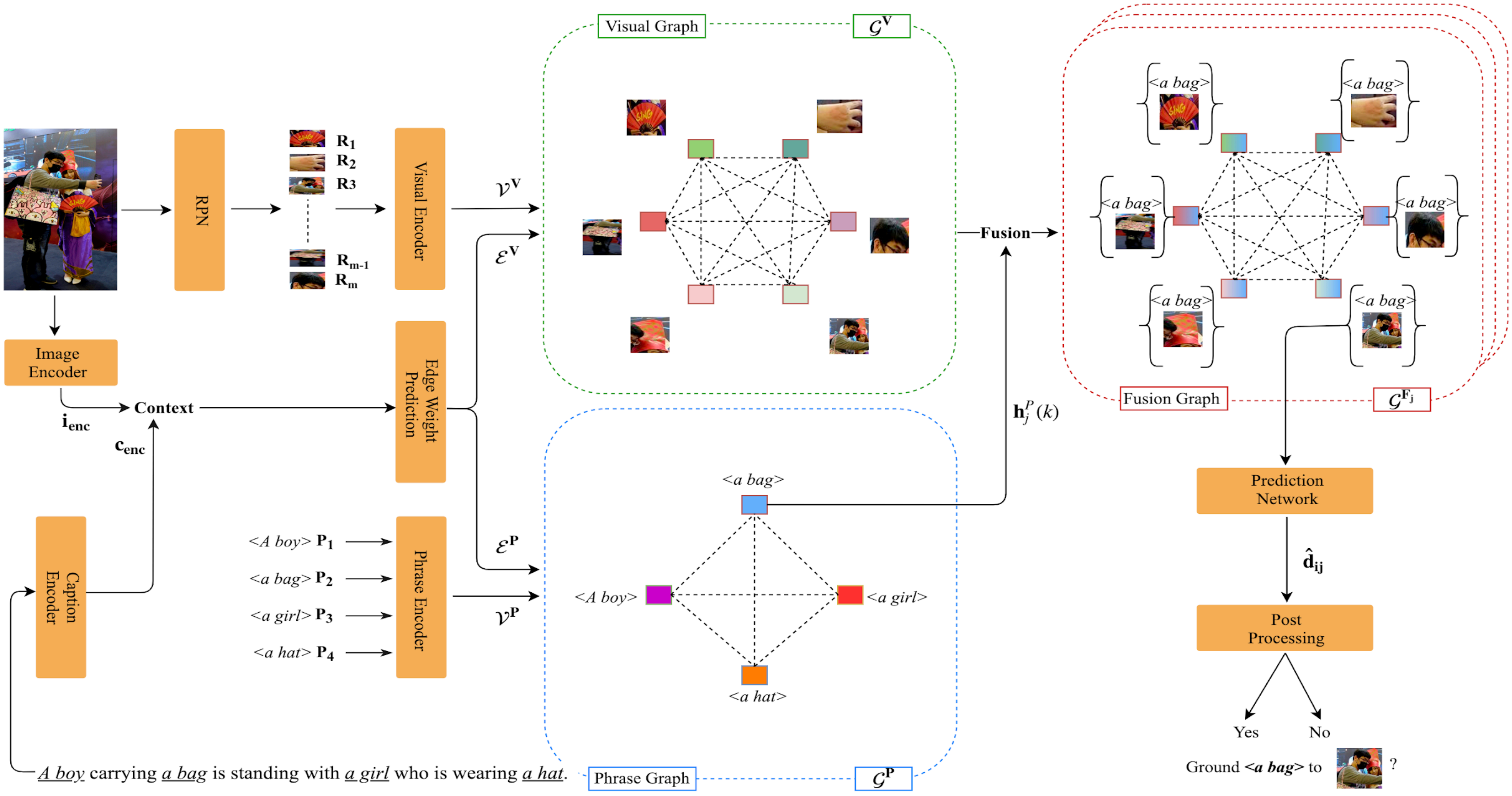
Proposed **Architecture**



Proposed **Architecture**



Proposed **Architecture**



Experiments

Datasets

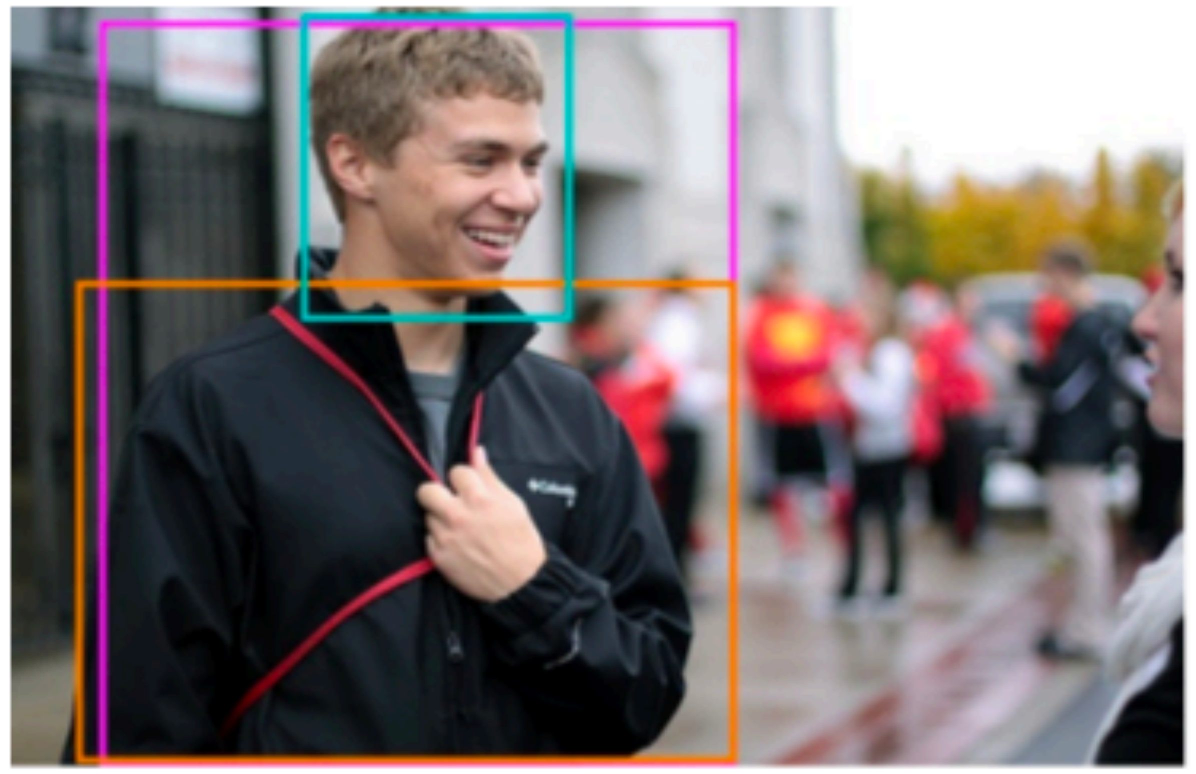
— **Flickr30K Entities**: (mostly noun) Phrases parsed from image captions

— **ReferIt Game**: Unambiguous single phrases

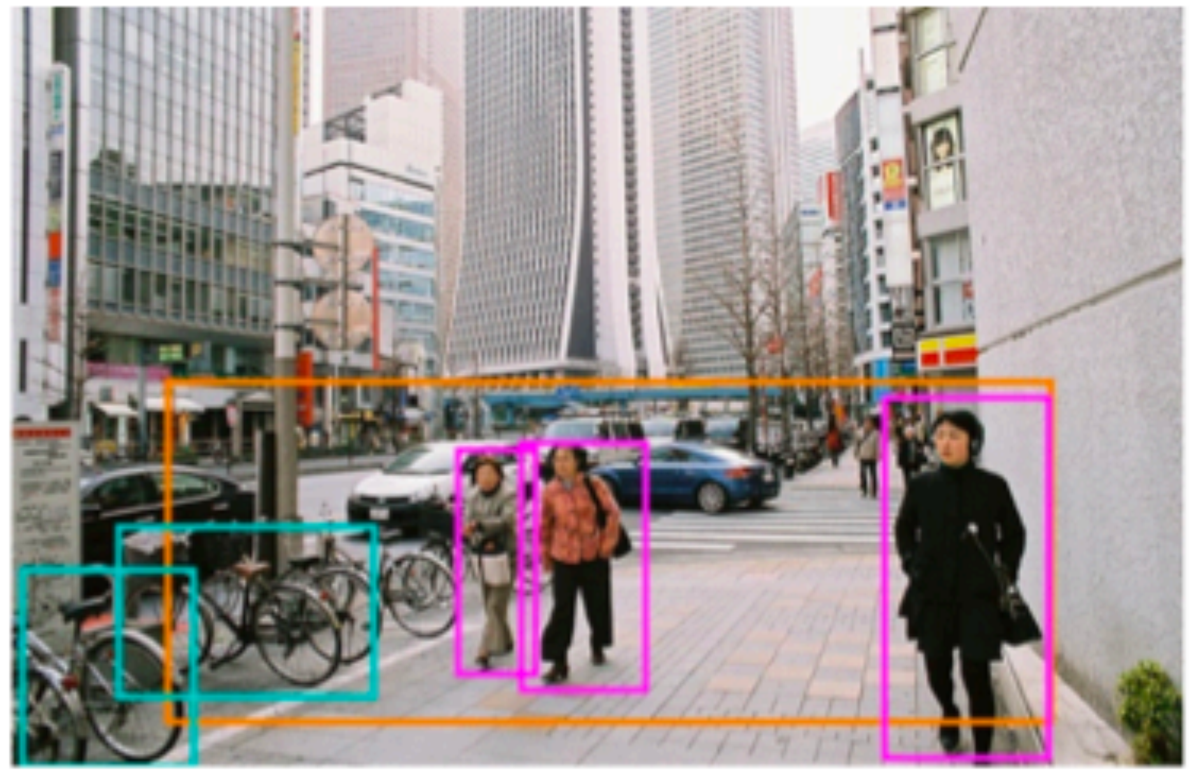
Evaluation

— Ratio of correctly grounded phrases to the total phrases

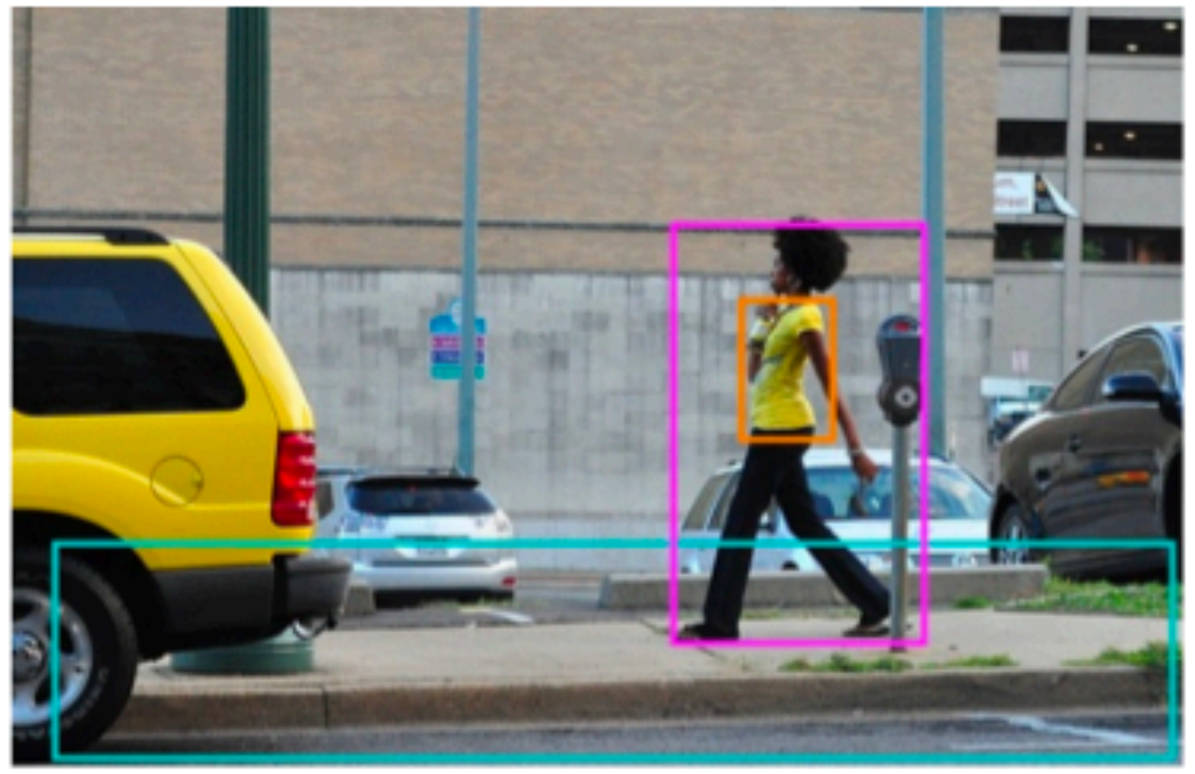
Qualitative Results: Flickr30K



(a) A man wearing a black-jacket has a smile on his face.



(b) People are walking on the street , with bikes parked up to the left of the picture.

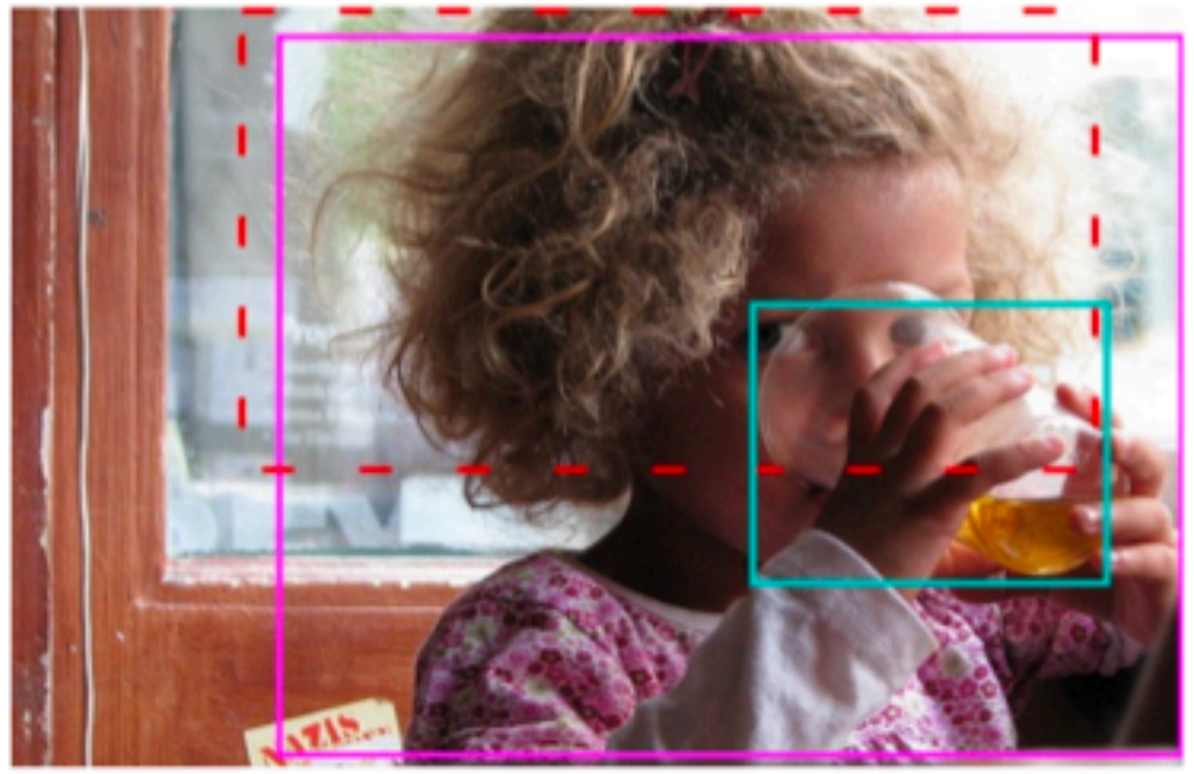


(c) A woman in a yellow shirt is walking down the sidewalk.

(d) A young boy is walking on wooden path in the middle of trees.

(e) Two women in colorful clothing are dancing inside a circle of other women.

(f) Lady wearing white shirt with blue umbrella in the rain.



(g) Young girl with curly hair is drinking out of a plastic cup.

(h) The bearded man keeps his blue Bic pen in hand while he plays the guitar.

Quantitative Results

Flickr30k Entities:

Method	Accuracy
SMPL [27]	42.08
NonlinearSP [26]	43.89
GroundeR [23]	47.81
MCB [7]	48.69
RtP [21]	50.89
Similarity Network [25]	51.05
IGOP [34]	53.97
SPC+PPC [20]	55.49
SS+QRN (VGGdet) [4]	55.99
CITE [19]	59.27
SeqGROUND	61.60
CITE [19] (finetuned)	61.89
QRC Net [4] (finetuned)	65.14
G^{3}RAPHGROUND++	**66.67**

Quantitative Results

Flickr30k Entities:

Method	Accuracy
SMPL [27]	42.08
NonlinearSP [26]	43.89
GroundeR [23]	47.81
MCB [7]	48.69
RtP [21]	50.89
Similarity Network [25]	51.05
IGOP [34]	53.97
SPC+PPC [20]	55.49
SS+QRN (VGGdet) [4]	55.99
CITE [19]	59.27
SeqGROUND	61.60
CITE [19] (finetuned)	61.89
QRC Net [4] (finetuned)	65.14
G^{3}RAPHGROUND++	**66.67**

Quantitative Results

Flickr30k Entities:

Method	Accuracy
SMPL [27]	42.08
NonlinearSP [26]	43.89
GroundeR [23]	47.81
MCB [7]	48.69
RtP [21]	50.89
Similarity Network [25]	51.05
IGOP [34]	53.97
SPC+PPC [20]	55.49
SS+QRN (VGGdet) [4]	55.99
CITE [19]	59.27
SeqGROUND	61.60
CITE [19] (finetuned)	61.89
QRC Net [4] (finetuned)	65.14
G^3RAPHGROUND++	**66.67**

ReferIt Game:

Method	Accuracy
SCRC [9]	17.93
MCB + Reg + Spatial [3]	26.54
GroundeR + Spatial [23]	26.93
Similarity Network + Spatial [25]	31.26
CGRE [17]	31.85
MNN + Reg + Spatial [3]	32.21
EB+QRN (VGGcls-SPAT) [4]	32.21
CITE [19]	34.13
IGOP [34]	34.70
QRC Net [4] (finetuned)	44.07
G^3RAPHGROUND++	**44.91**

Ablation

Method	Flickr30k	ReferIt
GG - VisualG - FusionG	56.32	32.89
GG - VisualG	62.23	38.82
GG - FusionG	59.13	36.54
GG - PhraseG	60.82	38.12
GGFusionBase	60.41	38.65
GG - ImageContext	62.32	40.92
GG - PhraseContext	62.73	*n.a.*
G^3RAPHGROUND (GG)	63.65	41.79
G^3RAPHGROUND++	**66.67**	**44.91**

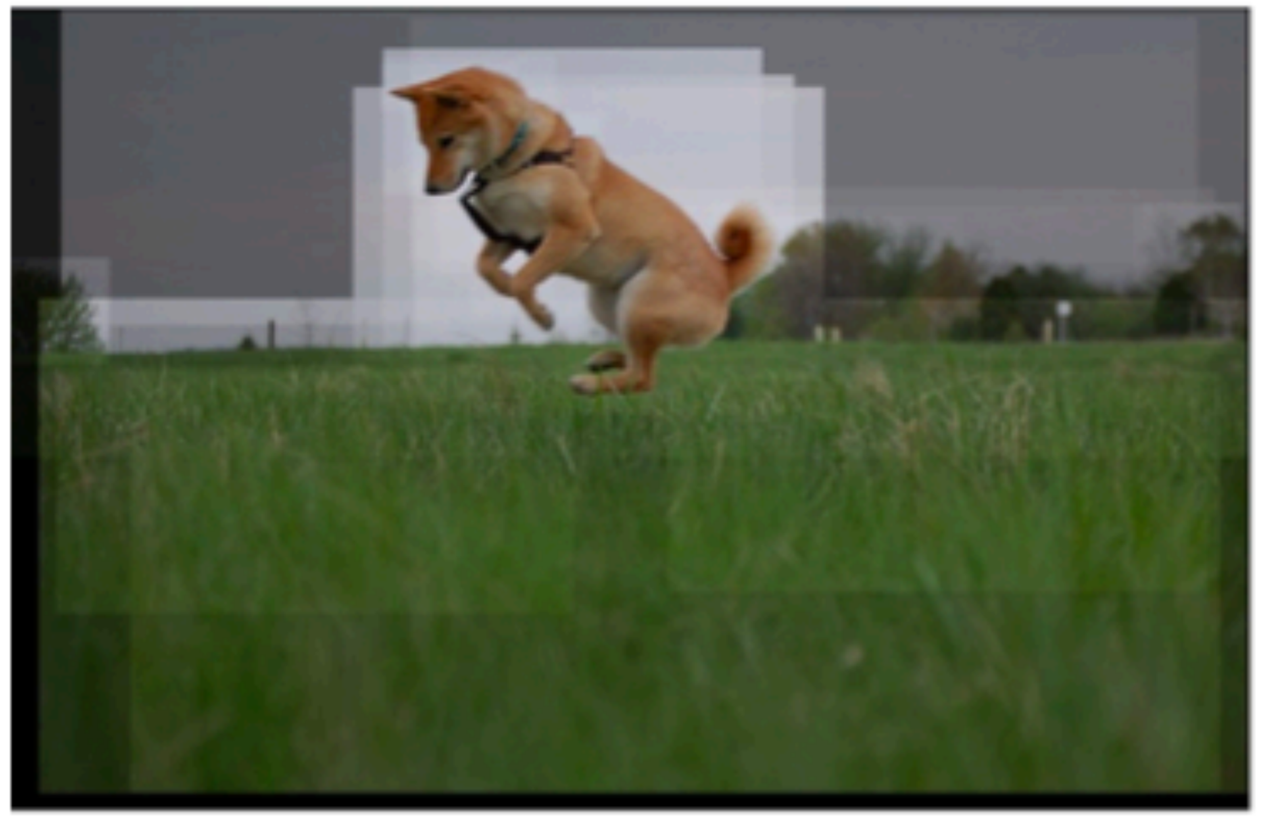
Ablation

Method	Flickr30k	ReferIt
GG - VisualG - FusionG	56.32	32.89
GG - VisualG	62.23	38.82
GG - FusionG	59.13	36.54
GG - PhraseG	60.82	38.12
GGFusionBase	60.41	38.65
GG - ImageContext	62.32	40.92
GG - PhraseContext	62.73	*n.a.*
G^{3}RAPHGROUND (GG)	63.65	41.79
G^{3}RAPHGROUND++	**66.67**	**44.91**

Visualizing Graph Attention

(a) A young boy is looking at a man painted in all gold.

(b) A man is checking his blue sneakers next to two men having a conversation.



(c) A brown dog jumps high on a field of grass.

(d) A woman stands in a field near a car and looks through binoculars.

THE UNIVERSITY OF BRITISH COLUMBIA

Energy-Based Learning for Scene Graph Generation

Mohammed Suhail

+ + +

Scene Graphs:

A **graph** based data structure for semantically representing image content

Scene Graphs

Scene Graphs

Scene Graphs

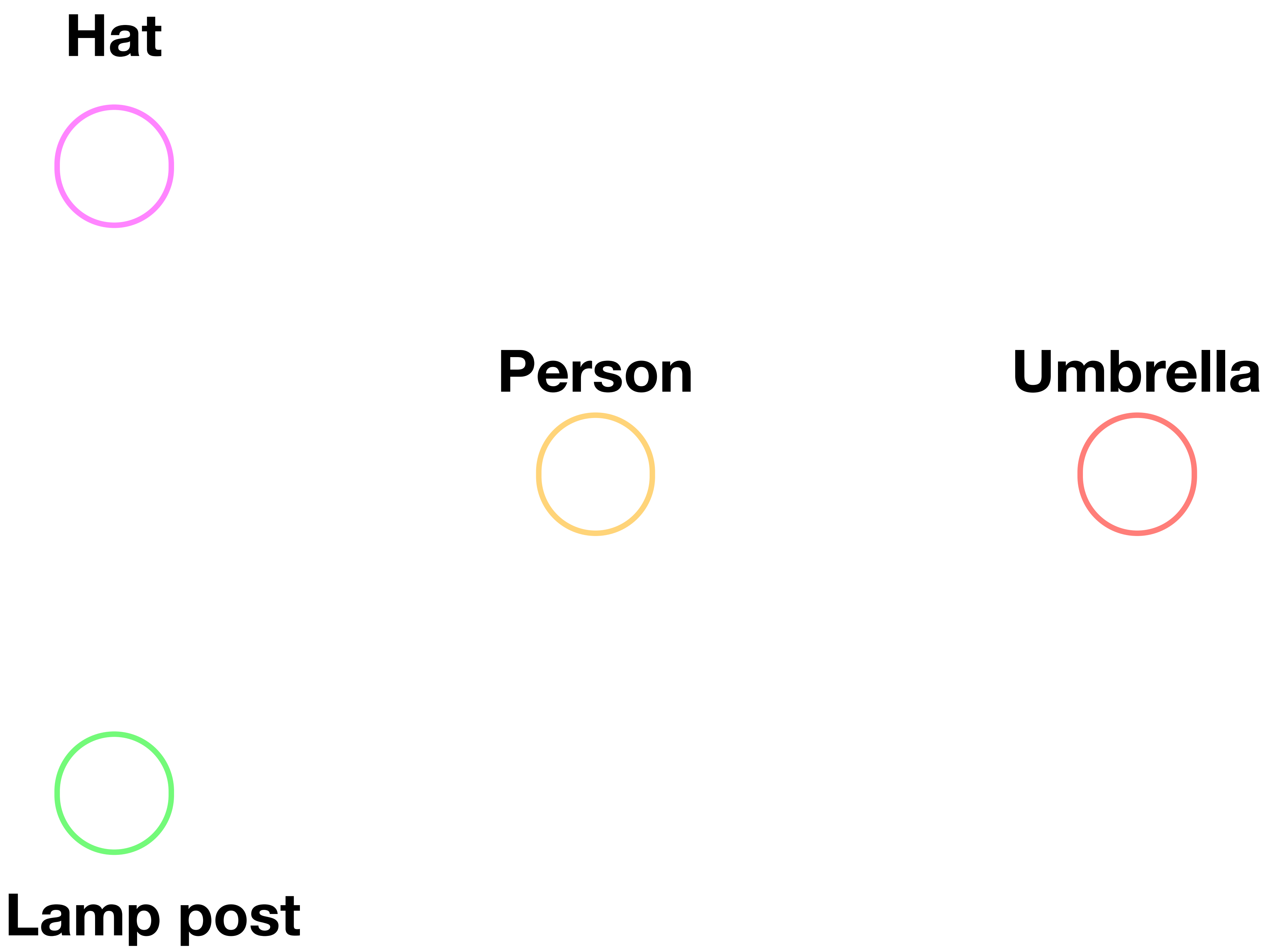
Hat

Umbrella

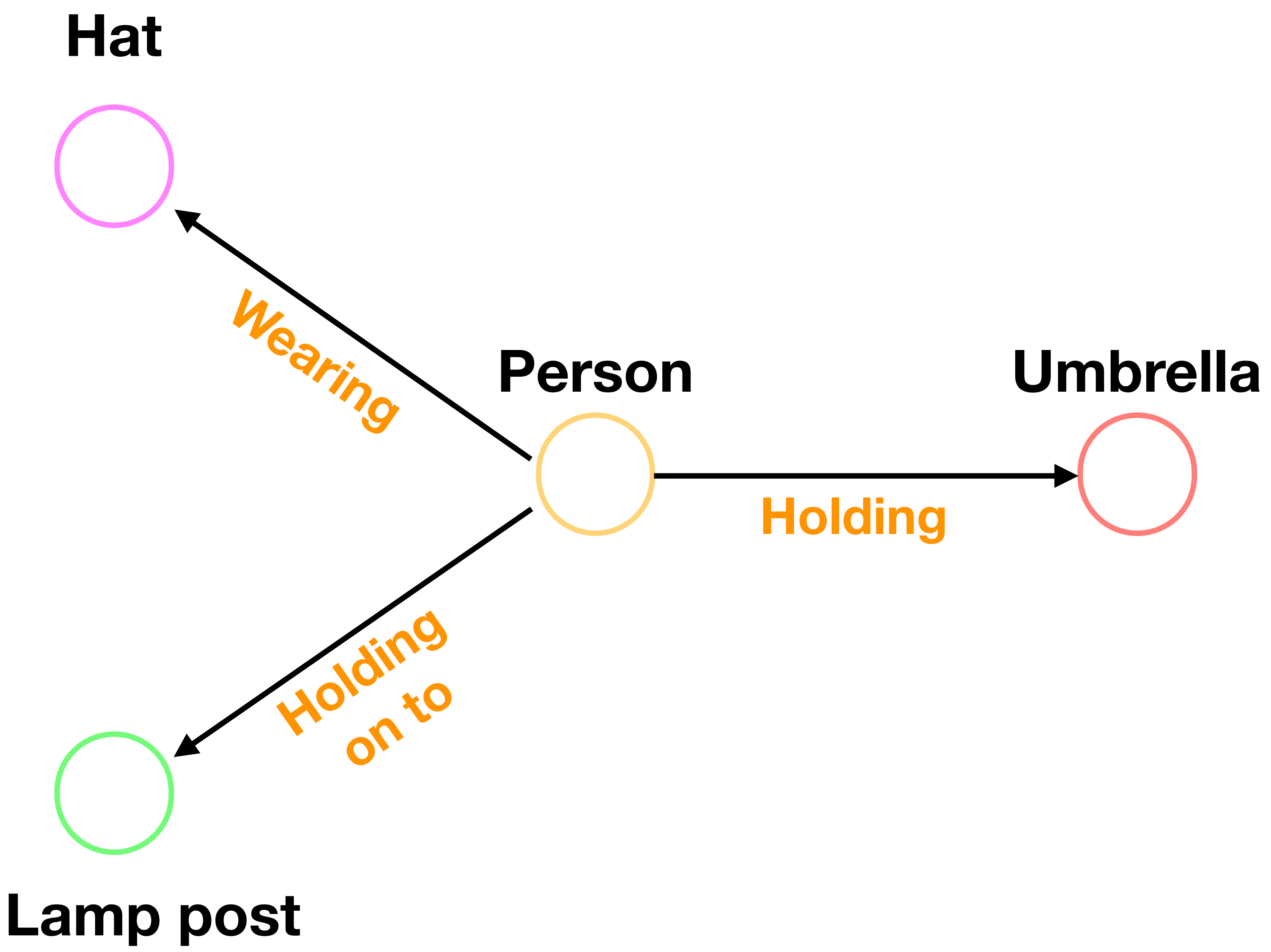
Lamp post

Person

Scene Graphs



Scene Graphs

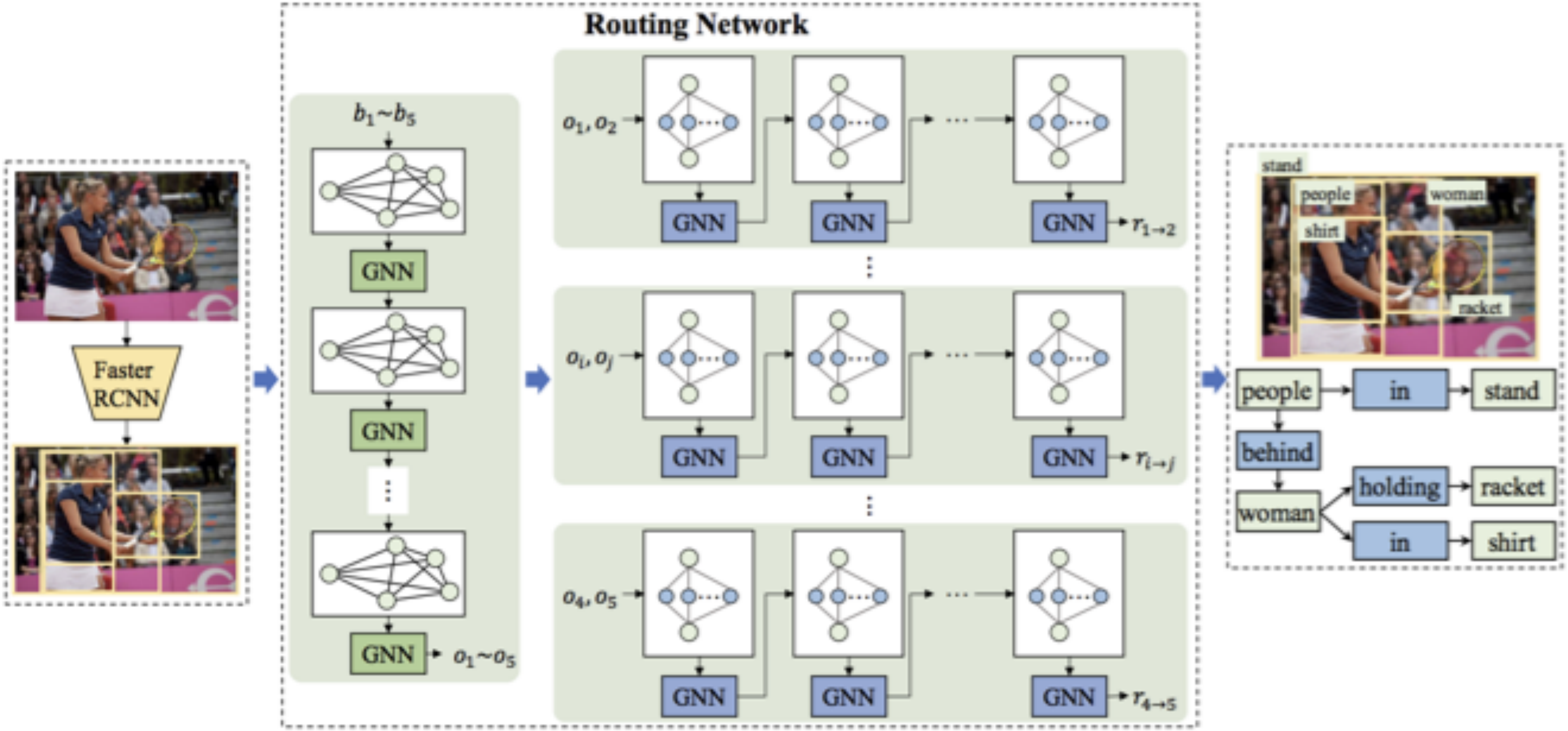


Scene Graph Generation Pipeline

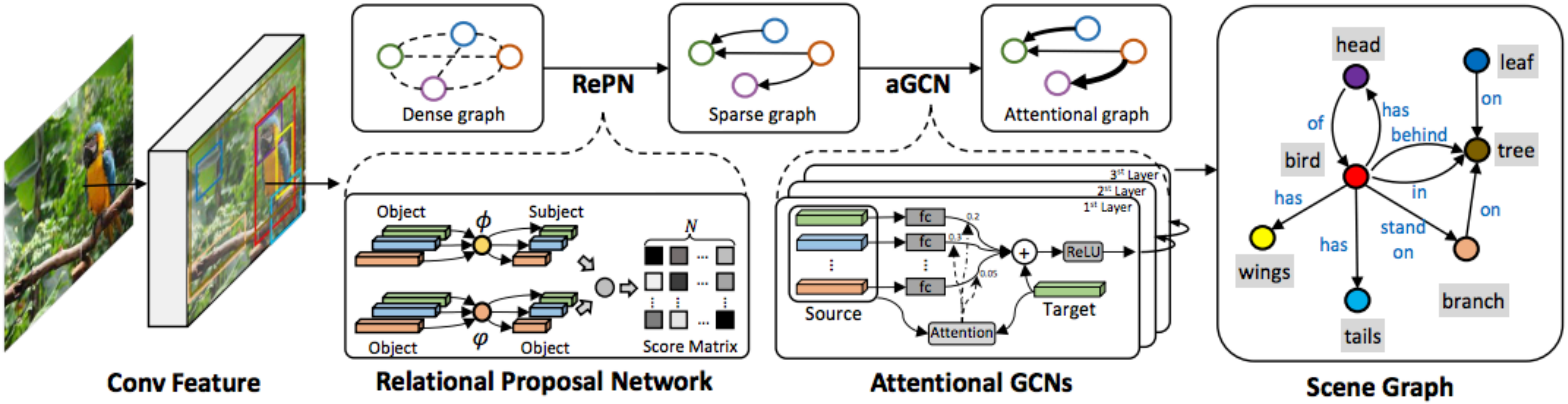

OBJECT DETECTOR
FEATURE EXTRACTION
STATE INITIALIZATION
CONTEXT
ENCODING
SCENE GRAPH
person
standing on
holding
beach
surfboard
FEATURE EMBEDDING
LABEL EMBEDDING
LABEL
PREDICTION
person
beach
surfboard
LOSS COMPUTATION
person
standing on
holding
beach
surfboard

OBJECT DETECTOR
FEATURE
EXTRACTION
STATE INITIALIZATION
CONTEXT
ENCODING
SCENE GRAPH
person
standing
on
holding
beach
surfboard
FEATURE
EMBEDDING
LABEL
EMBEDDING
LABEL
PREDICTION
person
beach
surfboard
LOSS
COMPUTATION
person
beach
surfboard
standing
on
holding
Cross
Entropy
Loss

KERN Architecture



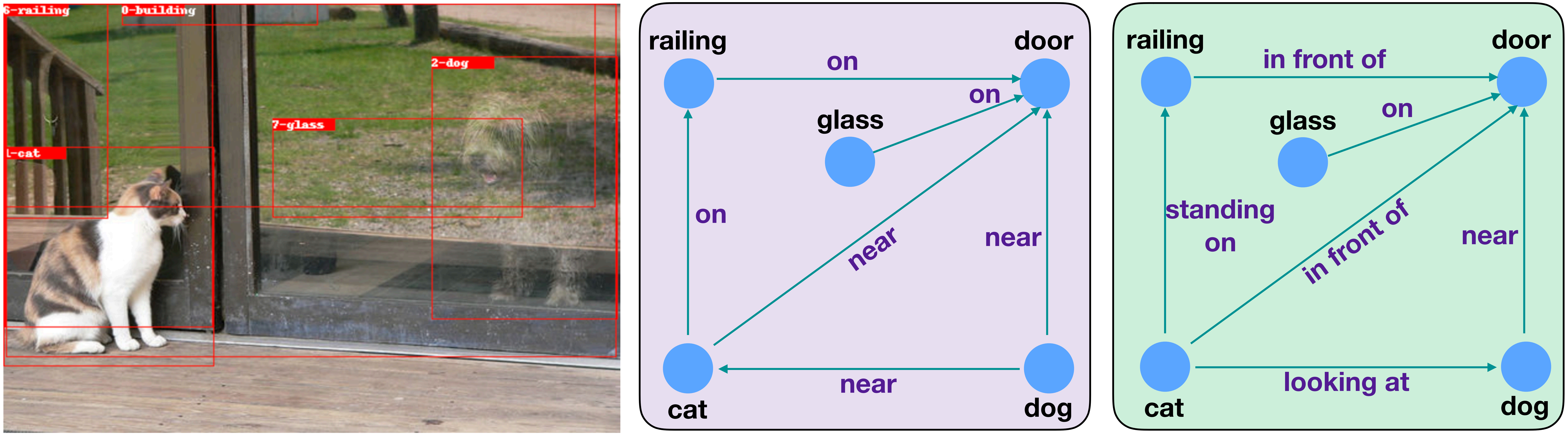
Graph RCNN

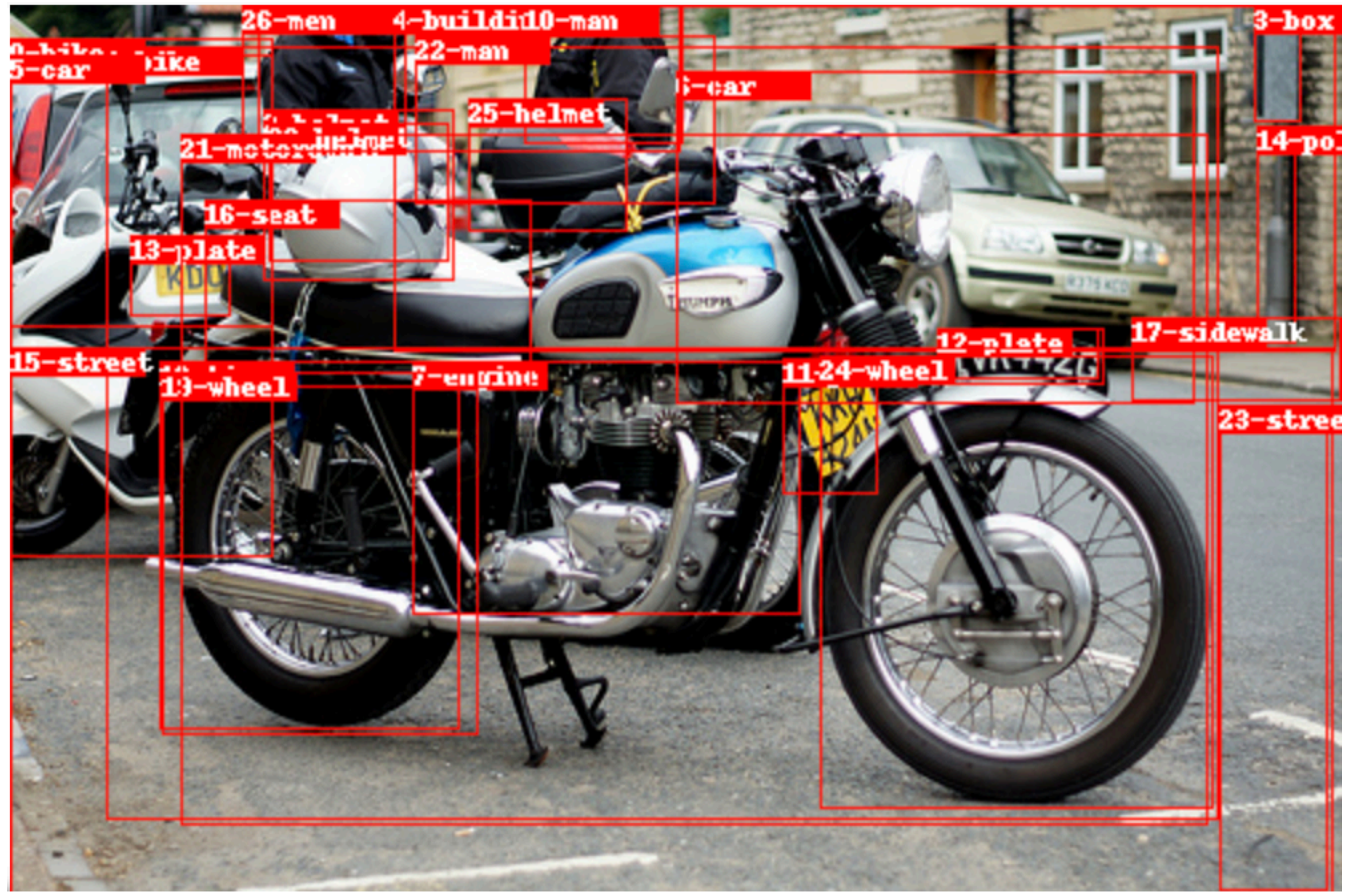

Dense graph
RePN
Sparse graph
aGCN
Attentional graph
Object
Subject
Object
Object
Score Matrix
3rd Layer
2nd Layer
1st Layer
fc
ReLU
Source
Target
Attention
head
leaf
on
of
has
behind
bird
tree
has
in
on
stand on
has
wings
tails
branch
Conv Feature
Relational Proposal Network
Attentional GCNs
Scene Graph

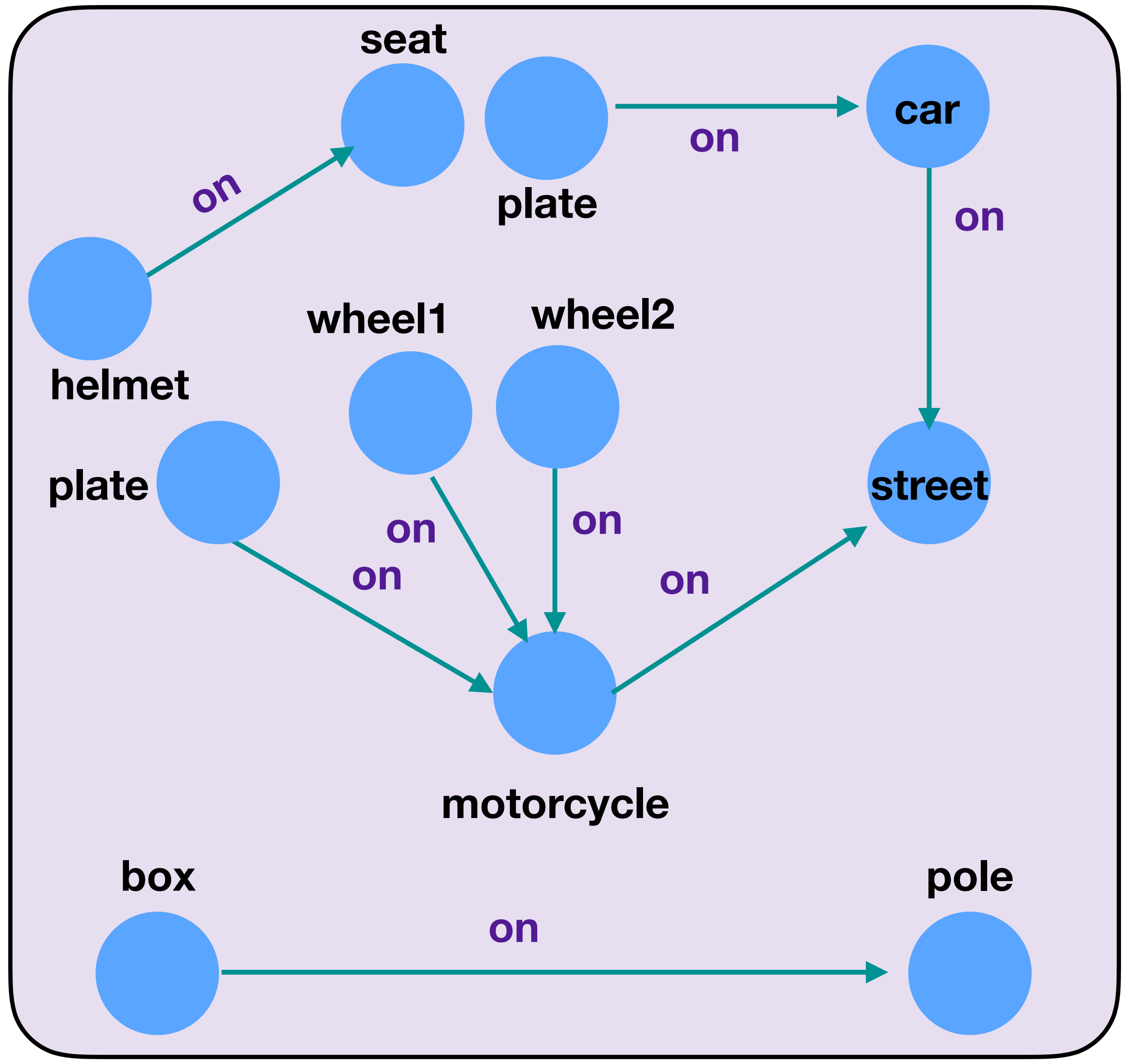

OBJECT DETECTOR
FEATURE
EXTRACTION
STATE INITIALIZATION
CONTEXT
ENCODING
SCENE GRAPH
person
standing
on
holding
beach
surfboard
FEATURE
EMBEDDING
LABEL
EMBEDDING
LABEL
PREDICTION
person
beach
surfboard
LOSS
COMPUTATION
person
beach
surfboard
standing
on
holding
Cross
Entropy
Loss
Structure information
is lost

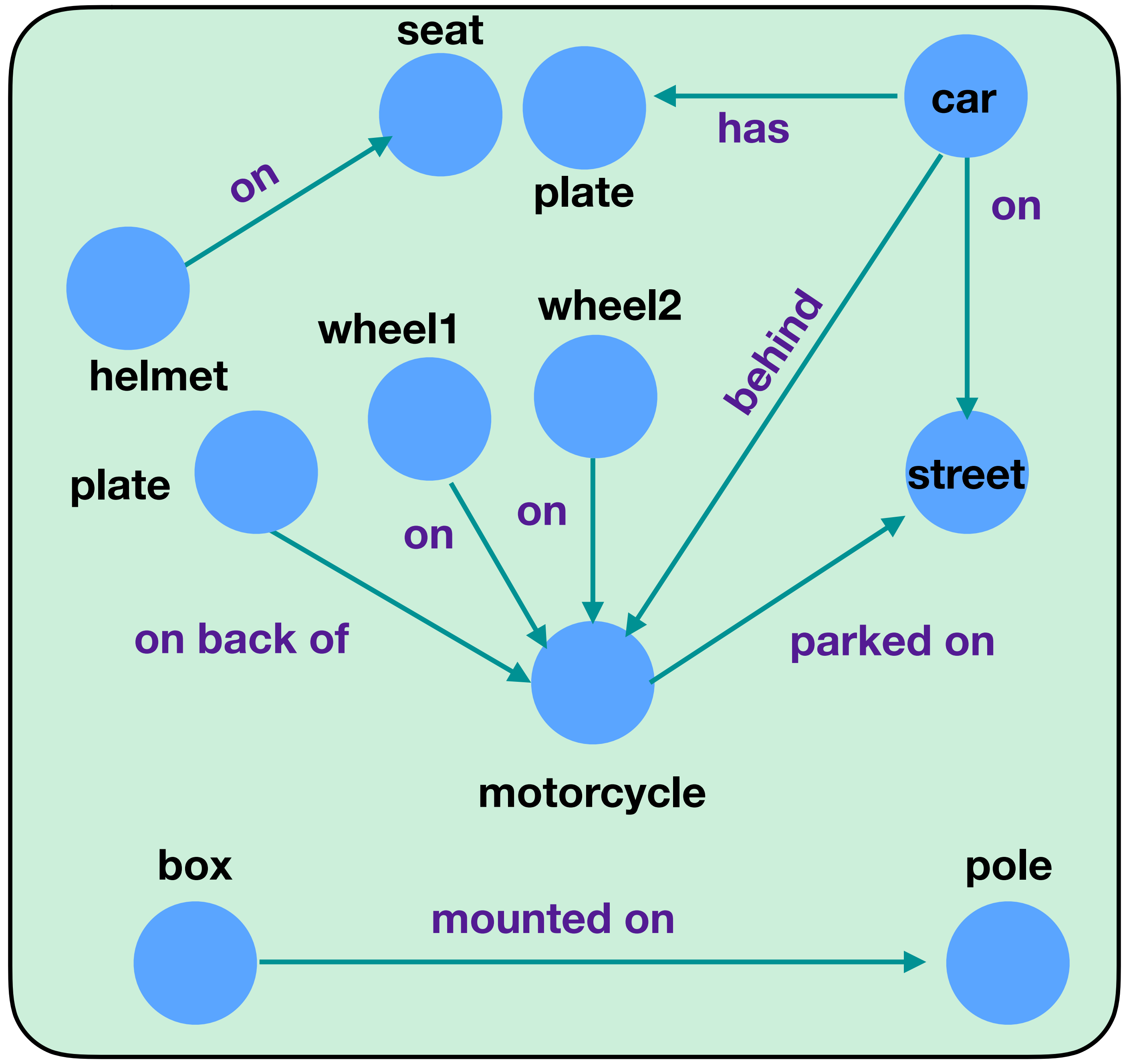
OBJECT DETECTOR
FEATURE EXTRACTION
STATE INITIALIZATION
CONTEXT
ENCODING
SCENE GRAPH
person
standing on
holding
beach
surfboard
FEATURE EMBEDDING
LABEL EMBEDDING
LABEL
PREDICTION
person
beach
surfboard
FEATURE EMBEDDING
IMAGE GRAPH
person
standing on
holding
beach
surfboard
EGNN
GNN
EDGED GATED POOLING
GATED POOLING
MLP
ENERGY VALUE

Visualizations


6-railing
0-building
2-dog
7-glass
1-cat
railing
door
on
glass
on
on
near
near
cat
near
dog
railing
in front of
door
glass
on
standing on
in front of
near
cat
looking at
dog


26-men
22-man
25-helmet
3-box
16-seat
13-plate
17-sidewalk
12-plate
15-street


seat
plate
on
car
on
on
helmet
wheel1
wheel2
plate
on
on
on
on
street
motorcycle
box
on
pole


seat
plate
has
car
on
on
helmet
wheel1
wheel2
behind
plate
on
on
street
on back of
parked on
motorcycle
box
mounted on
pole


railing
door
on
glass
on
on
near
near
near
cat
dog
railing
door
in front of
glass
on
standing on
in front of
near
looking at
cat
dog
seat
plate
car
on
on
on
helmet
wheel1 wheel2
plate
street
on
on
on
on
motorcycle
box
on
pole
seat
plate
has
car
on
on
helmet
wheel1
wheel2
behind
plate
street
on
on
on back of
parked on
motorcycle
box
mounted on
pole
person1
behind
person4
on
near
on
person0
on
street
person1
behind
person4
walking on
walking on
with
person0
walking on
street
girl
wearing
shirt
on
person1
street
on
wheel
behind
has
person2
on
horse
girl
wearing
shirt
standing on
person1
street
on
wheel
behind
has
person2
riding
horse
elephant
has
trunk
glasses
on
wearing
wearing
pant
woman
wearing
shirt
elephant
has
trunk
has
glasses
behind
wearing
pant
woman
wearing
shirt
Zero Shot
elephant
has
glasses
stand
near
door
coat
above
near
on
cabinet
near
table
stand
near
door
coat
on
near
over
on
over
cabinet
near
table
Zero Shot
coat
on
door

Conclusions

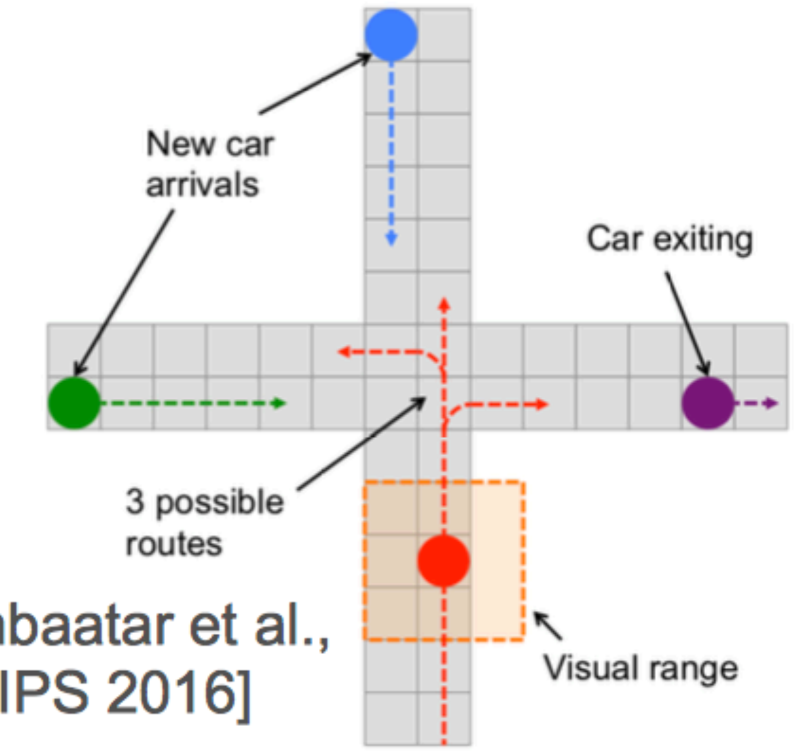
— **Deep learning on graphs works and is very effective!**

— Exciting area: lots of new applications and extensions (hard to keep up)

Relational reasoning

[Santoro et al., NIPS 2017]

Multi-Agent RL



[Sukhbaatar et al., NIPS 2016]

GCN for recommendation on 16 billion edge graph!

[Leskovec lab, Stanford]

Open problems:

— Theory

— Scalable, stable generative models

— Learning on large, evolving data

— Multi-modal and cross-model learning (e.g., sequence2graph)

* slide from Thomas Kipf, **University of Amsterdam**