THE UNIVERSITY OF BRITISH COLUMBIA

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

Lecture 20: Graph Neural Networks (cont)

Logistics

— **Paper readings 3** & **4** — Your choices, quiz should be visible

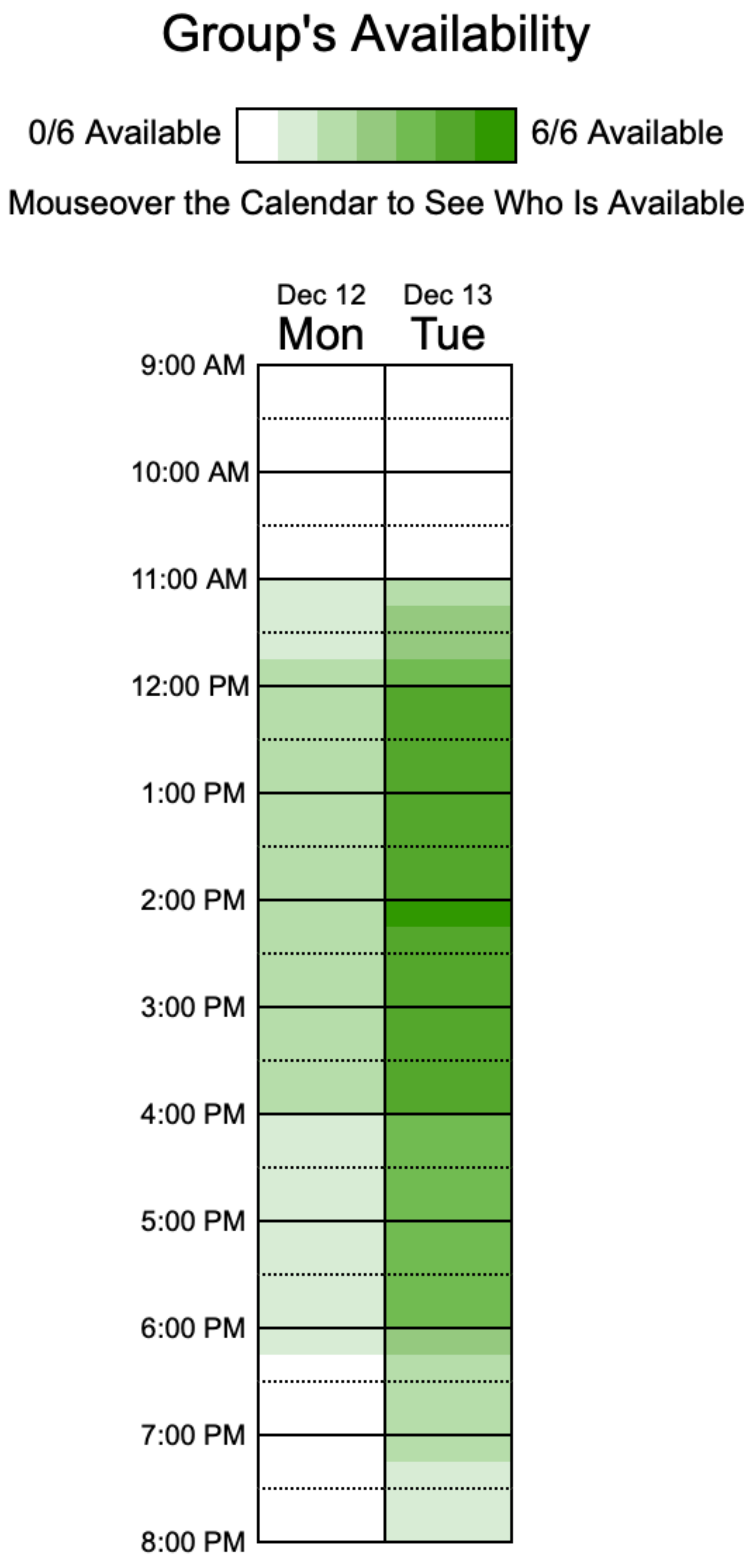
— **Assignment 3** & **4** are being graded, out this week

— Survey for **final presentations** is out:

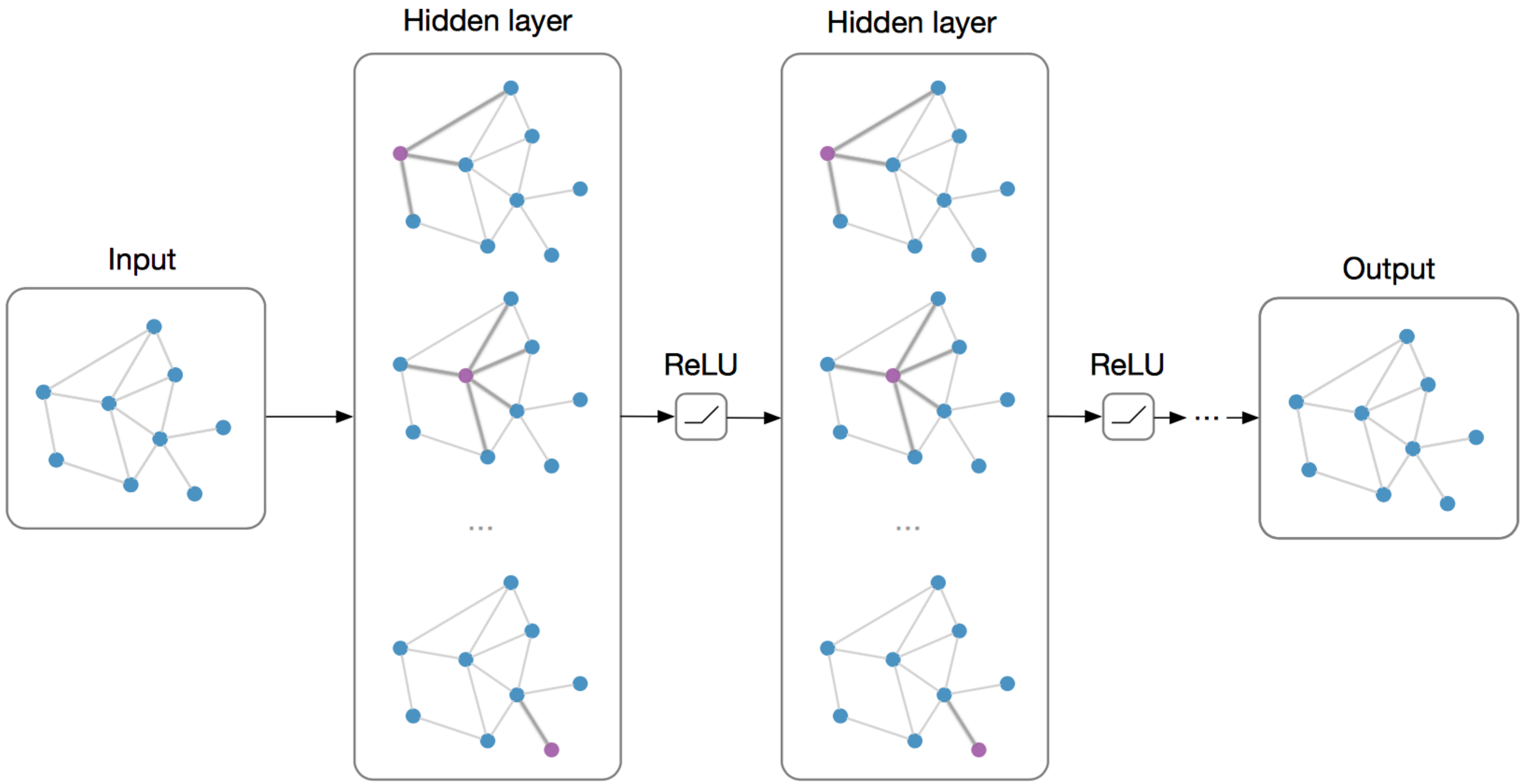
https://www.when2meet.com/?17859912-5BHpw

mark time unavailable if you PHYSICALLY can't make it

(e.g., another course exam)



Graph Neural Networks (GNNs)



Main Idea: Pass massages between pairs of nodes and agglomerate

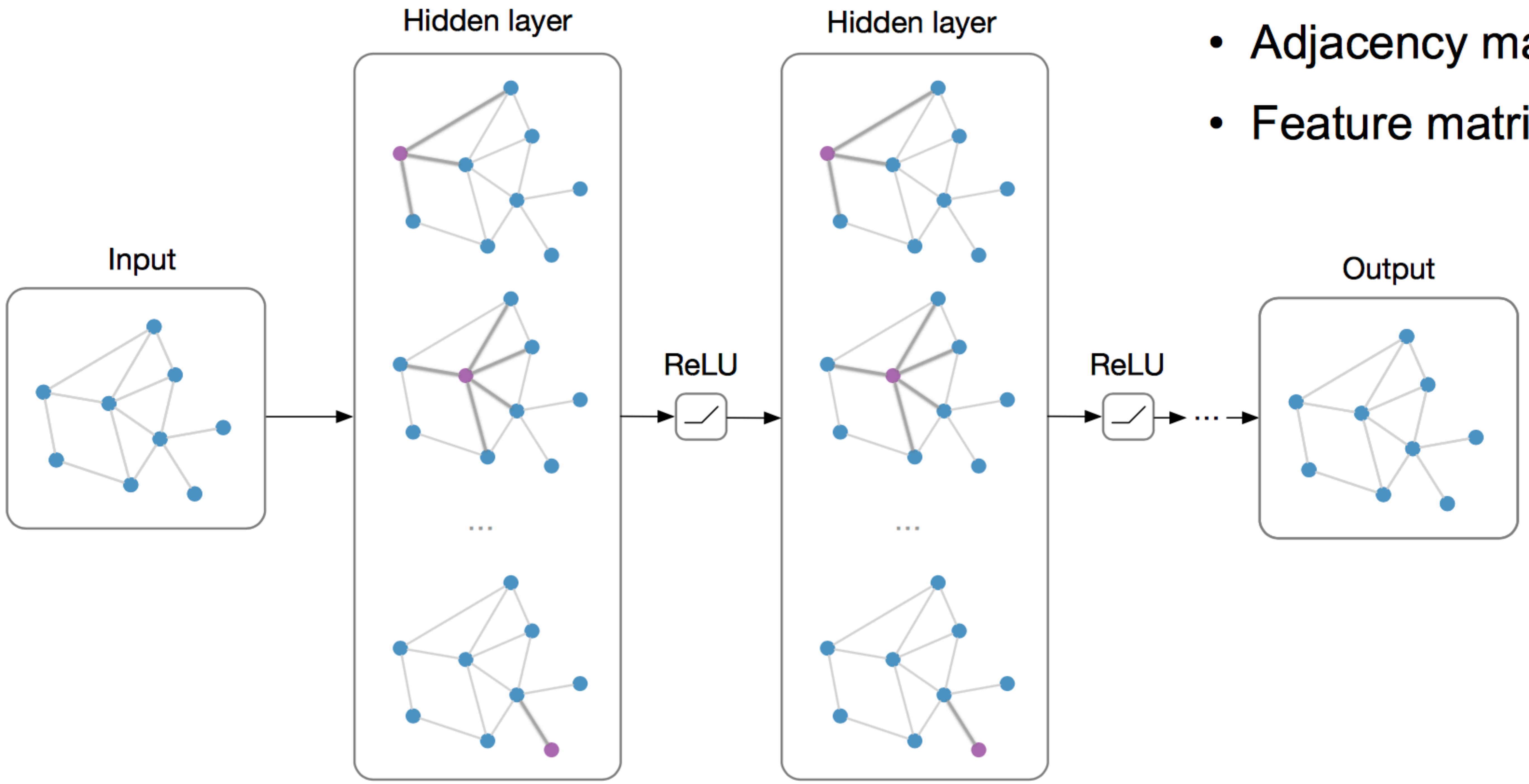
Alternative Interpretation: Pass massages between nodes to refine node (and possibly edge) representations

* 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 $\mathbf{X} \in \mathbb{R}^{N \times F}$

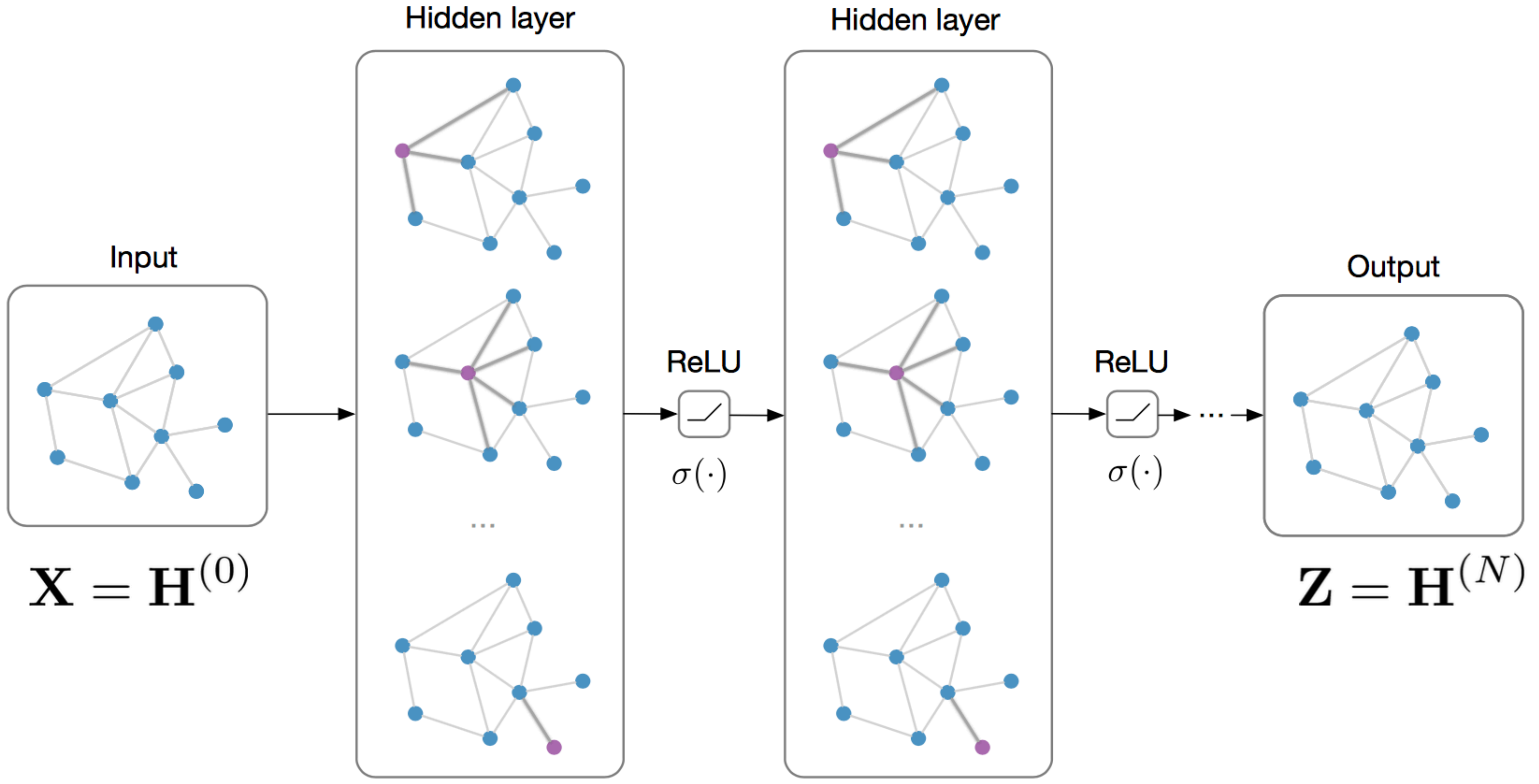


Main Idea: Pass massages between pairs of nodes and agglomerate

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

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

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

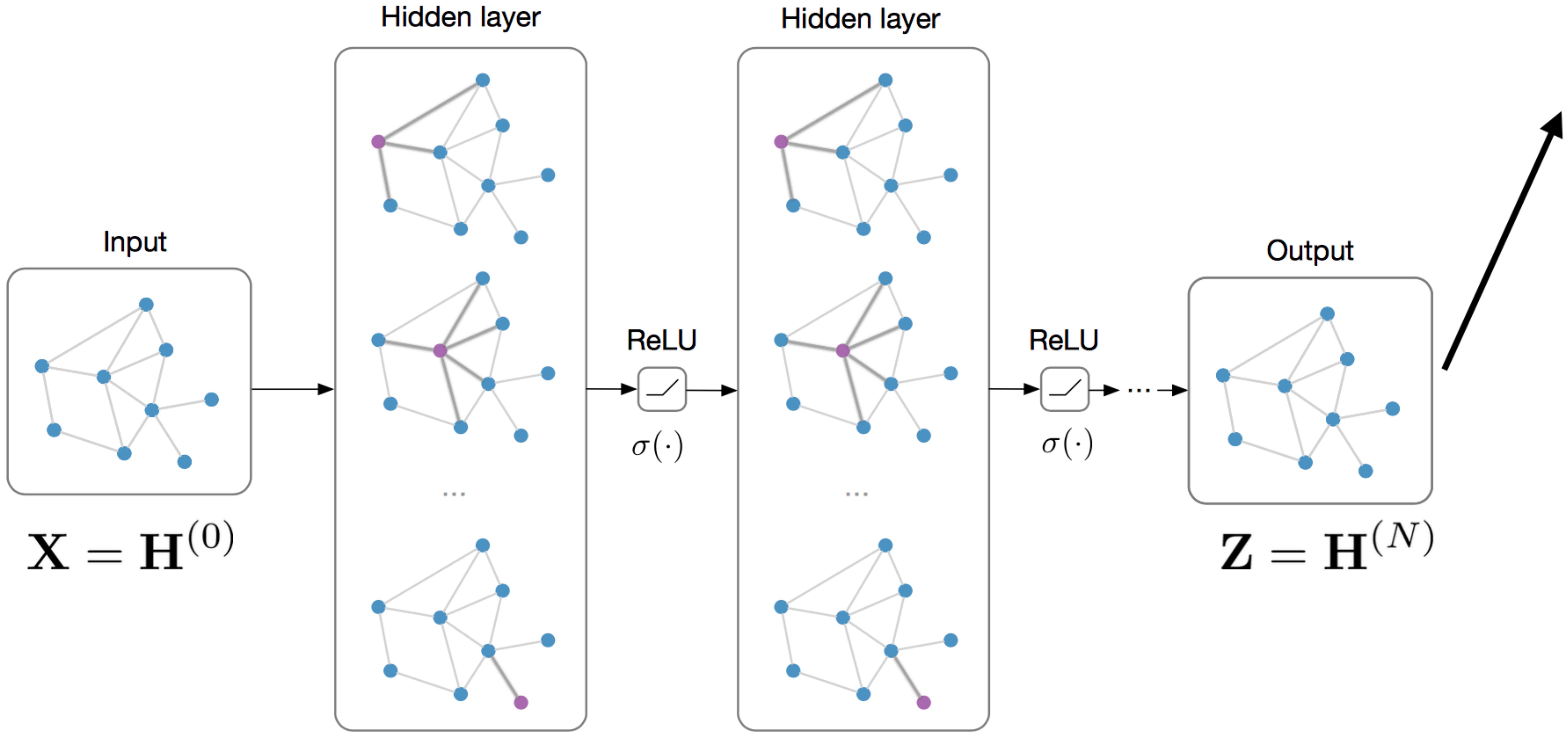


$$\mathbf{H}^{(l+1)} = \text{Message Passing}\left(\mathbf{A}, \mathbf{H}^{(l)}\right)$$

* 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 $\hat{\mathbf{A}}$



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

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

Node classification:

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

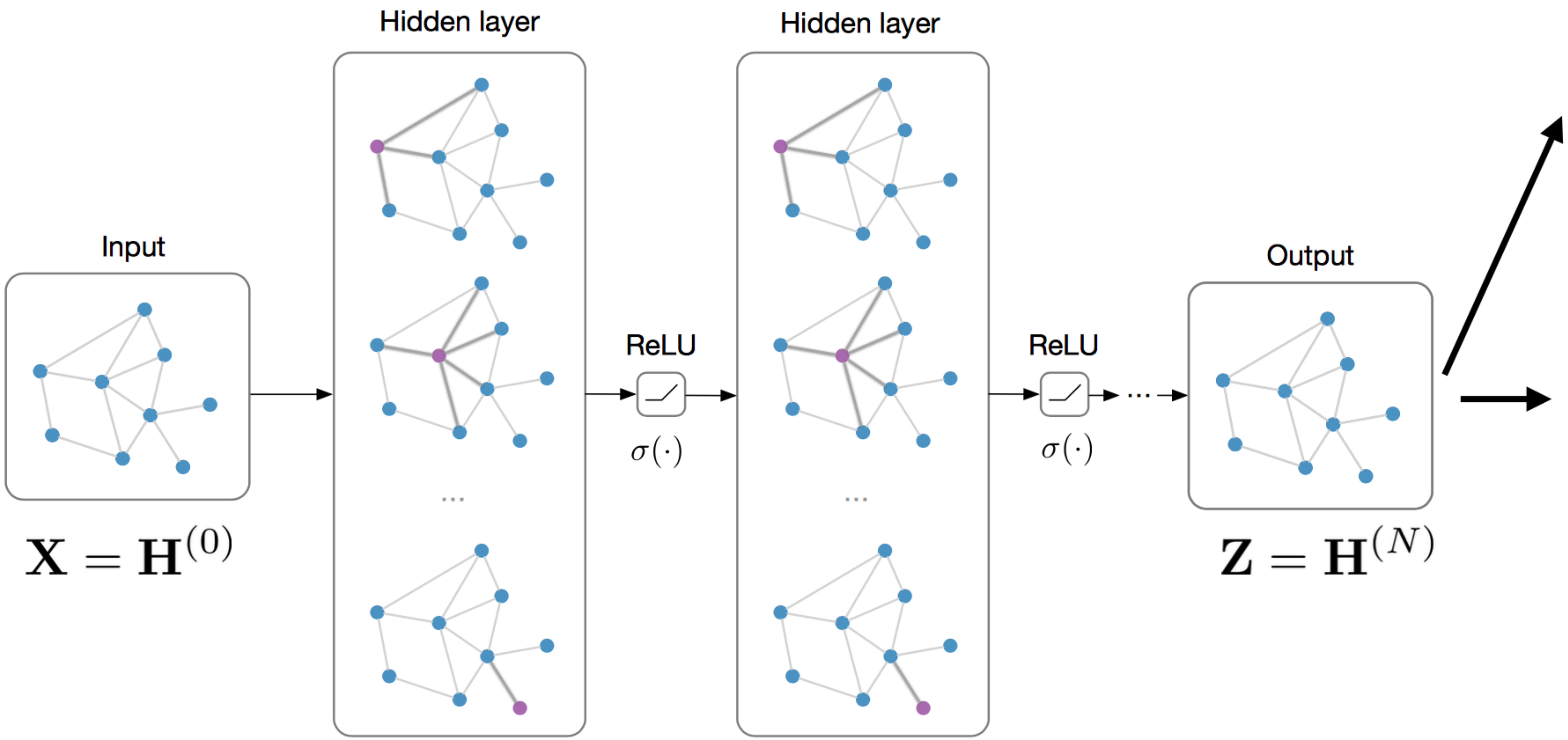
e.g. Kipf & Welling (ICLR 2017)

$$\mathbf{H}^{(l+1)} = \text{Message Passing}\left(\mathbf{A}, \mathbf{H}^{(l)}\right)$$

* 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 $\hat{\mathbf{A}}$



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

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

Node classification:

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

e.g. Kipf & Welling (ICLR 2017)

Graph classification:

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

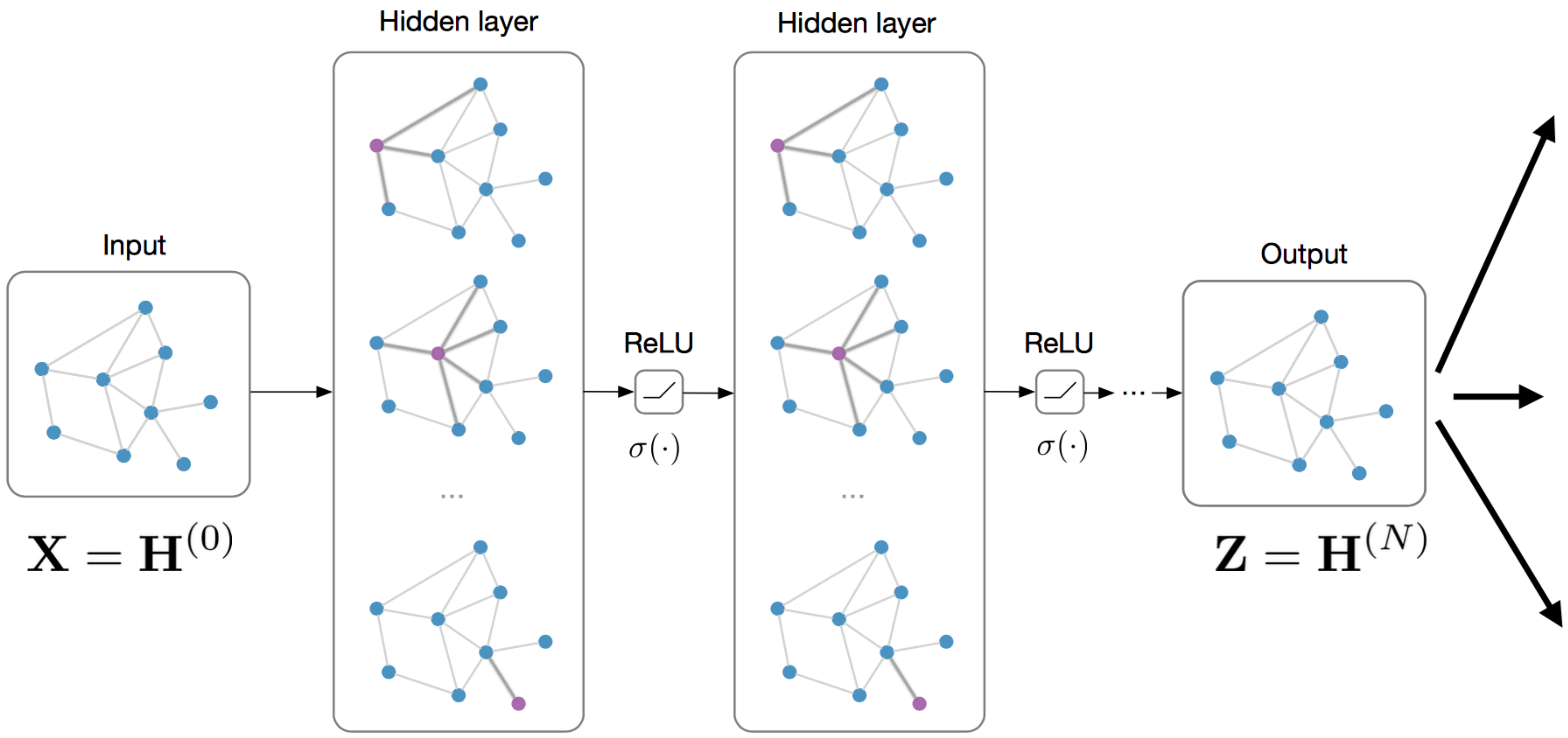
e.g. Duvenaud et al. (NIPS 2015)

$$\mathbf{H}^{(l+1)} = \text{Message Passing}\left(\mathbf{A}, \mathbf{H}^{(l)}\right)$$

* 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 $\hat{\mathbf{A}}$



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

$$\mathbf{H}^{(l+1)} = \text{Message Passing}\left(\mathbf{A}, \mathbf{H}^{(l)}\right)$$

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)

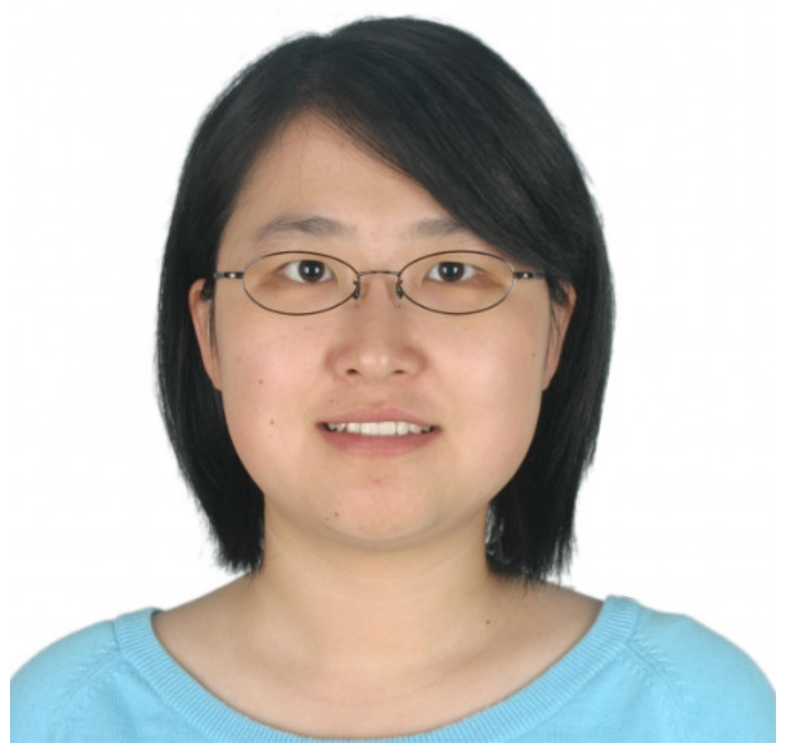
Link prediction:

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

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

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

G^3raphGround: Graph-based Language Grounding

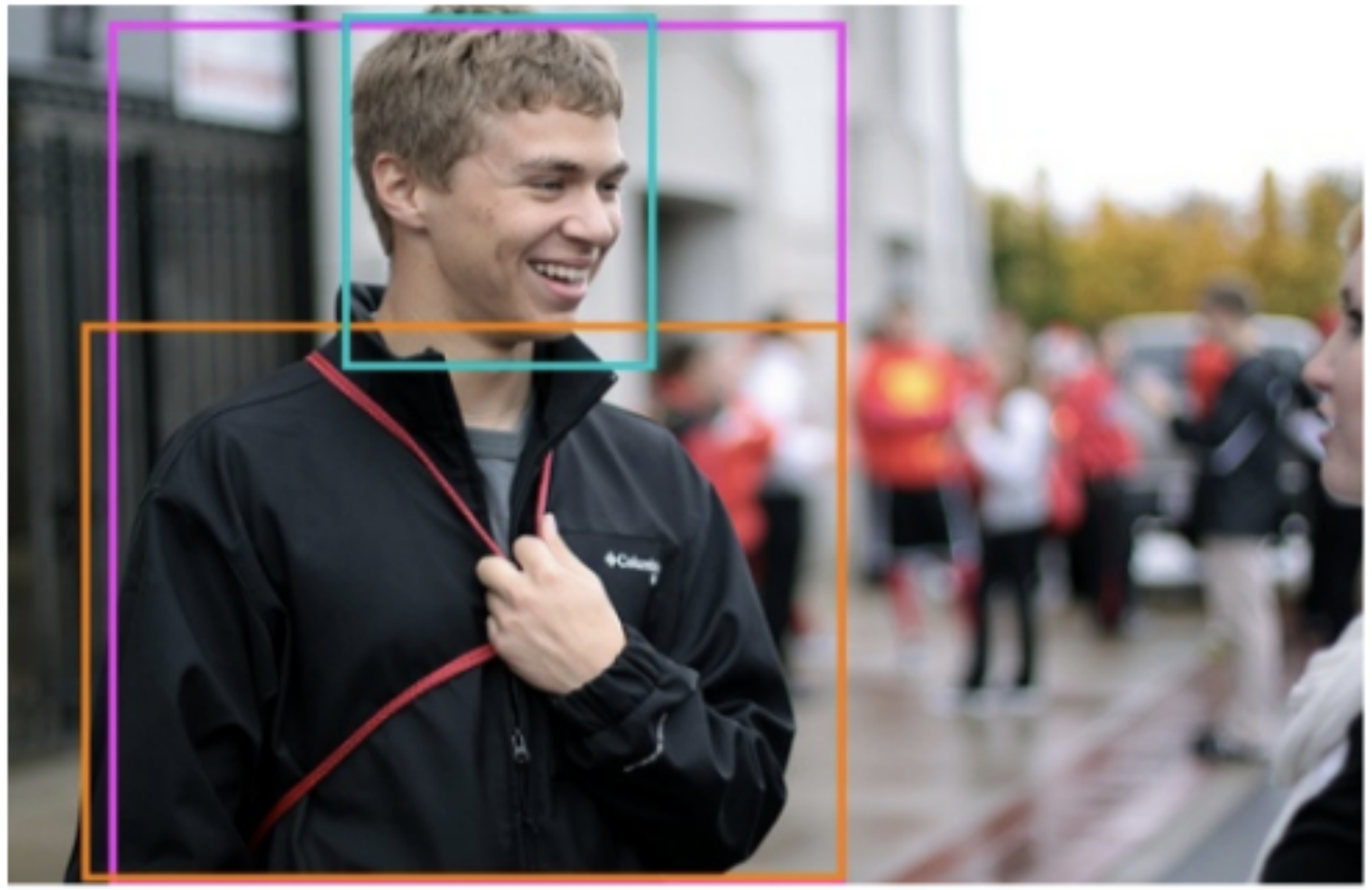

Mohit Bajaj

Lanjun Wang

Leonid Sigal

Image Grounding: Beyond Object Detection

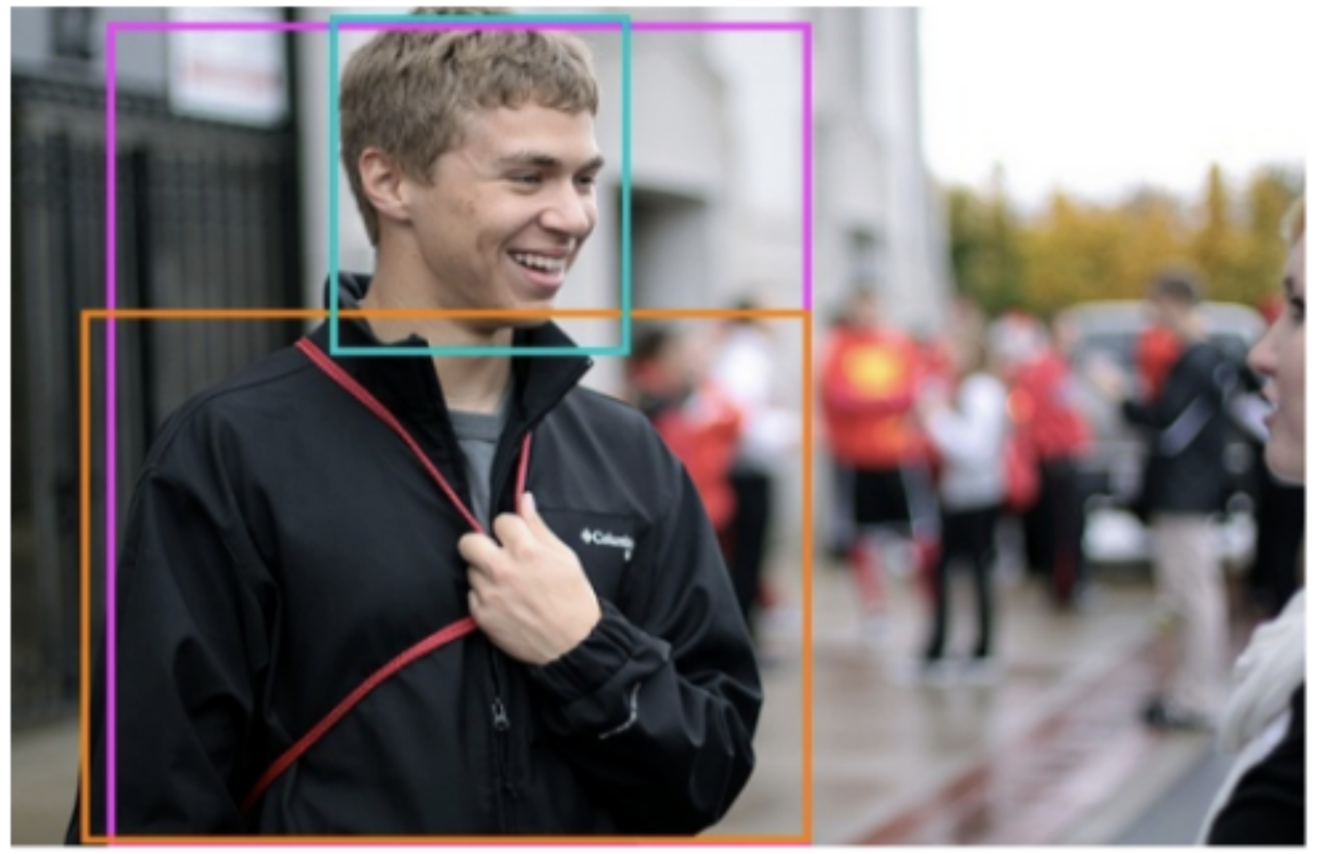
Given the **image** and one or more **natural language phrases**, locate regions that correspond to those phrases.



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.



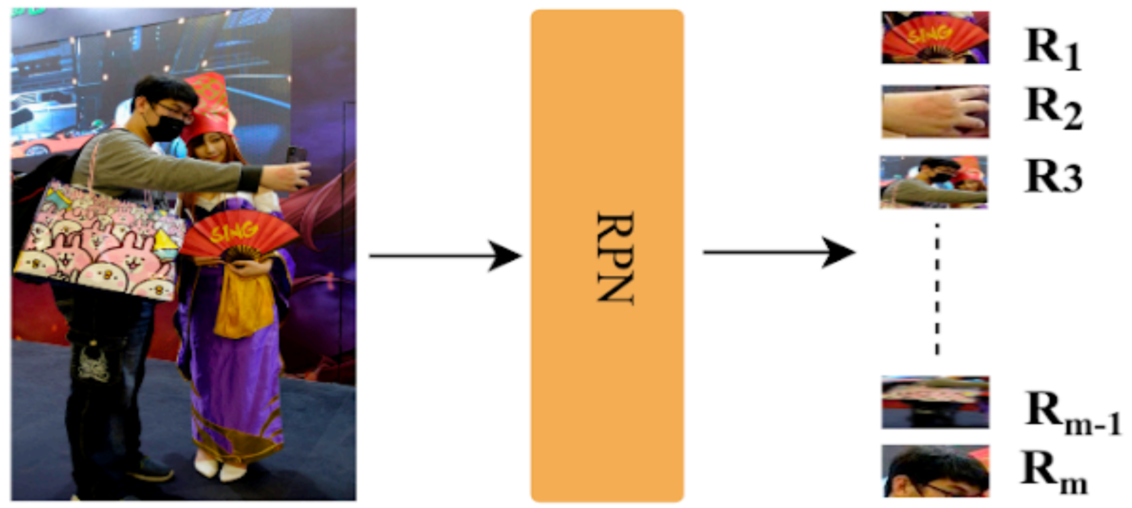
A man wearing a black-jacket has a smile on his face.

Fundamental task for **image / video understanding**

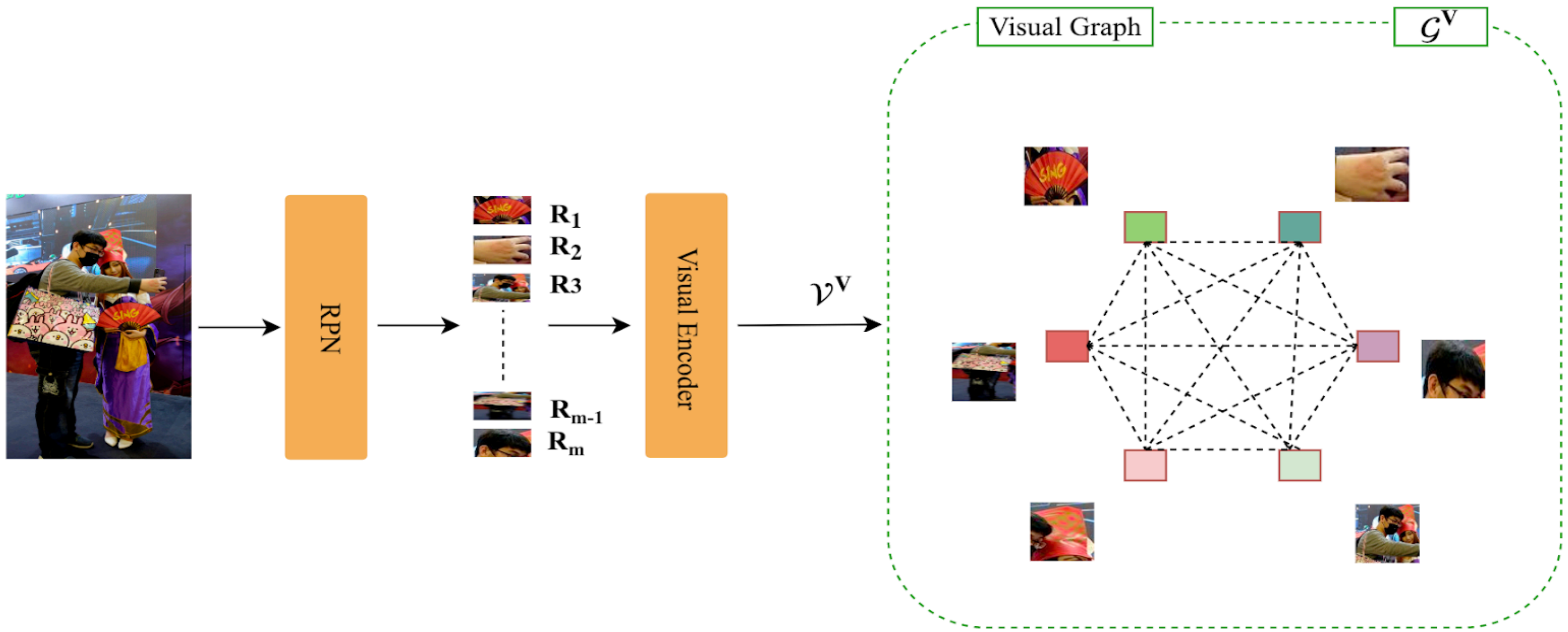
— Helps improve performance on other tasks (e.g., image captioning, VQA)

Proposed **Architecture**

Proposed **Architecture**

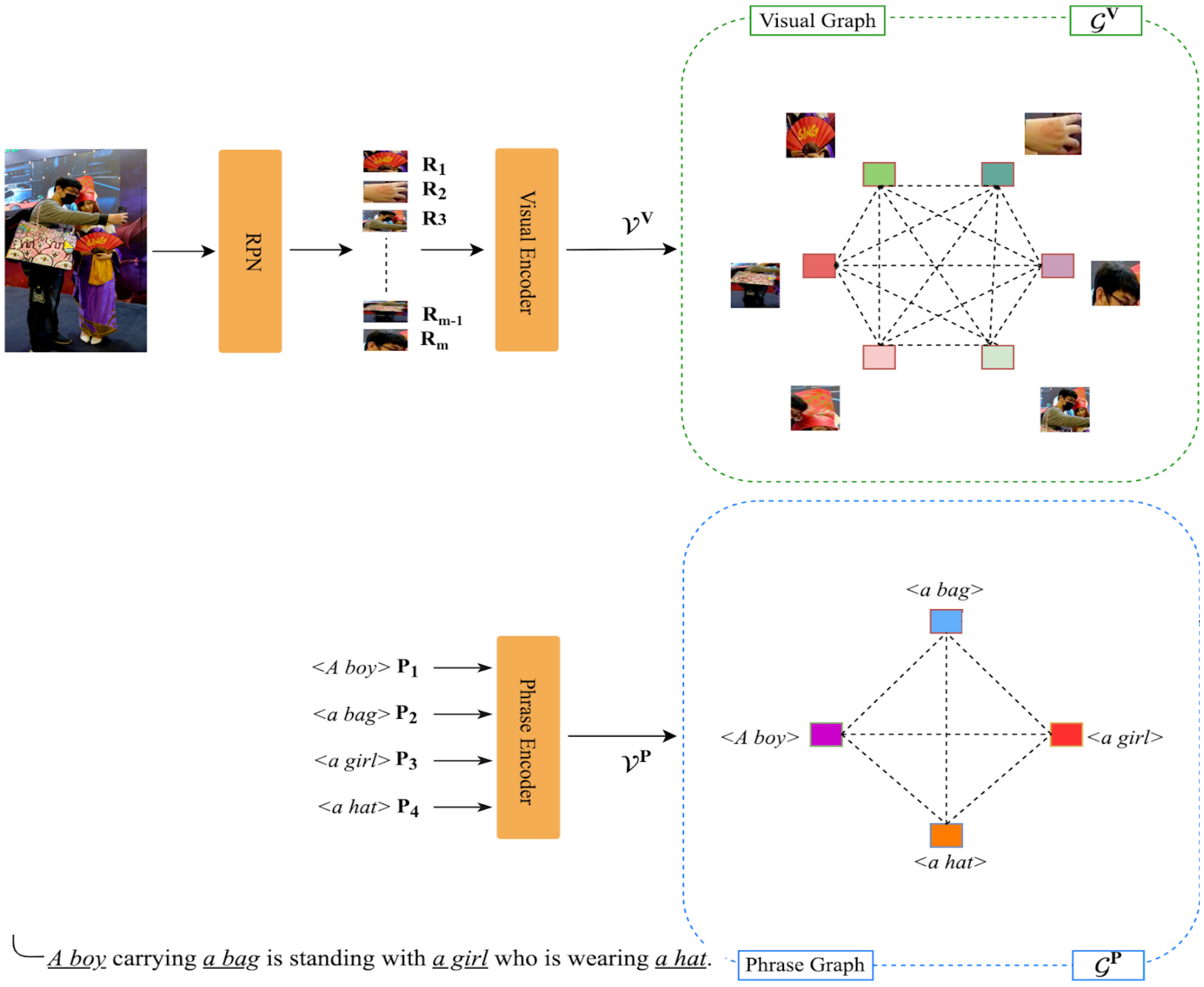


Proposed **Architecture**



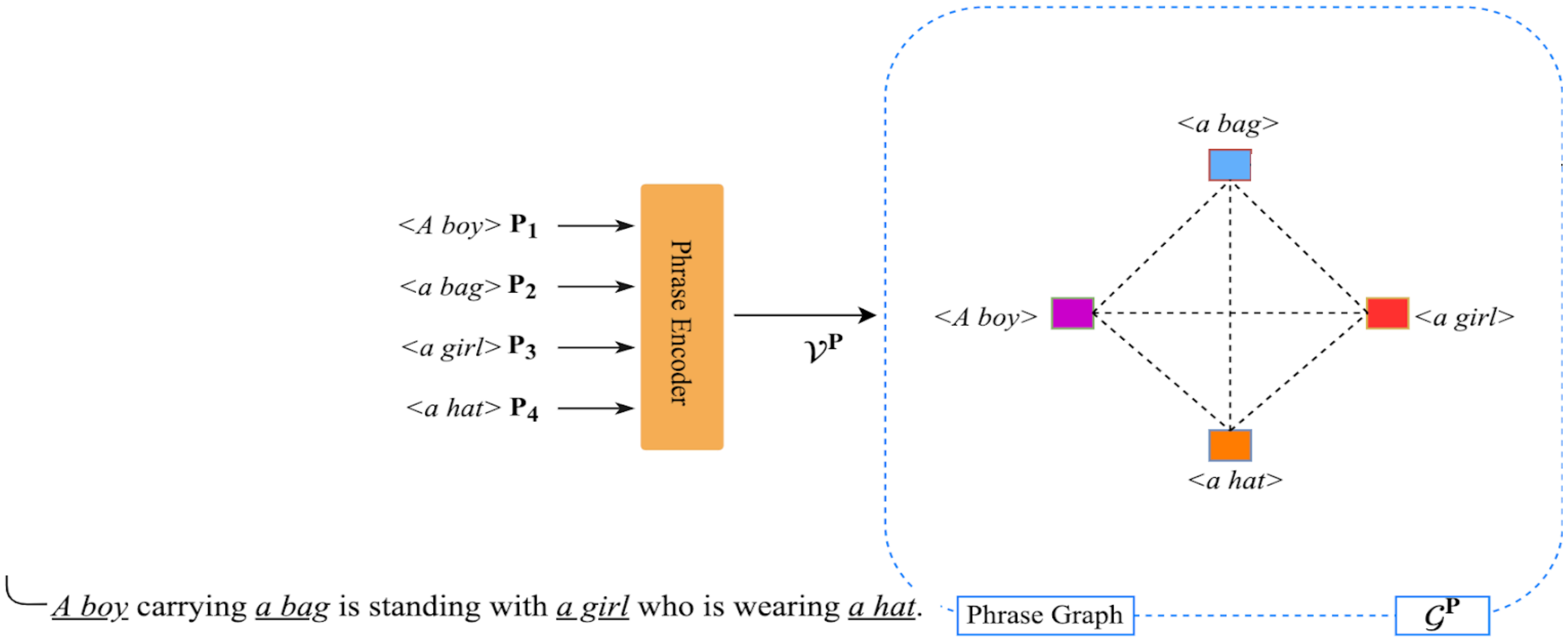
Each visual node is initialized = VGG16 representation (4096) -> 300 Dim + (x, y, w, h)

Proposed **Architecture**

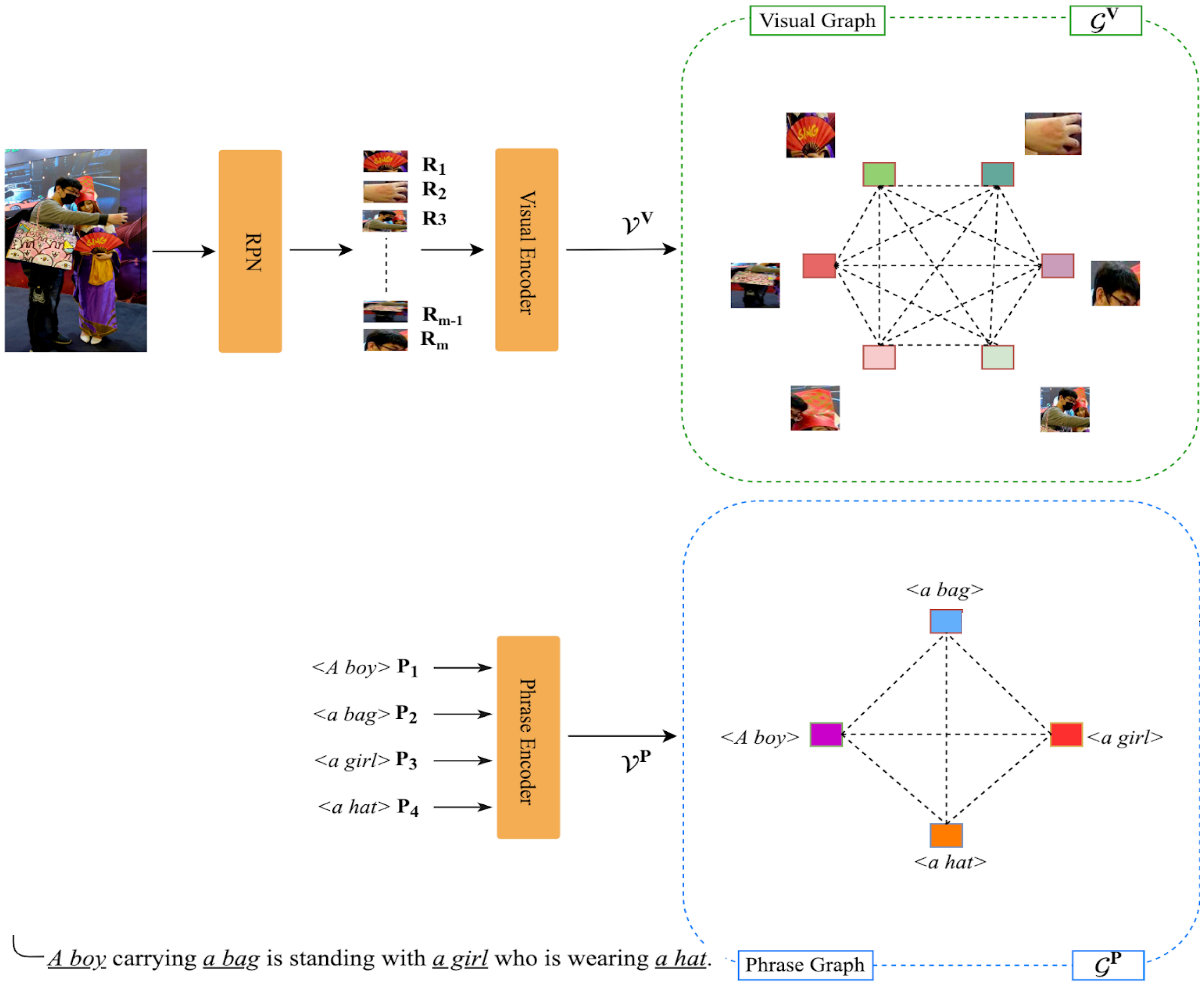


Proposed **Architecture**

Each phrase node is initialized = Bi-directional LSTM last hidden state



Proposed **Architecture**



1. Compute Messages

$$\alpha_{ij}^k \mathbf{W}^k \vec{h}_j$$

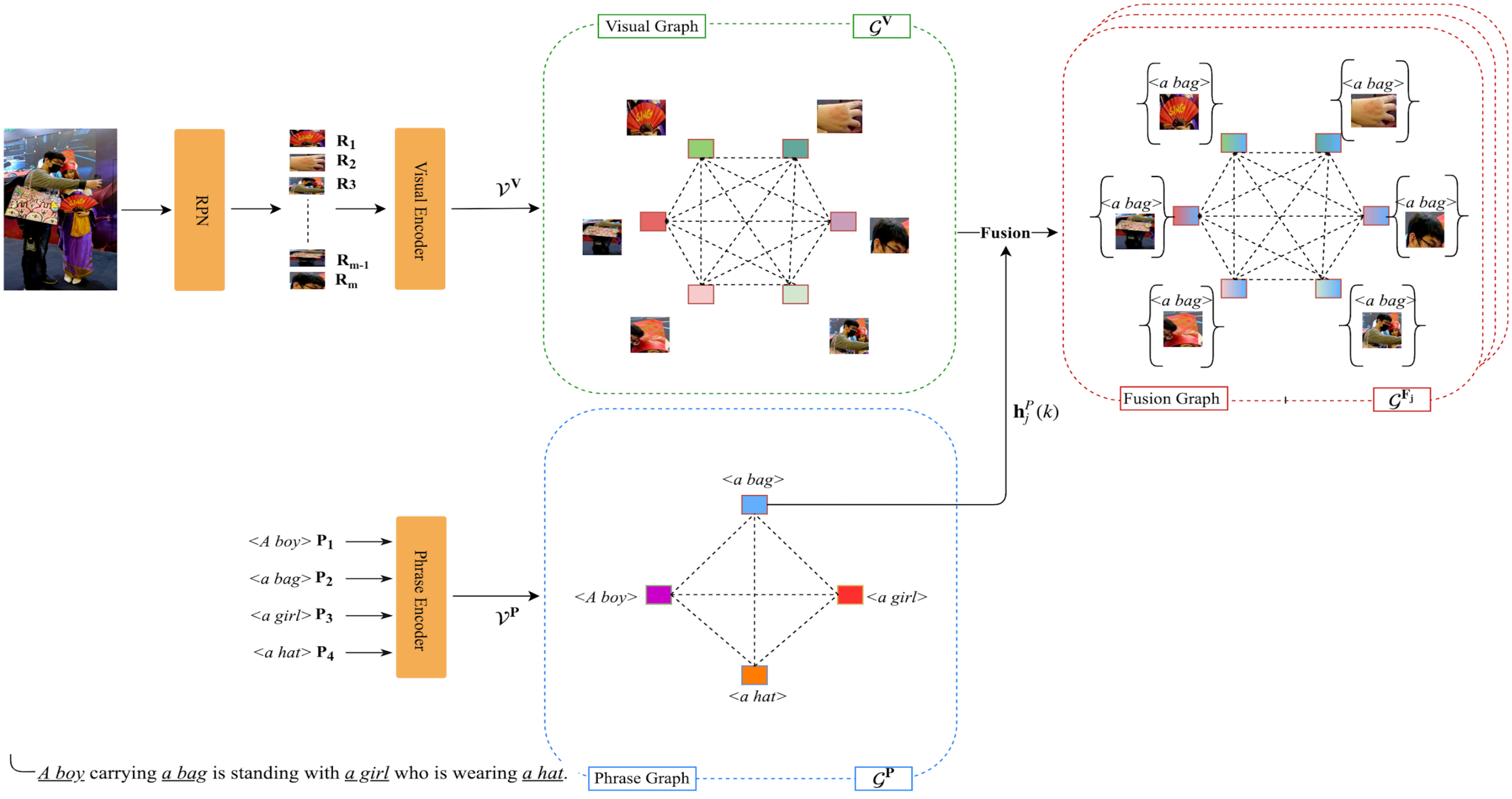
2. Aggregate Messages

$$f_{\text{agg}}\left(\{\mathbf{m}_{ji}^t | j \in \mathcal{N}_i\}\right) = \sum_{j \in \mathcal{N}_i} \mathbf{m}_{ji}^t$$

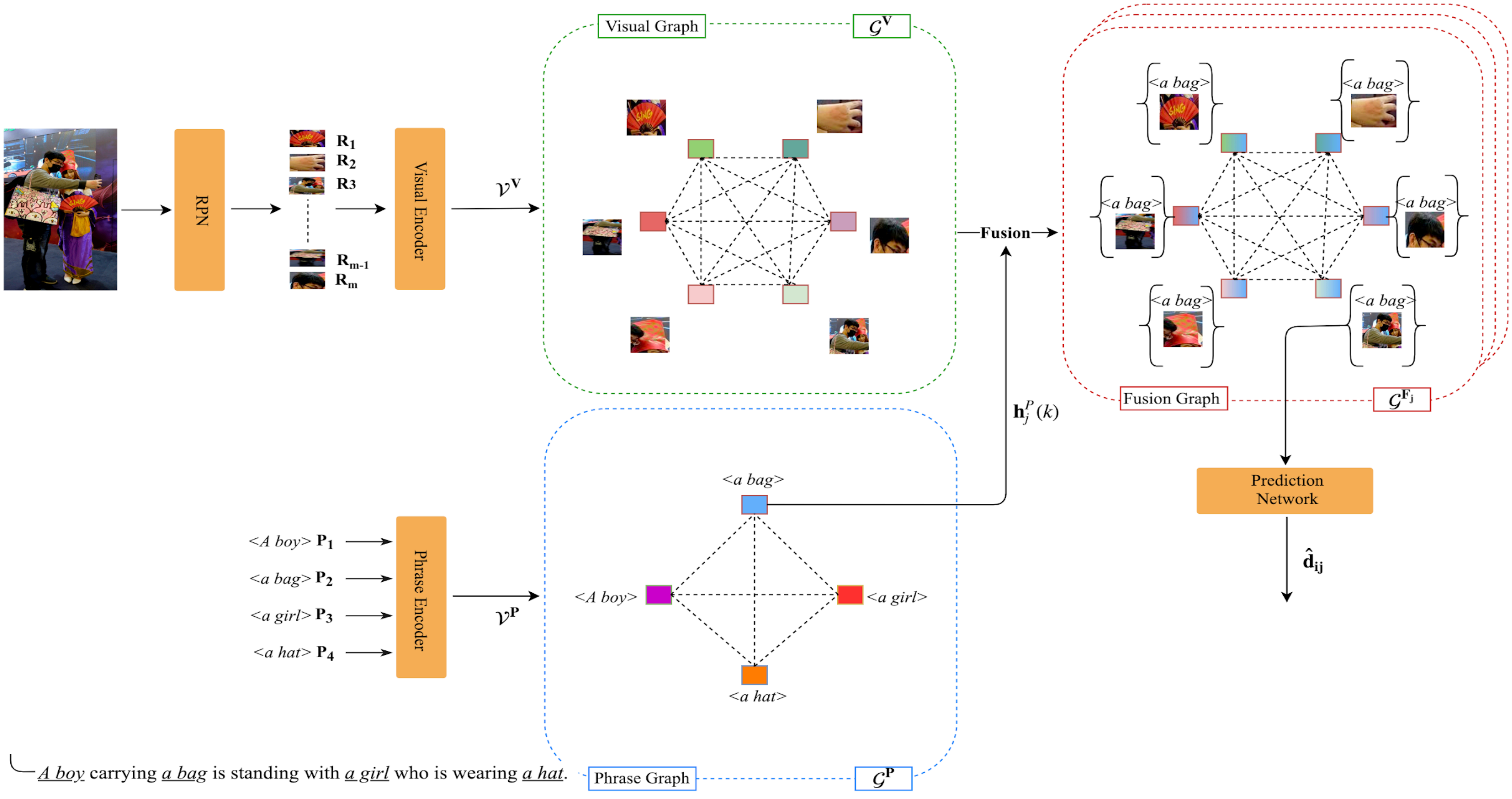
3. Update Node Representations

$$f_{\text{update}}(\mathbf{h}_i^t, \bar{\mathbf{m}}_i^t) = \text{GRU}(\mathbf{h}_i^t, \bar{\mathbf{m}}_i^t)$$

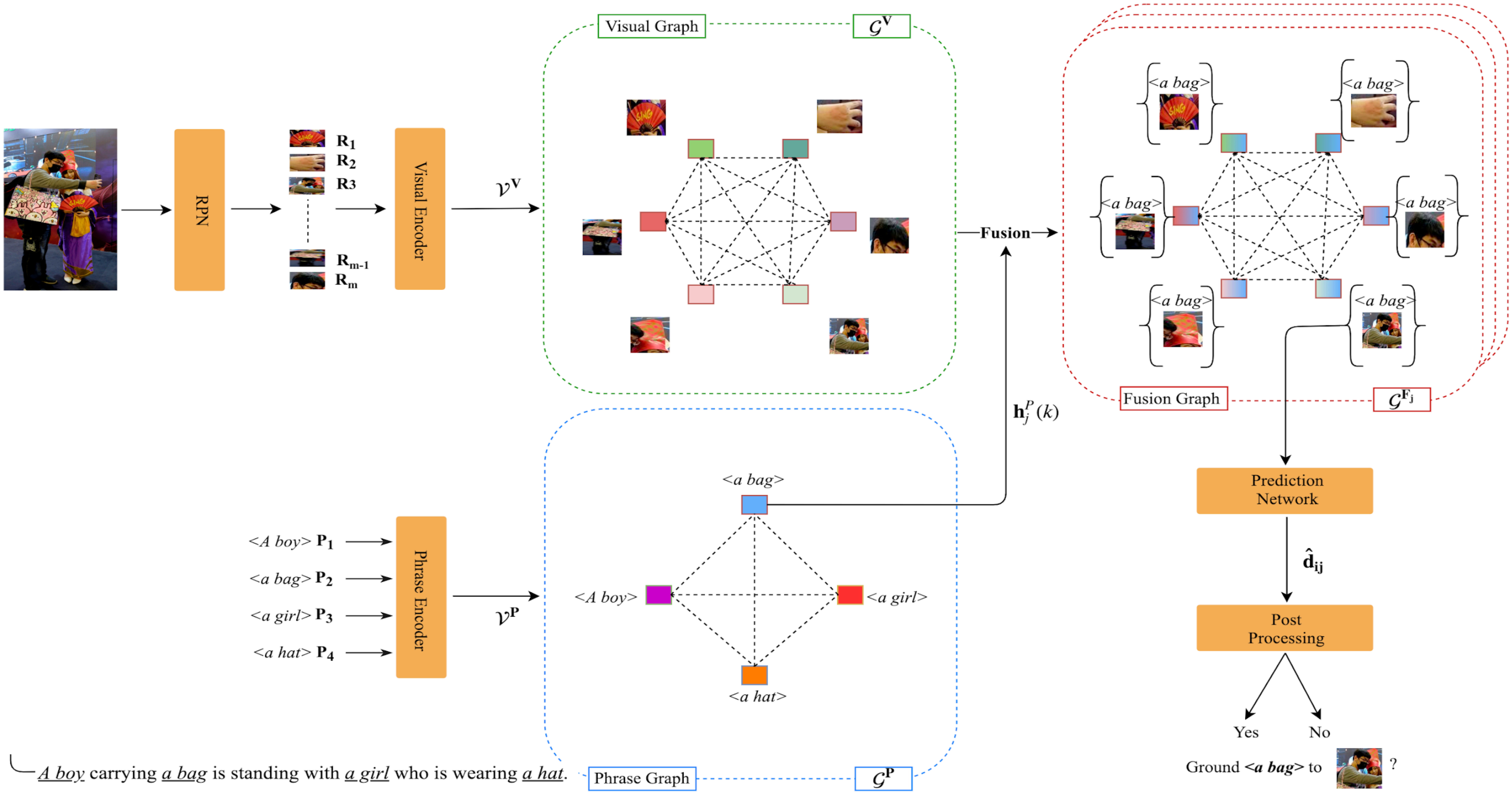
Proposed **Architecture**



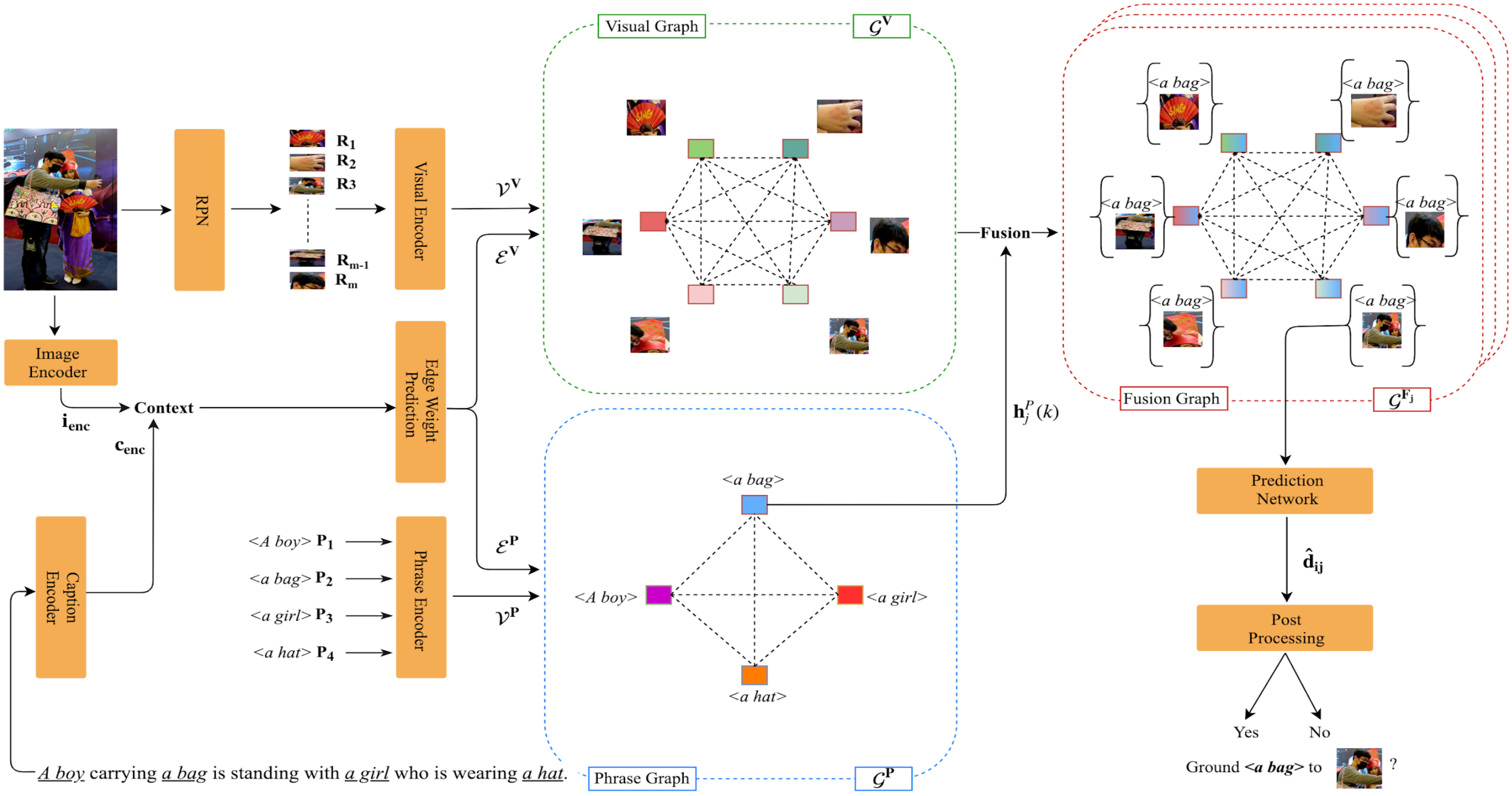
Proposed **Architecture**



Proposed **Architecture**

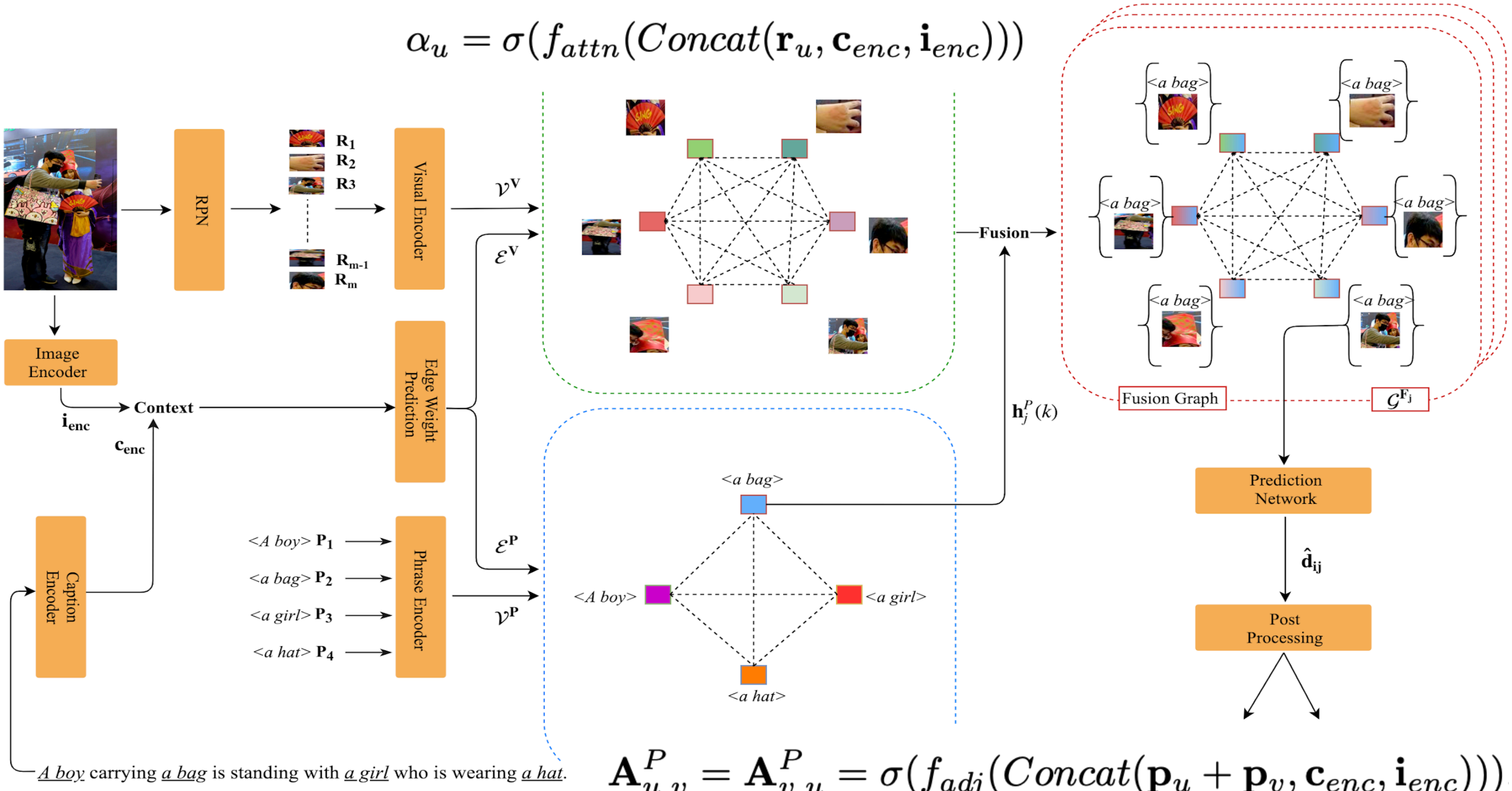


Proposed **Architecture**



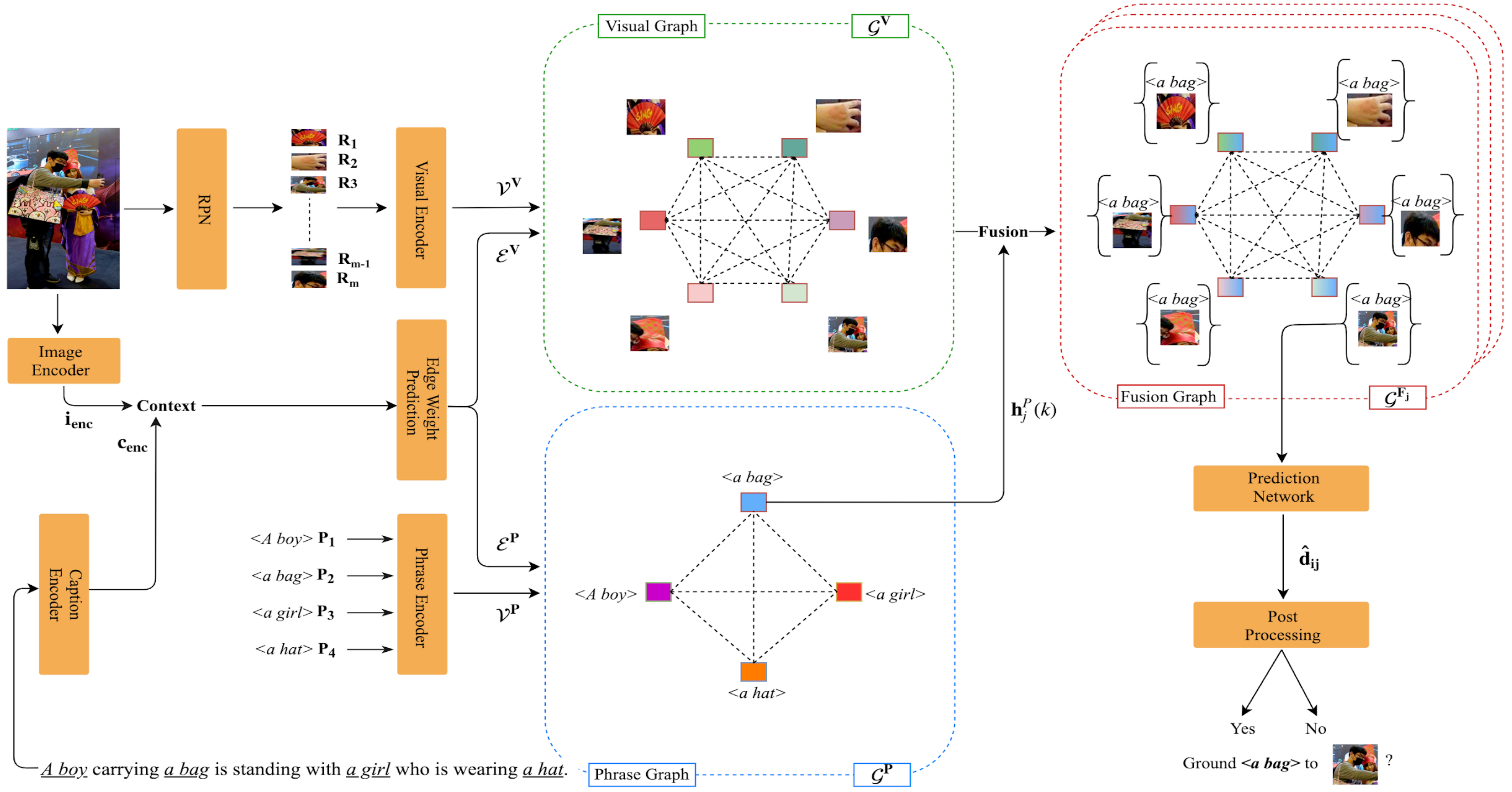
Proposed **Architecture**

$$\alpha_u = \sigma(f_{attn}(Concat(\mathbf{r}_u, \mathbf{c}_{enc}, \mathbf{i}_{enc})))$$



$$\mathbf{A}^P_{u,v} = \mathbf{A}^P_{v,u} = \sigma(f_{adj}(Concat(\mathbf{p}_u + \mathbf{p}_v, \mathbf{c}_{enc}, \mathbf{i}_{enc})))$$

Proposed **Architecture**



Experiments

Datasets

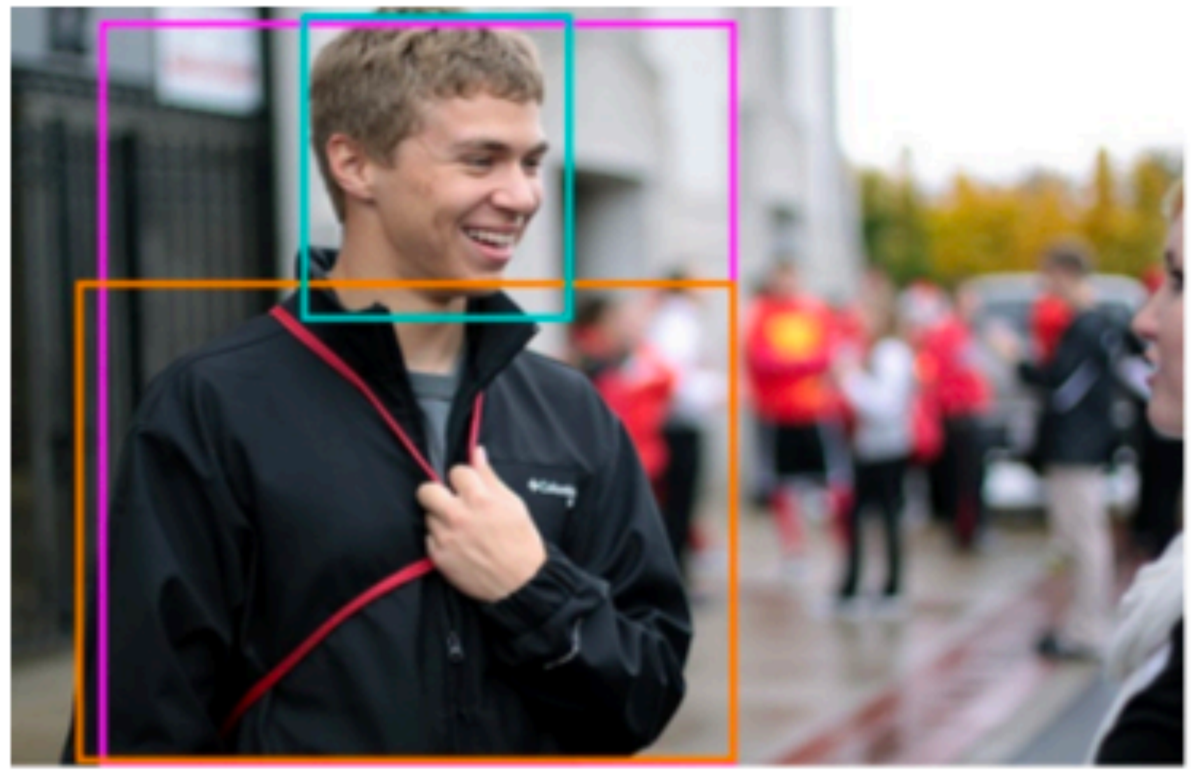
— **Flickr30K Entities**: (mostly noun) Phrases parsed from image captions

— **ReferIt Game**: Unambiguous single phrases

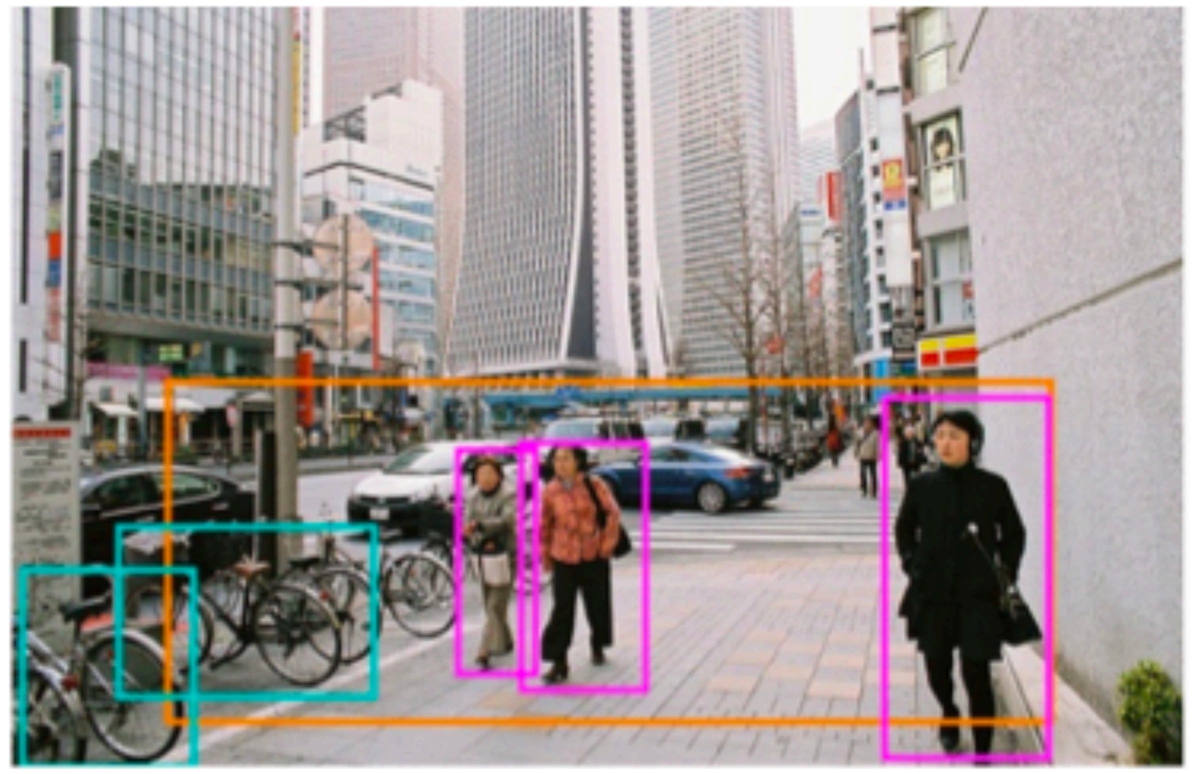
Evaluation

— Ratio of correctly grounded phrases to the total phrases

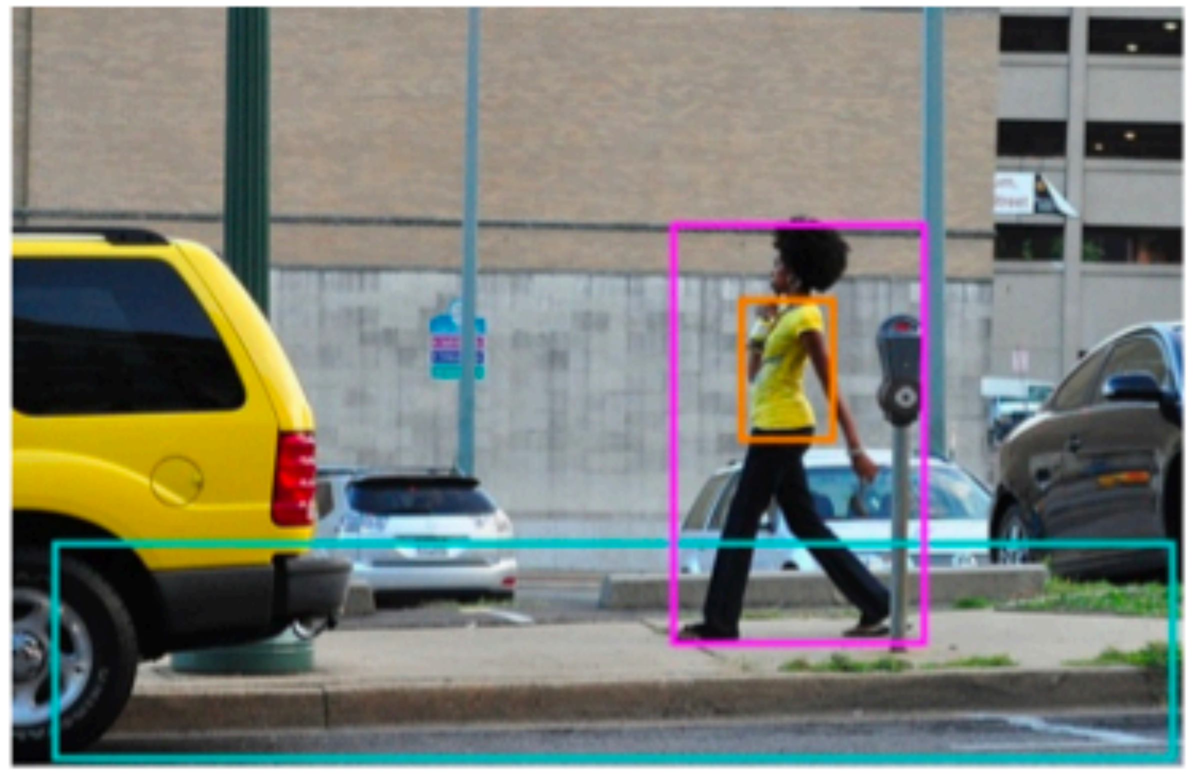
Qualitative Results: Flickr30K



(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.

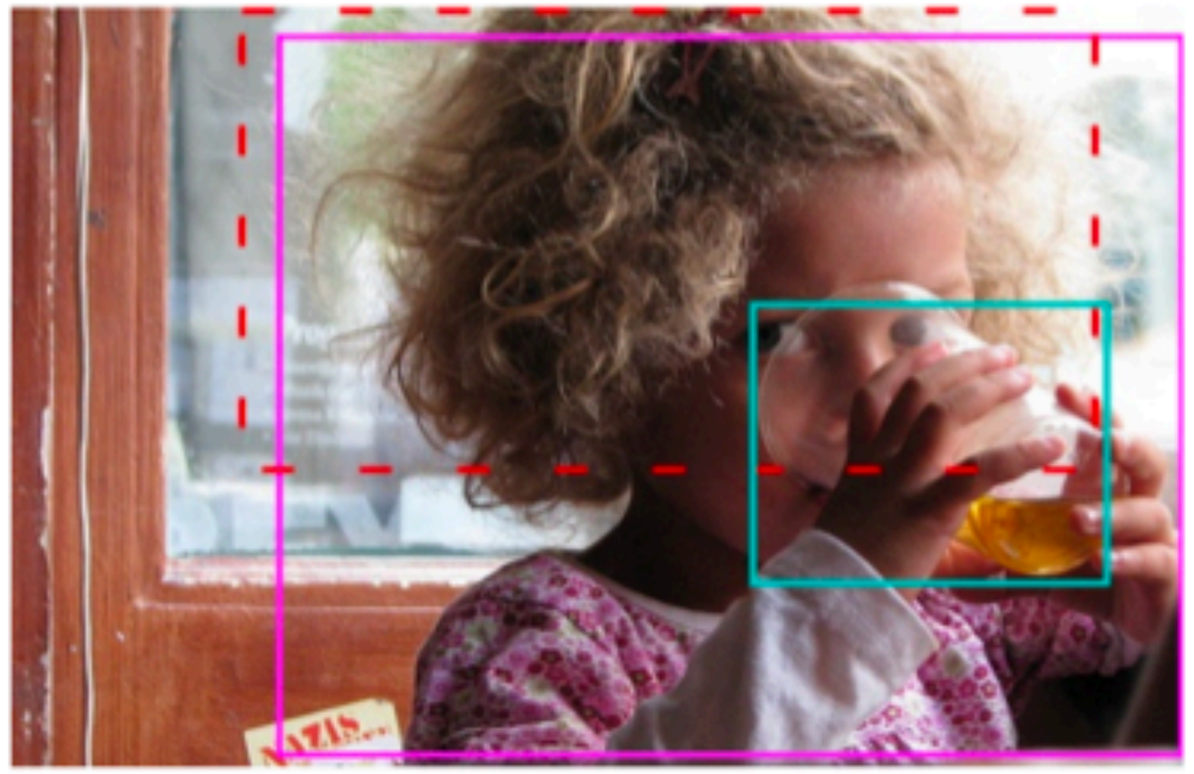


(c) A woman in a yellow shirt is walking down the sidewalk.

(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
G³RAPHGROUND++	**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^{3}RAPHGROUND++	**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^{3}RAPHGROUND++	**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**

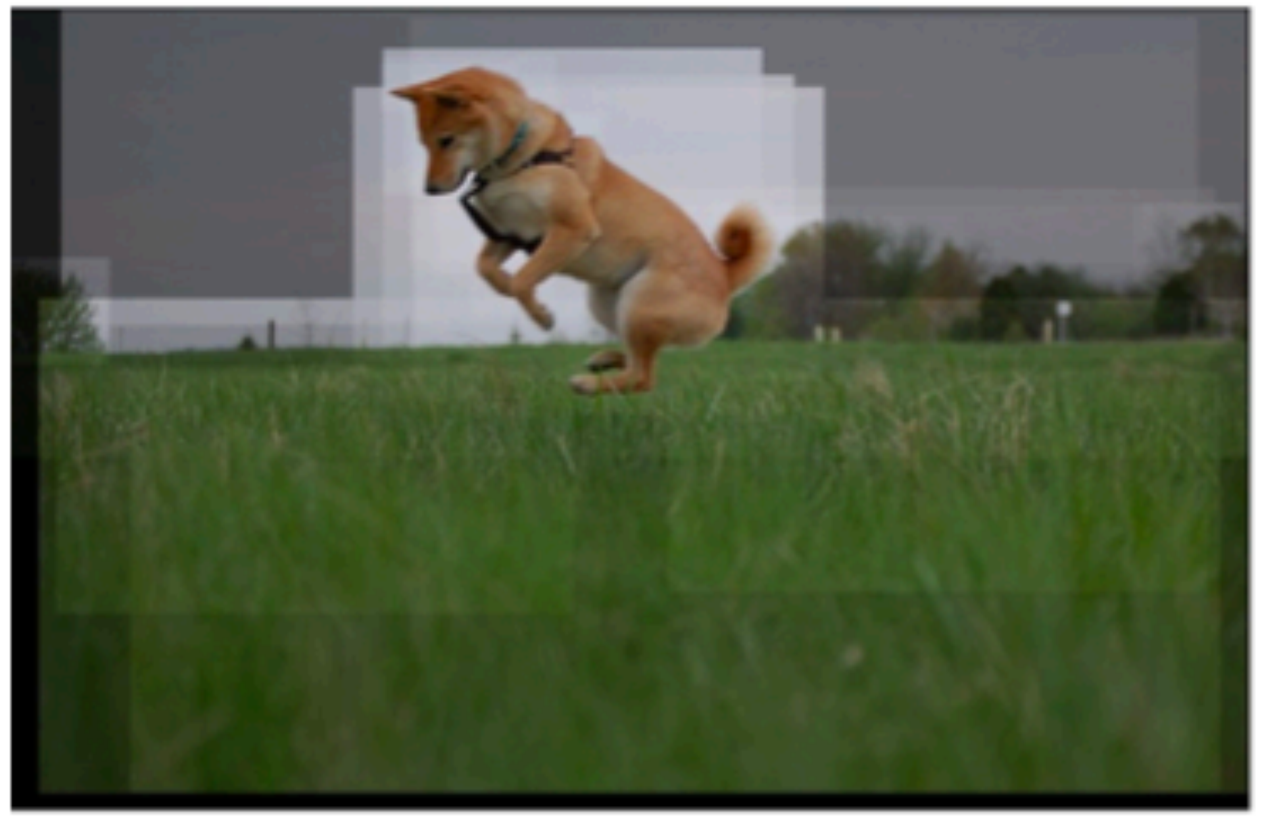
Ablation

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

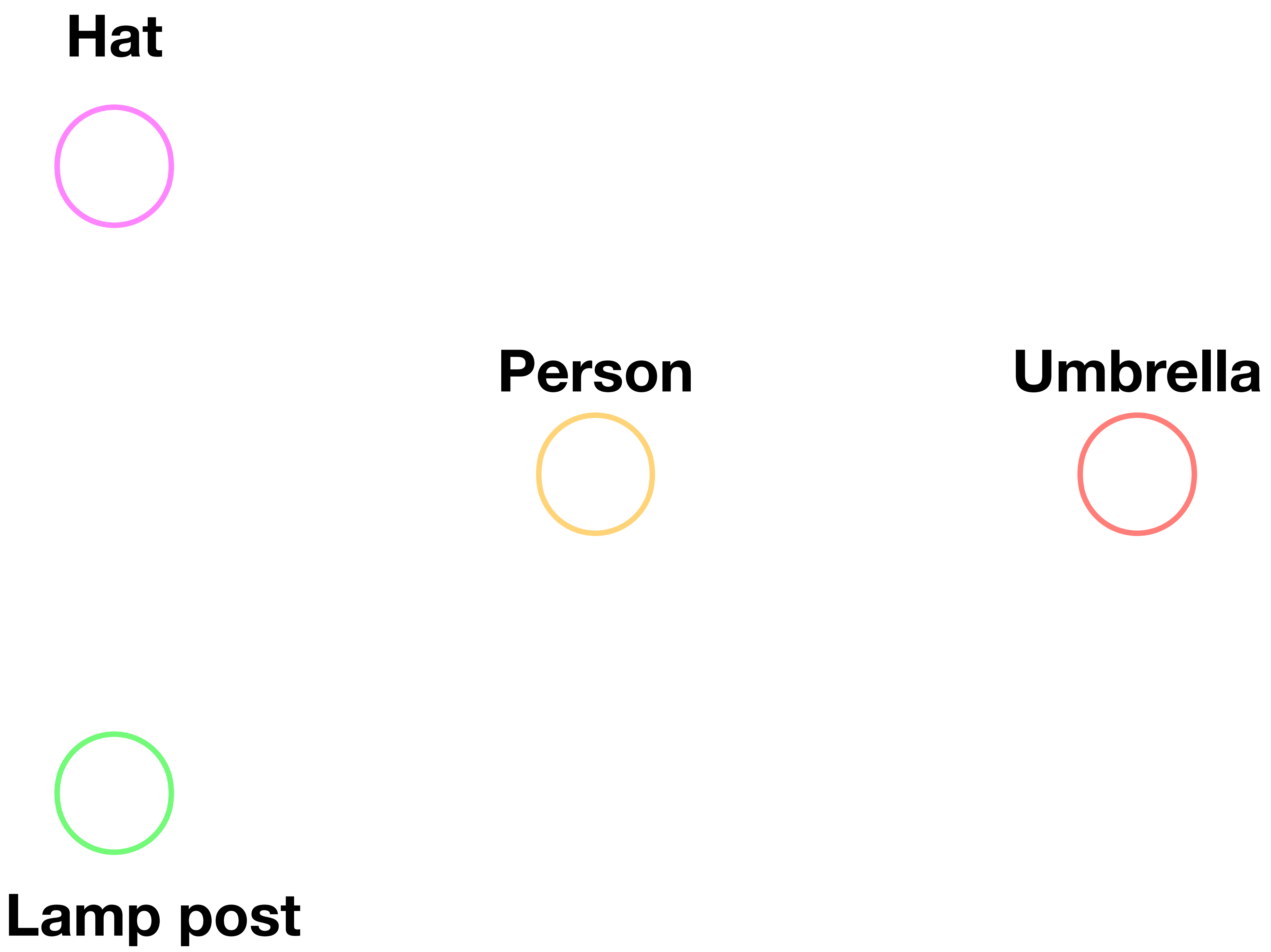
Hat

Umbrella

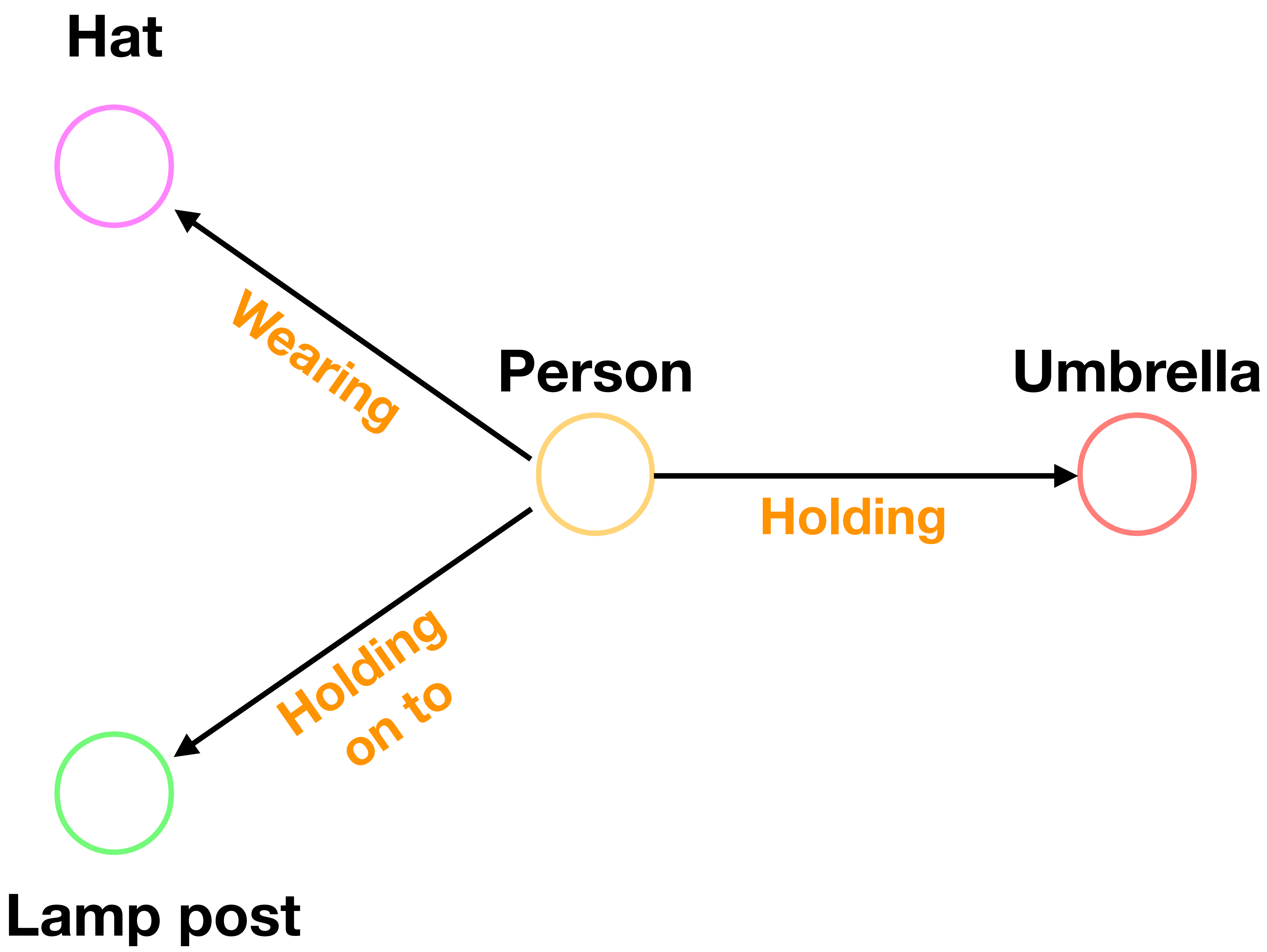
Lamp post

Person

Scene Graphs



Scene Graphs



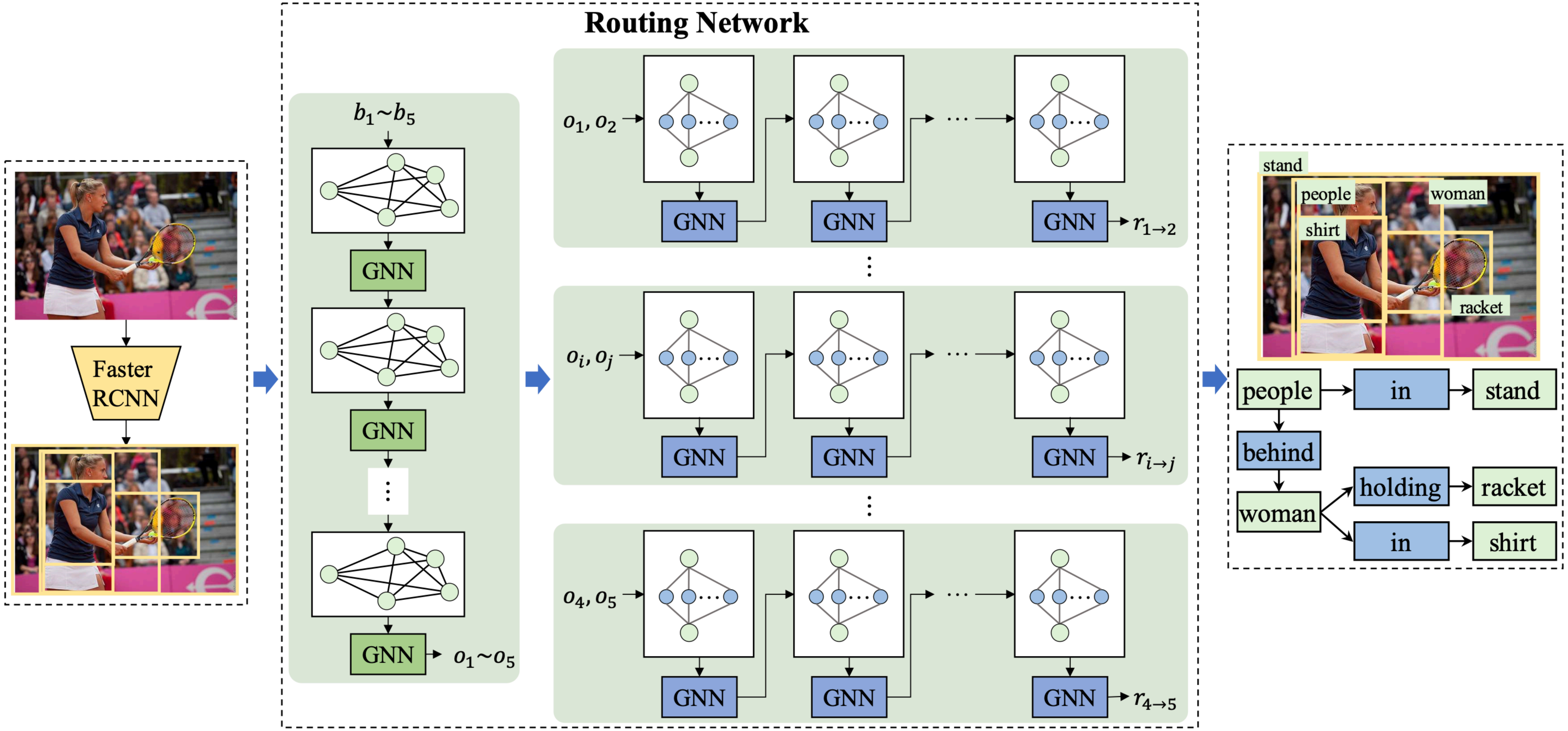
Scene Graph Generation Pipeline

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
SCENE GRAPH
CONTEXT
ENCODING
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

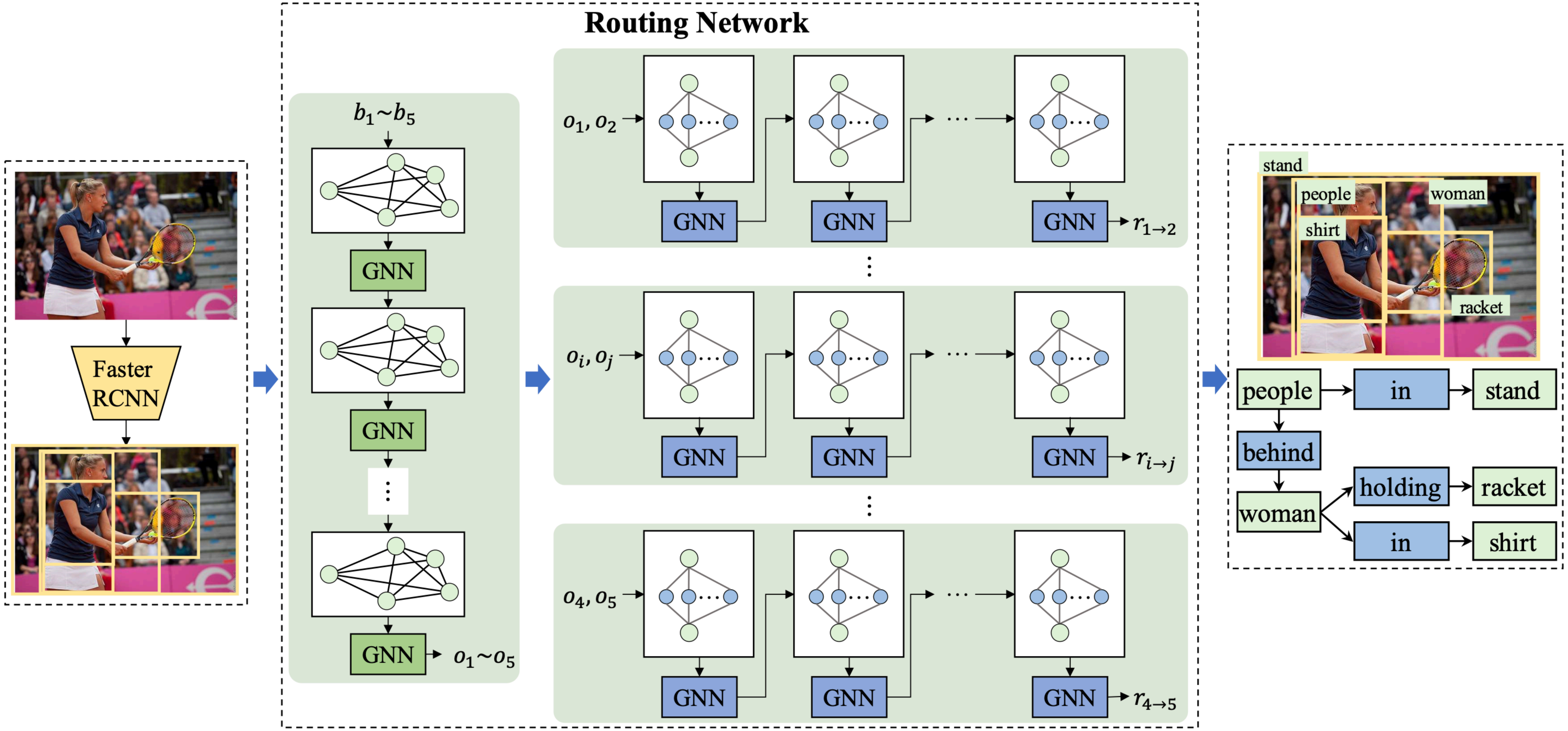
KERN Architecture

[Chen et al, 2019]



KERN Architecture

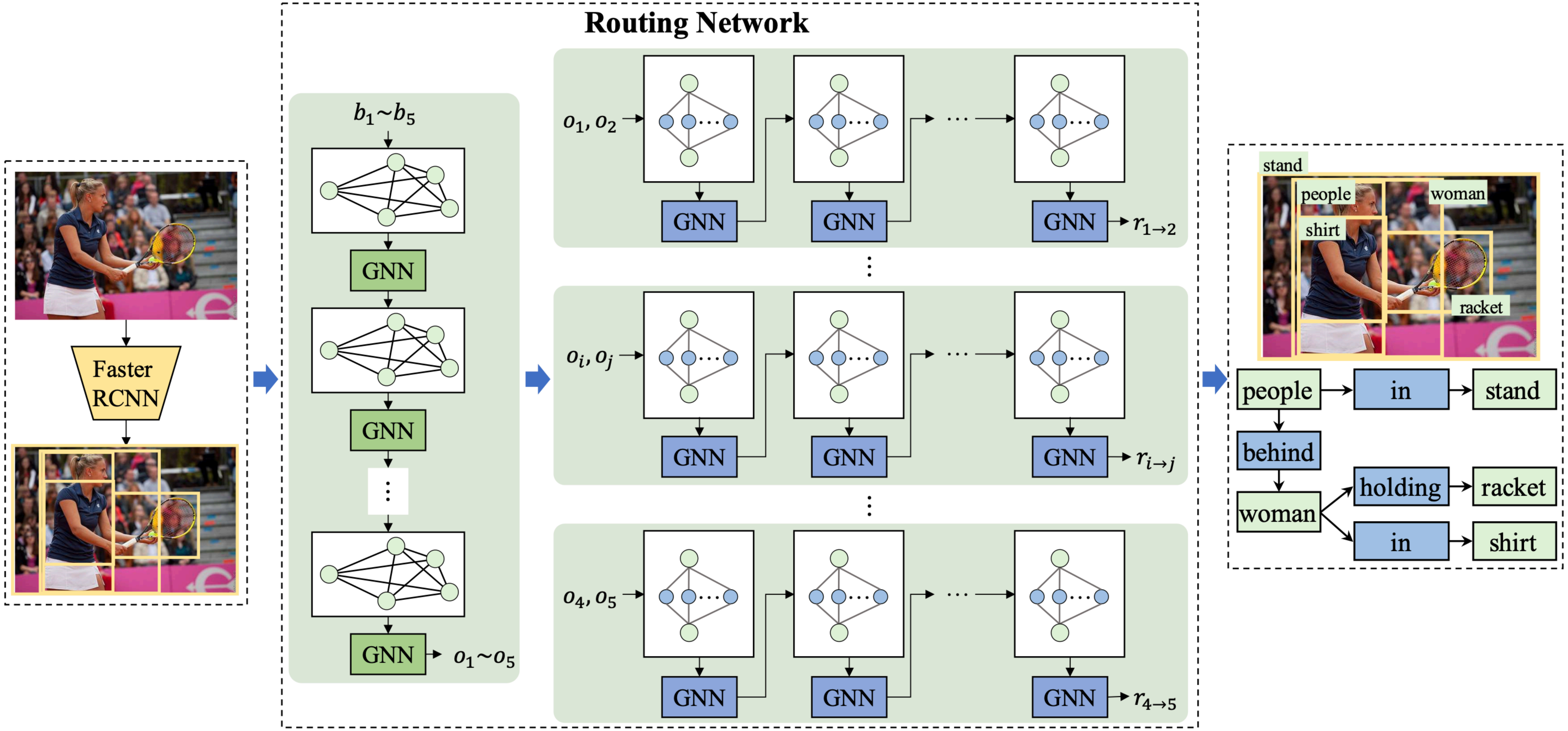
[Chen et al, 2019]



Step 1: GNN for objects (nodes are objects and edges are interactions between objects)

KERN Architecture

[Chen et al, 2019]

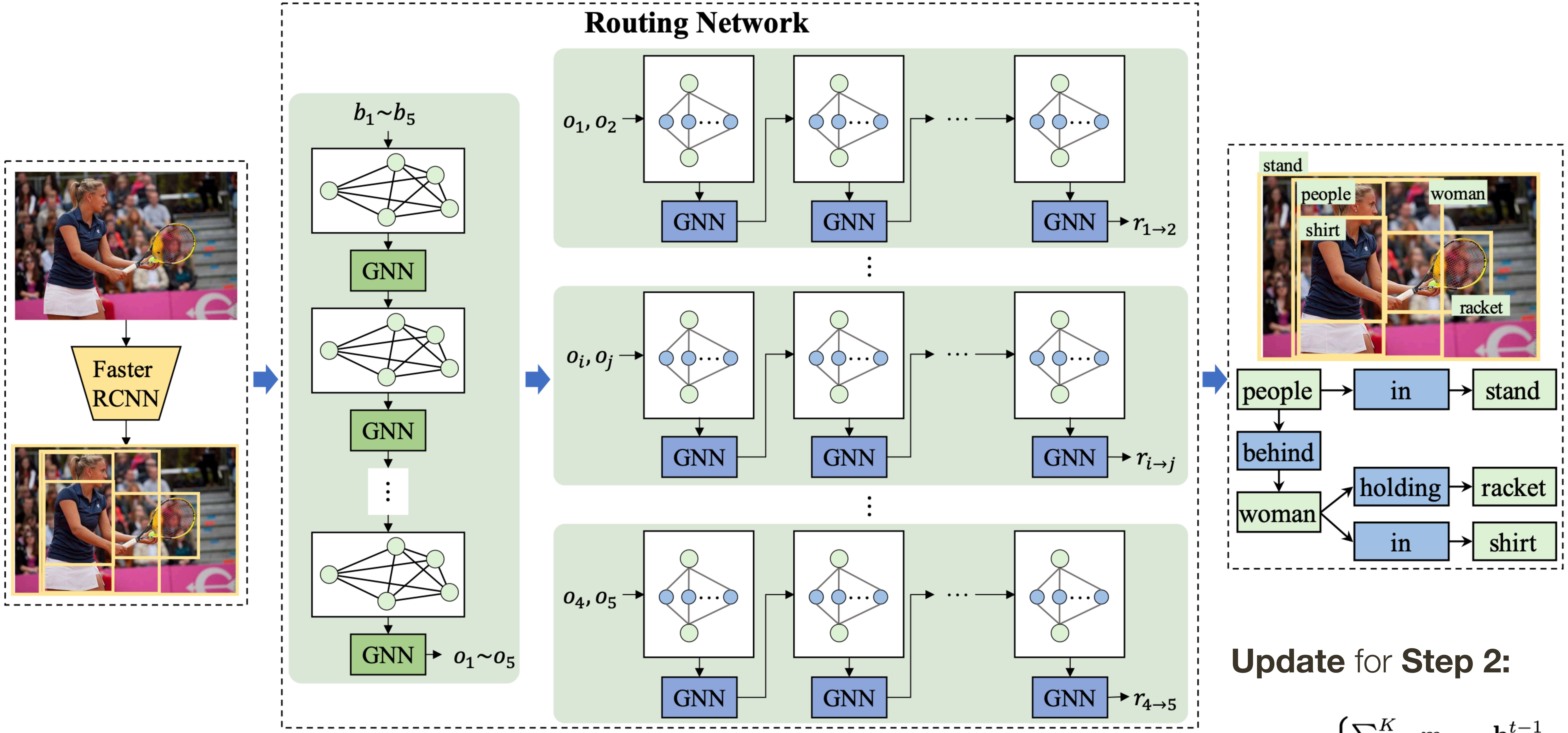


Step 1: GNN for objects (nodes are objects and edges are interactions between objects)

Step 2: GNN for each object pair, where nodes are objects and relations

KERN Architecture

[Chen et al, 2019]



Update for **Step 2:**

$$\mathbf{a}_v^t = \begin{cases} \sum_{k=1}^{K} m_{o_i o_j k} \mathbf{h}_k^{t-1} & \text{if } v \text{ is a object node} \\ m_{o_i o_j k}(\mathbf{h}_{o_i}^{t-1} + \mathbf{h}_{o_j}^{t-1}) & \text{if } v \text{ is the relationship node } k \end{cases}.$$

Step 1: GNN for objects (nodes are objects and edges are interactions between objects)

Readout for **Step 2:**

$$\mathbf{f}_v^o = o_r([\mathbf{h}_v^{T_r}, \mathbf{h}_v^0])$$
$$\mathbf{x}_{i\to j} = \phi_r([\mathbf{f}_{o_i}^o, \mathbf{f}_{o_j}^o, \mathbf{f}_1^o, \ldots, \mathbf{f}_K^o]).$$

Step 2: GNN for each object pair, where nodes are objects and relations

KERN Architecture

[Chen et al, 2019]

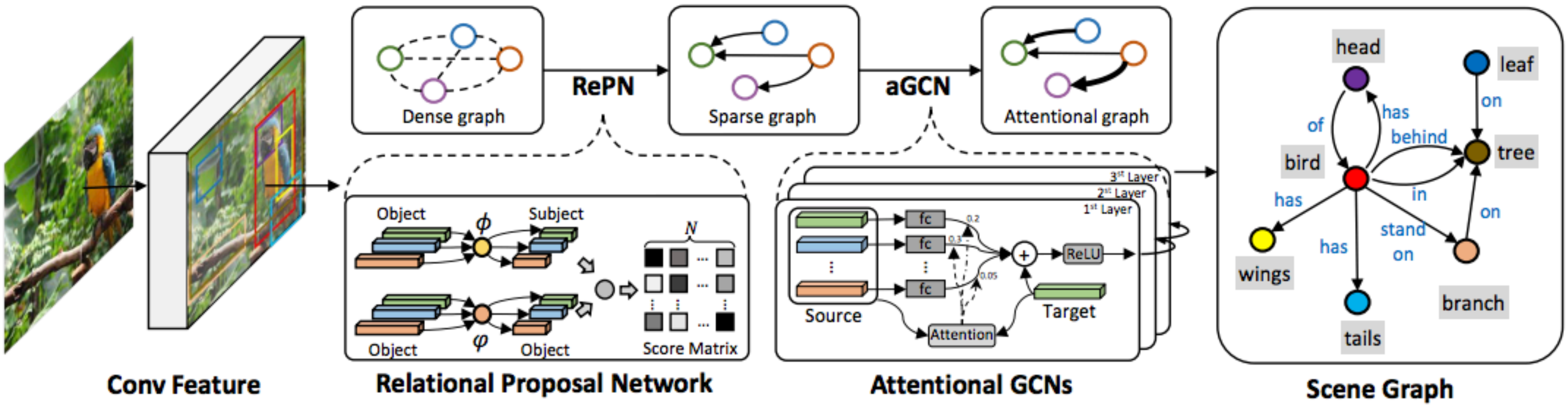
	Methods	SGGen		SGCls		PredCls		
		R@50	R@100	R@50	R@100	R@50	R@100	Mean
Constraint	VRD [19]	0.3	0.5	11.8	14.1	27.9	35.0	14.9
	IMP [30]	3.4	4.2	21.7	24.4	44.8	53.0	25.3
	IMP+ [30, 33]	20.7	24.5	34.6	35.4	59.3	61.3	39.3
	FREQ [33]	23.5	27.6	32.4	34.0	59.9	64.1	40.3
	SMN [33]	**27.2**	**30.3**	35.8	36.5	65.2	67.1	43.7
	Ours	27.1	29.8	**36.7**	**37.4**	**65.8**	**67.6**	**44.1**
No constraint	AE [23]	9.7	11.3	26.5	30.0	68.0	75.2	36.8
	IMP+ [30, 33]	22.0	27.4	43.4	47.2	75.2	83.6	49.8
	FREQ [33]	25.3	30.9	40.5	43.7	71.3	81.2	48.8
	SMN [33]	30.5	35.8	44.5	47.7	81.1	88.3	54.7
	Ours	**30.9**	**35.8**	**45.9**	**49.0**	**81.9**	**88.9**	**55.4**

Table 2. Comparison of the R@50 and R@100 in % with and without constraint on the three tasks of the VG dataset. We compute Mean R by averaging R@50 and R@100 over the three tasks.

Methods	SGGen		SGCls		PredCls		
	mR@50	mR@100	mR@50	mR@100	mR@50	mR@100	Mean
Ours w/o rk & w/o ok	5.1	5.8	6.1	6.5	10.5	11.5	7.6
Ours w/o rk	5.2	5.9	6.5	6.9	11.1	12.0	7.9
Ours	**6.4**	**7.3**	**9.4**	**10.0**	**17.7**	**19.2**	**11.7**
	R@50	R@100	R@50	R@100	R@50	R@100	Mean
Ours w/o rk & w/o ok	25.2	27.9	33.9	34.8	58.7	61.0	40.3
Ours w/o rk	25.5	28.0	34.3	35.2	59.2	61.5	40.6
Ours	**27.1**	**29.8**	**36.7**	**37.4**	**65.8**	**67.6**	**44.1**

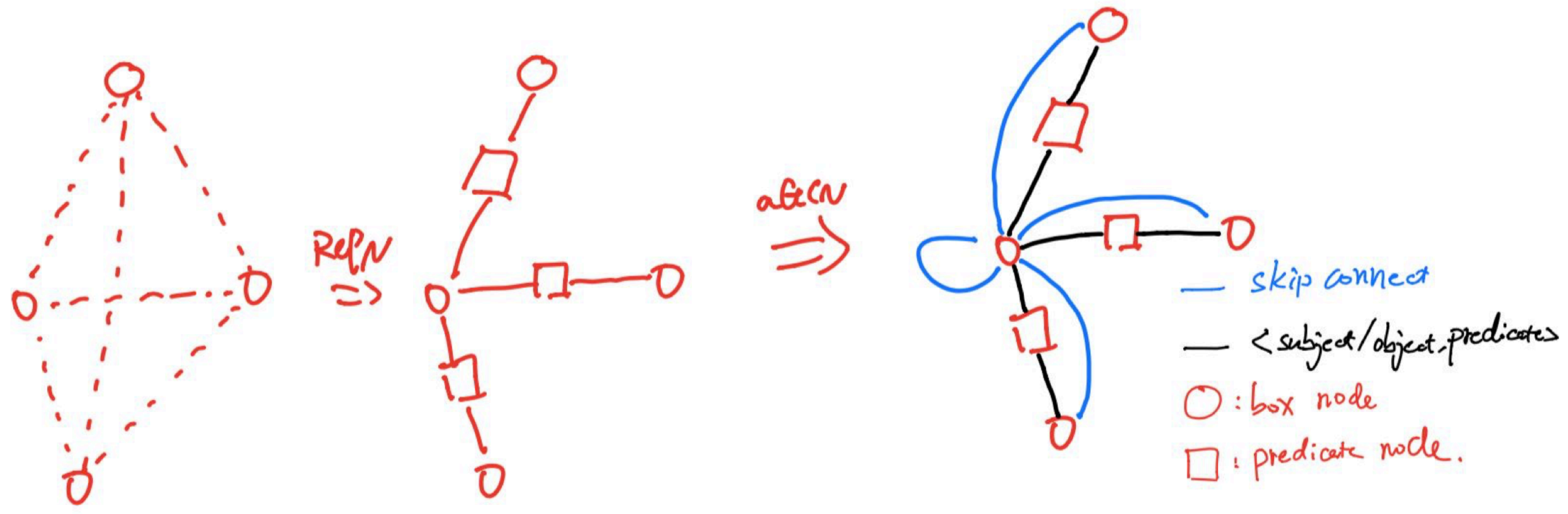
Table 3. Comparison of the mR@50, mR@100 (above) and the R@50, R@100 (below) with constraint in % of our full model, our model without relationship correlation (w/o rc), and our model without relationship correlation and object correlation (w/o rc & oc). We compute Mean mR by averaging mR@50 and mR@100 over the three tasks and mean R in the same way.

Graph RCNN



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

Graph RCNN



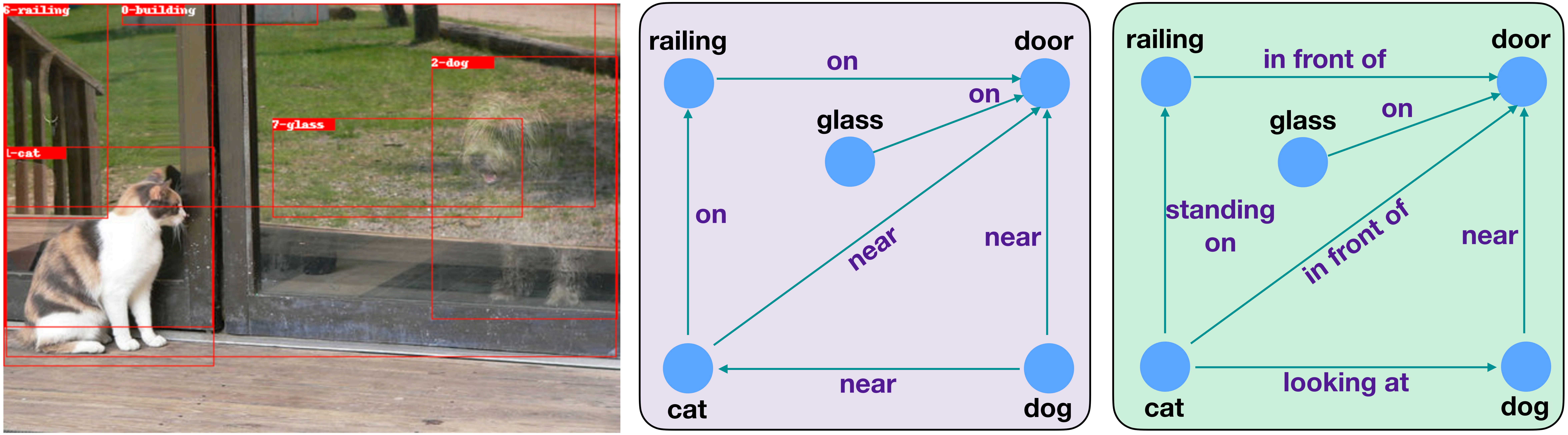
$$z_i^o = \sigma(\overbrace{W^{\texttt{skip}} Z^o \boldsymbol{\alpha}^{\texttt{skip}}}^{\substack{\text{Message from}\\ \text{Other Objects}}} + \overbrace{W^{sr} Z^r \boldsymbol{\alpha}^{sr} + W^{or} Z^r \boldsymbol{\alpha}^{or}}^{\substack{\text{Messages from}\\ \text{Neighboring Relationships}}})$$

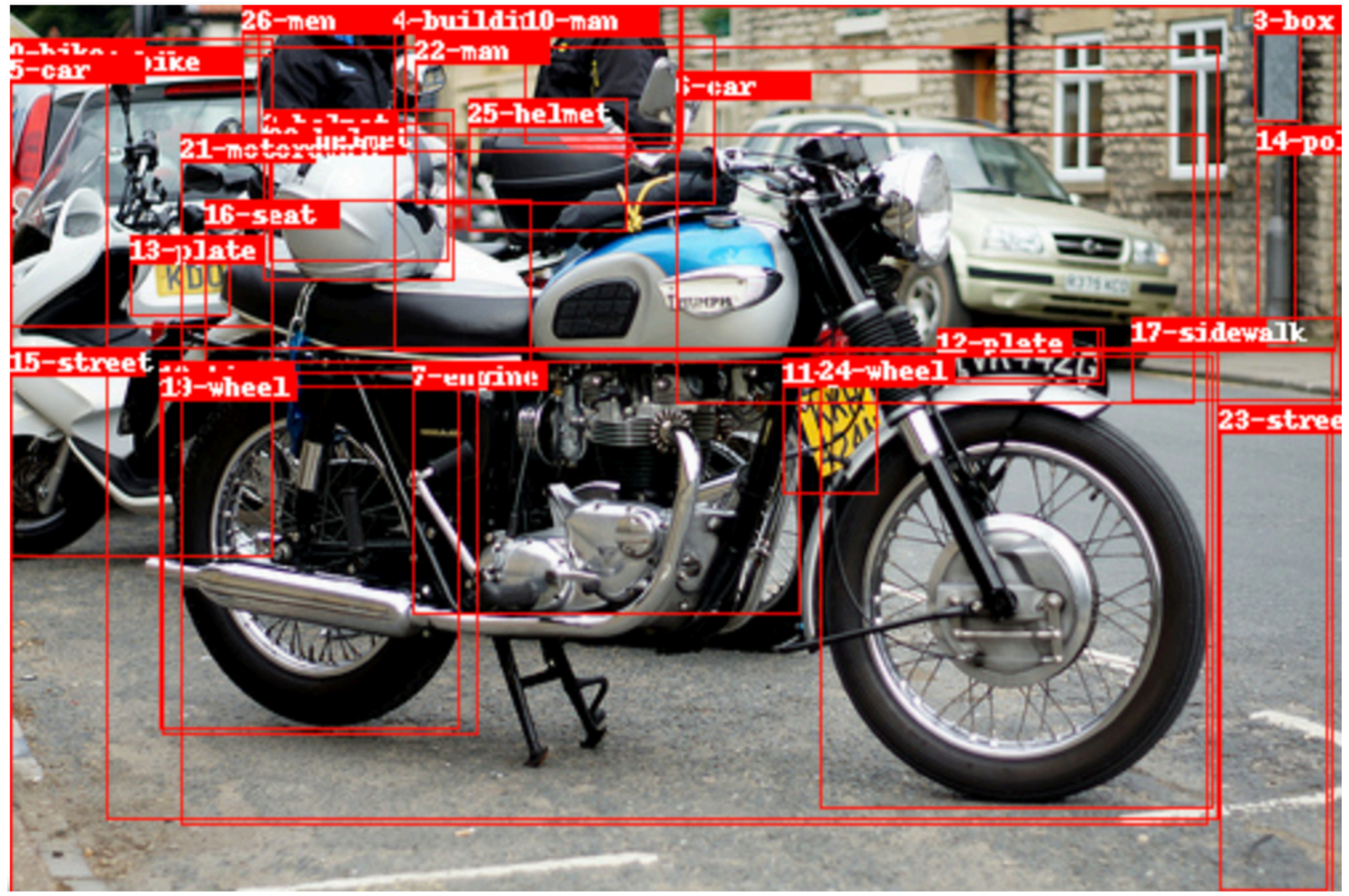
$$z_i^r = \sigma(z_i^r + \underbrace{W^{rs} Z^o \boldsymbol{\alpha}^{rs} + W^{ro} Z^o \boldsymbol{\alpha}^{ro}}_{\text{Messages from Neighboring Objects}}).$$


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

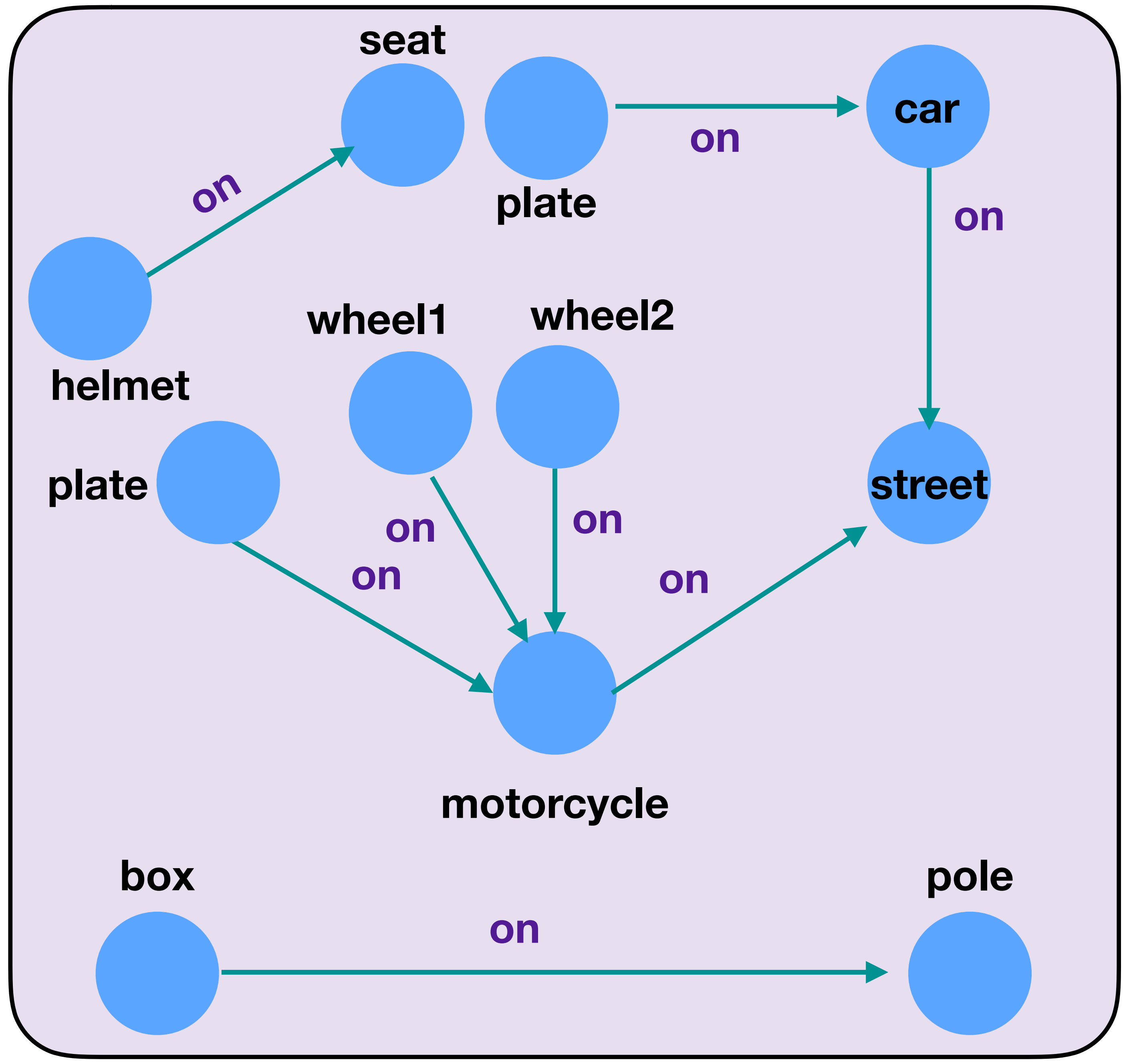

OBJECT DETECTOR
FEATURE EXTRACTION
STATE INITIALIZATION
CONTEXT
ENCODING
SCENE GRAPH
person
standing on
holding
beach
surfboard
FEATURE EMBEDDING
LABEL EMBEDDING
LABEL
PREDICTION
person
beach
surfboard
FEATURE EMBEDDING
IMAGE GRAPH
person
standing on
holding
beach
surfboard
EGNN
GNN
EDGED GATED POOLING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

Visualizations

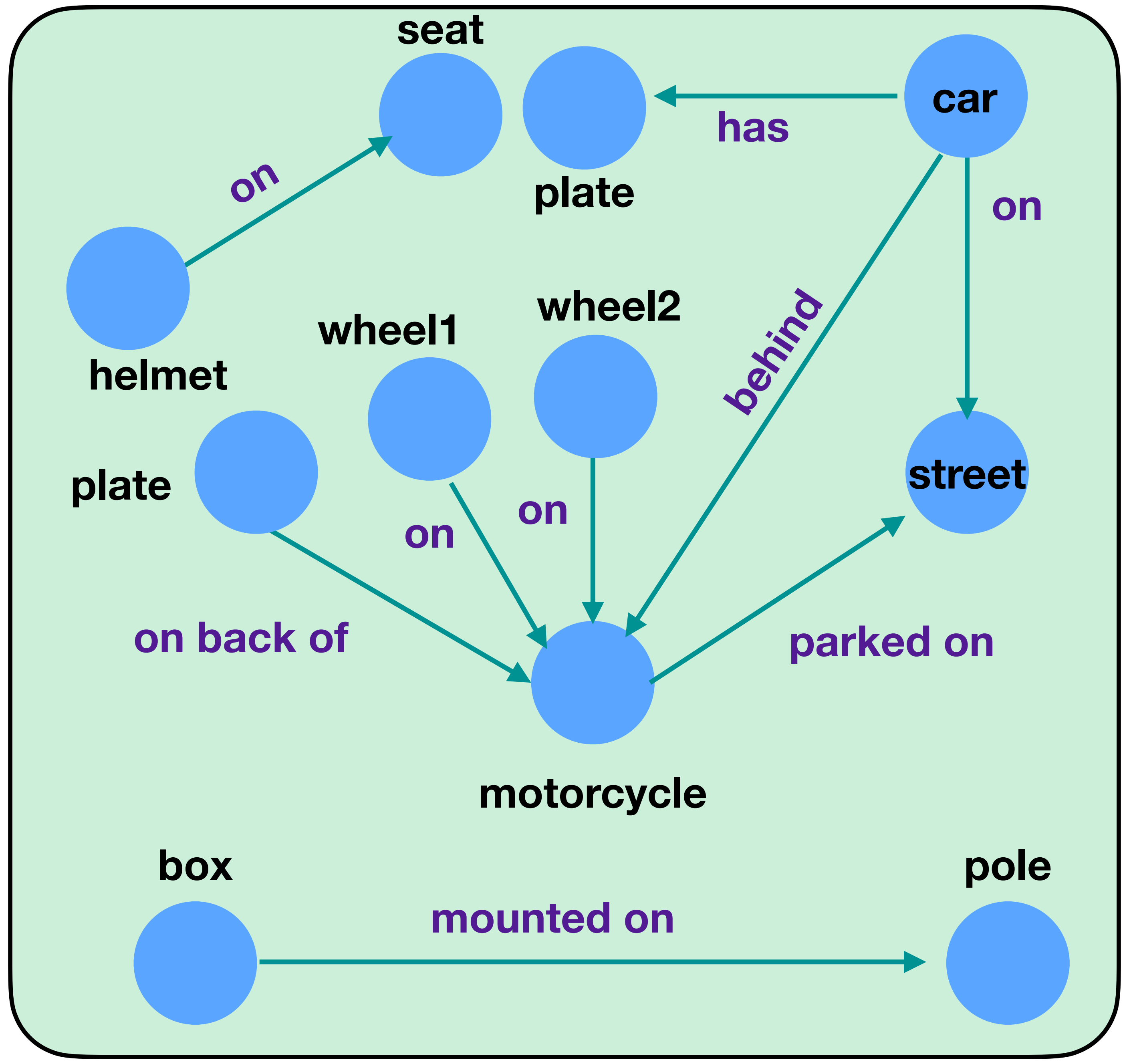

6-railing
0-building
2-dog
7-glass
1-cat
railing
door
on
glass
on
on
near
near
cat
near
dog
railing
in front of
door
glass
on
standing
on
in front of
near
cat
looking at
dog



26-men
22-man
3-box
5-car
25-helmet
16-seat
13-plate
14-pol
15-street
13-wheel
7-engine
12-plate
17-sidewalk
23-stree



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



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

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
on
near
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

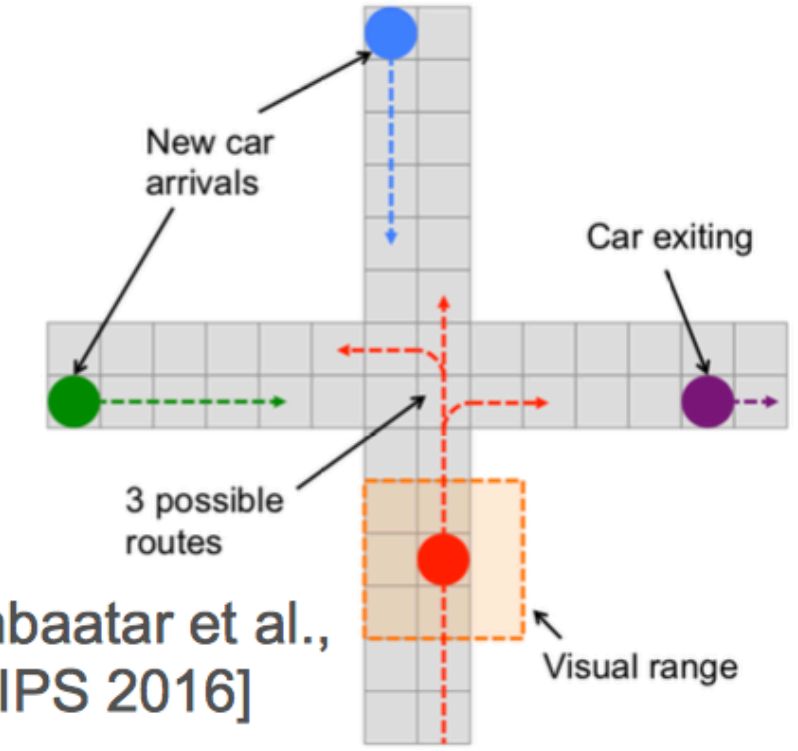
— **Deep learning on graphs works and is very effective!**

— 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 <u>billion</u> 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**

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

Lecture 21: Deep Reinforcement Learning

Types of **Learning**

Supervised training

— Learning from the teacher

— Training data includes desired output

Unsupervised training

— Training data does not include desired output

Reinforcement learning

— Learning to act under evaluative feedback (rewards)

* slide from Dhruv Batra

What is **Reinforcement Learning**

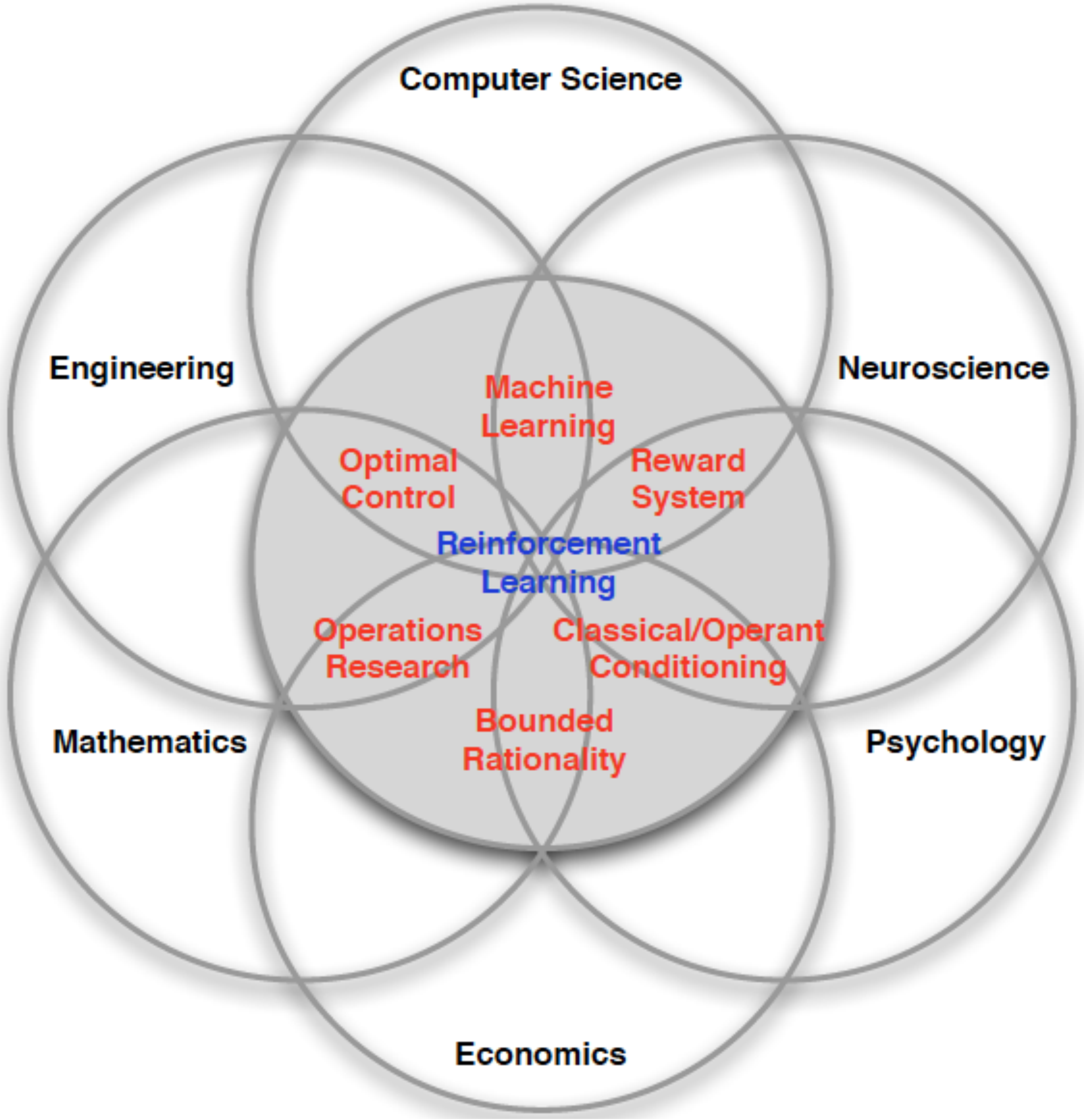
Agent-oriented learning — learning by interacting with an environment to achieve a goal

— More realistic and ambitious than other kinds of machine learning

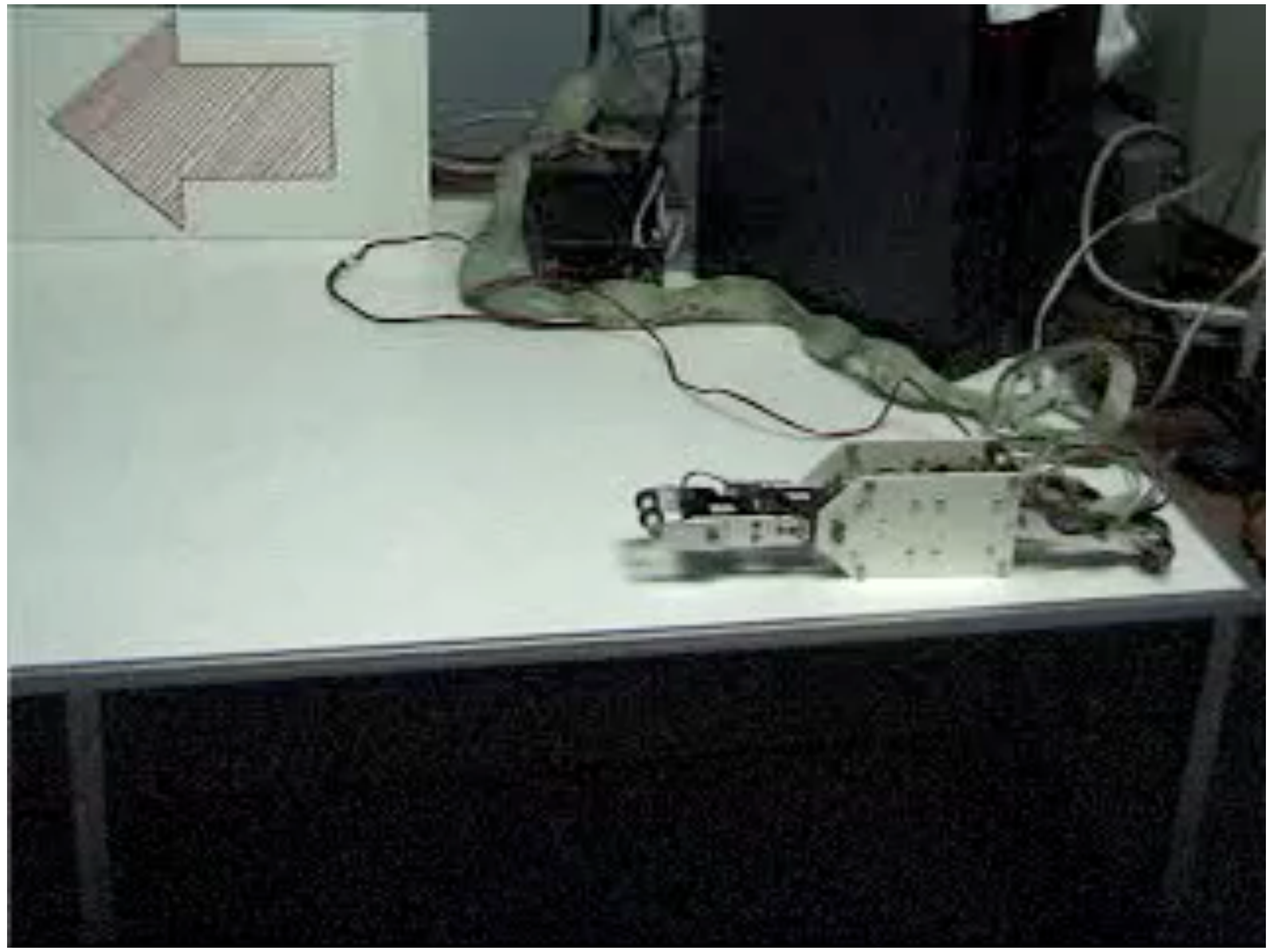
Learning **by trial and error**, with only delayed evaluative feedback (reward)

— The kind go machine learning most like natural learning

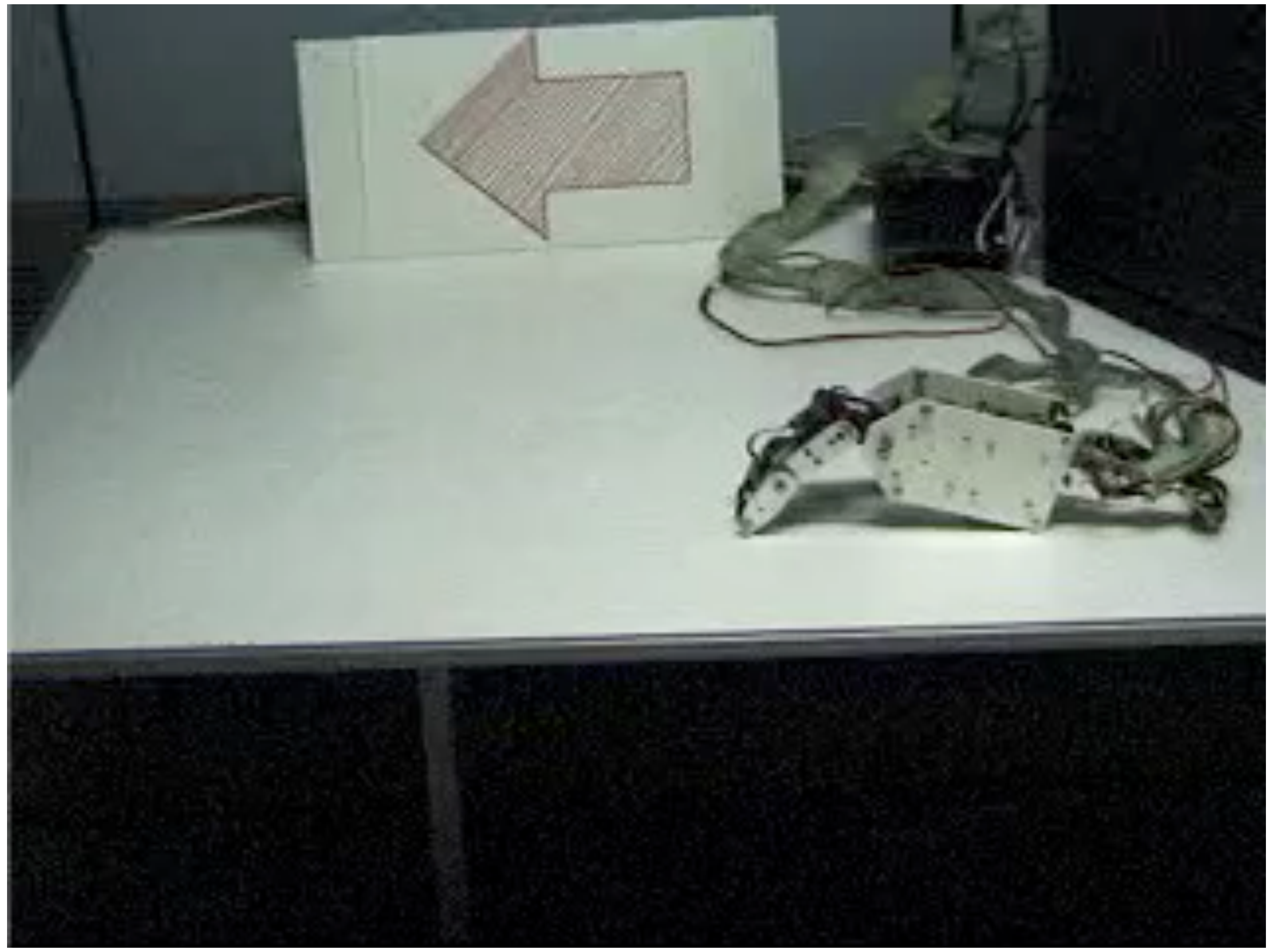
— Learning that can tell for itself when it is right or wrong



Example: Hajime Kimura's RL Robot



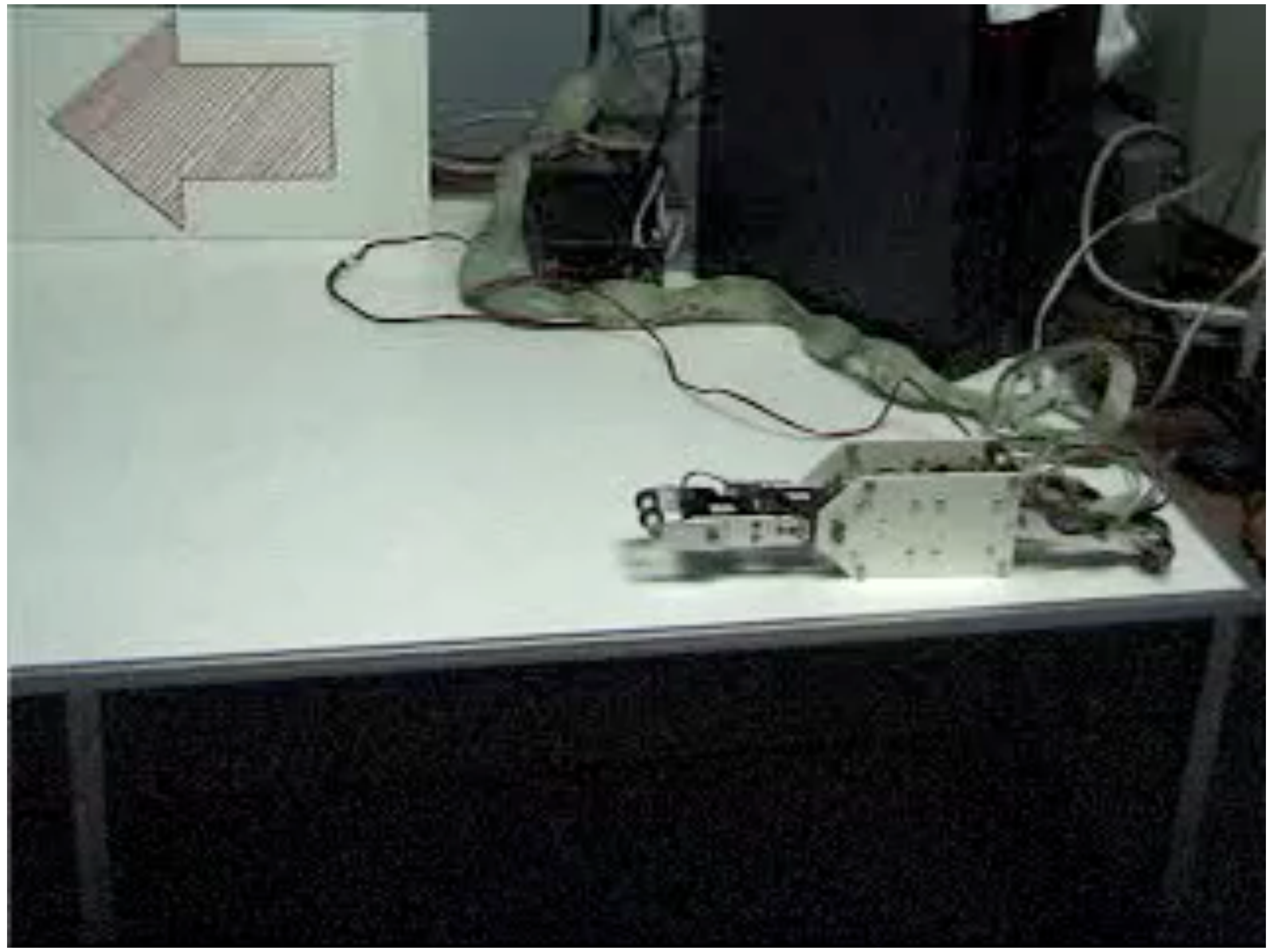
Before



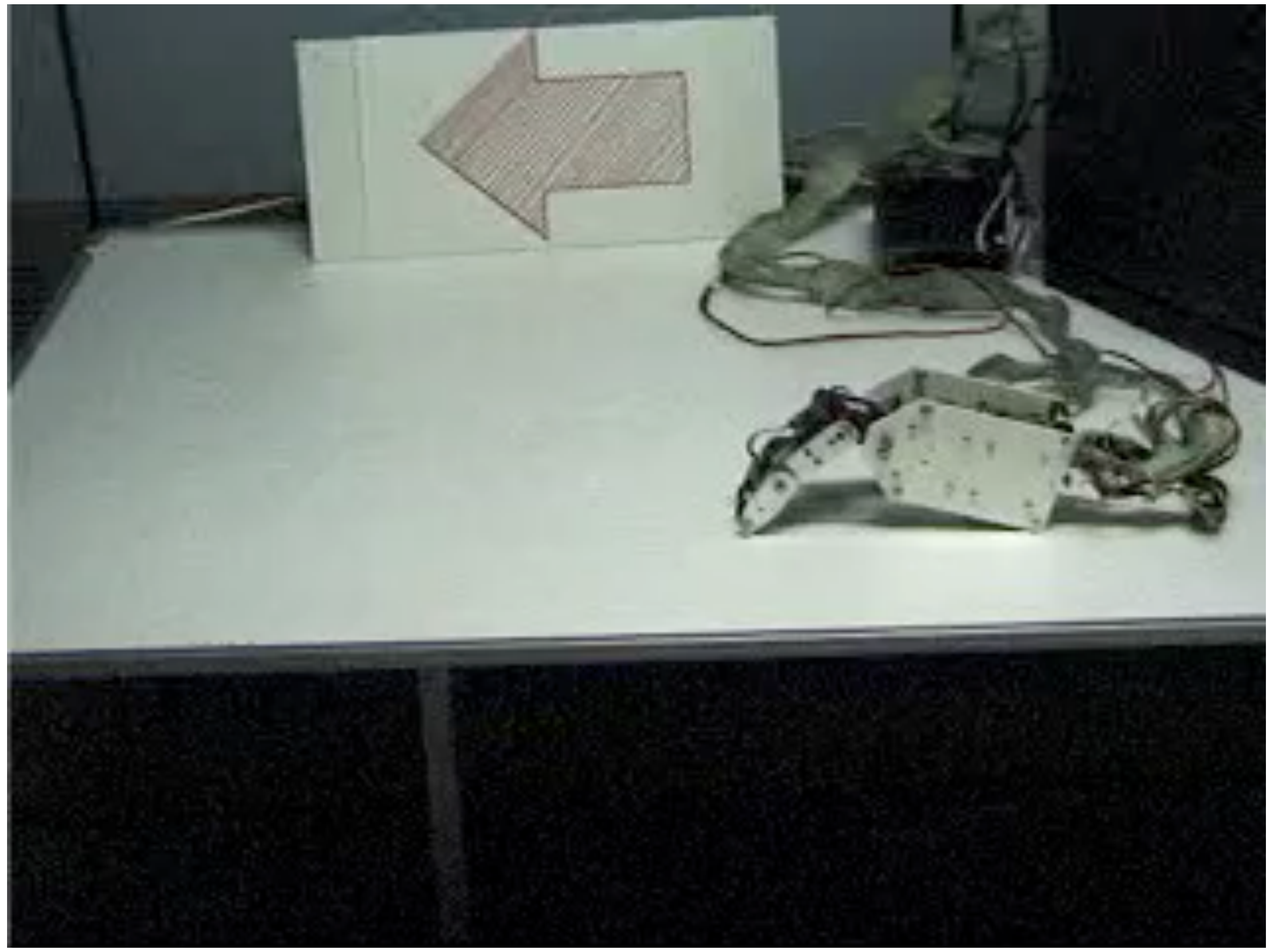
After

* slide from Rich Sutton

Example: Hajime Kimura's RL Robot



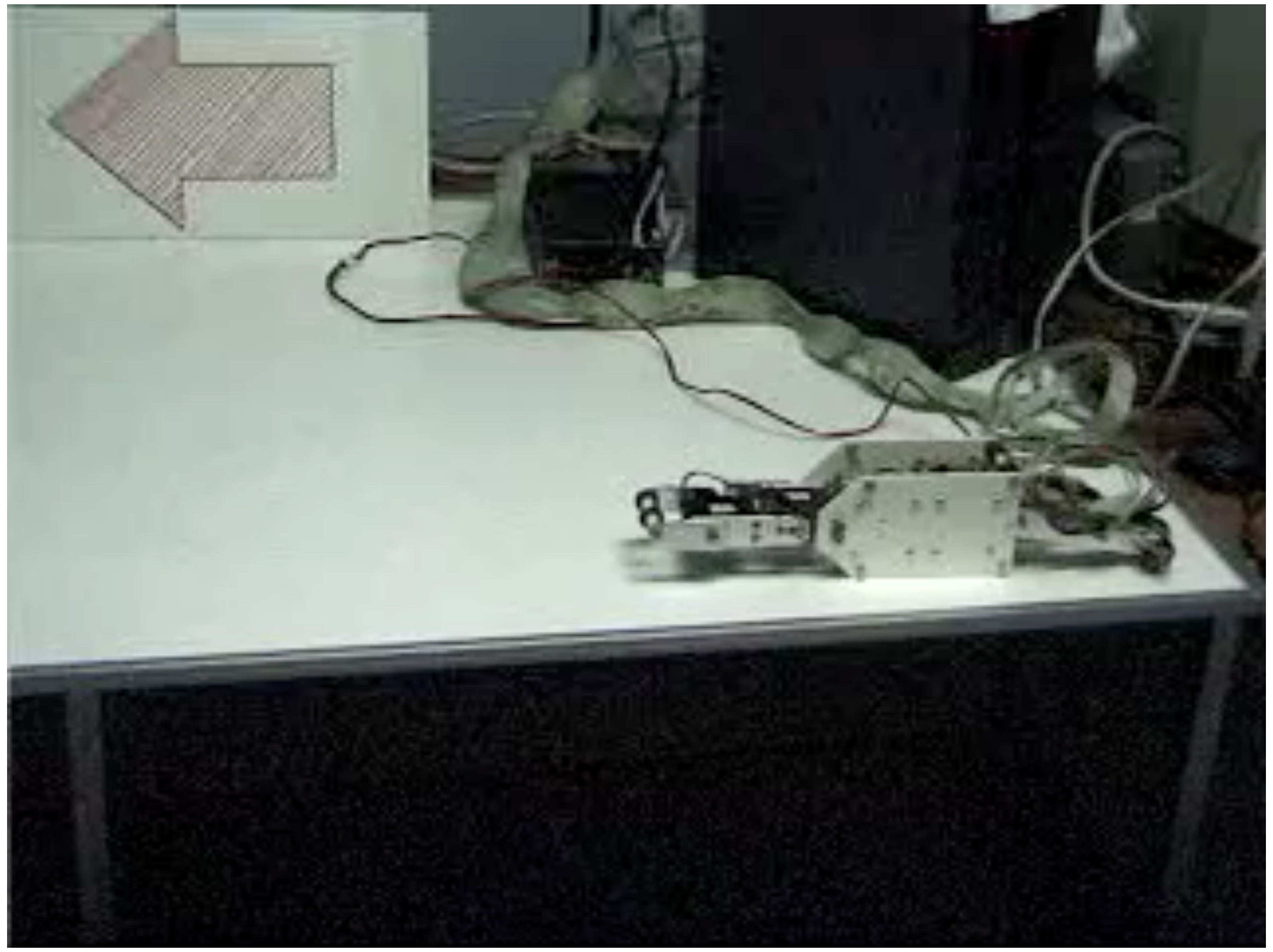
Before



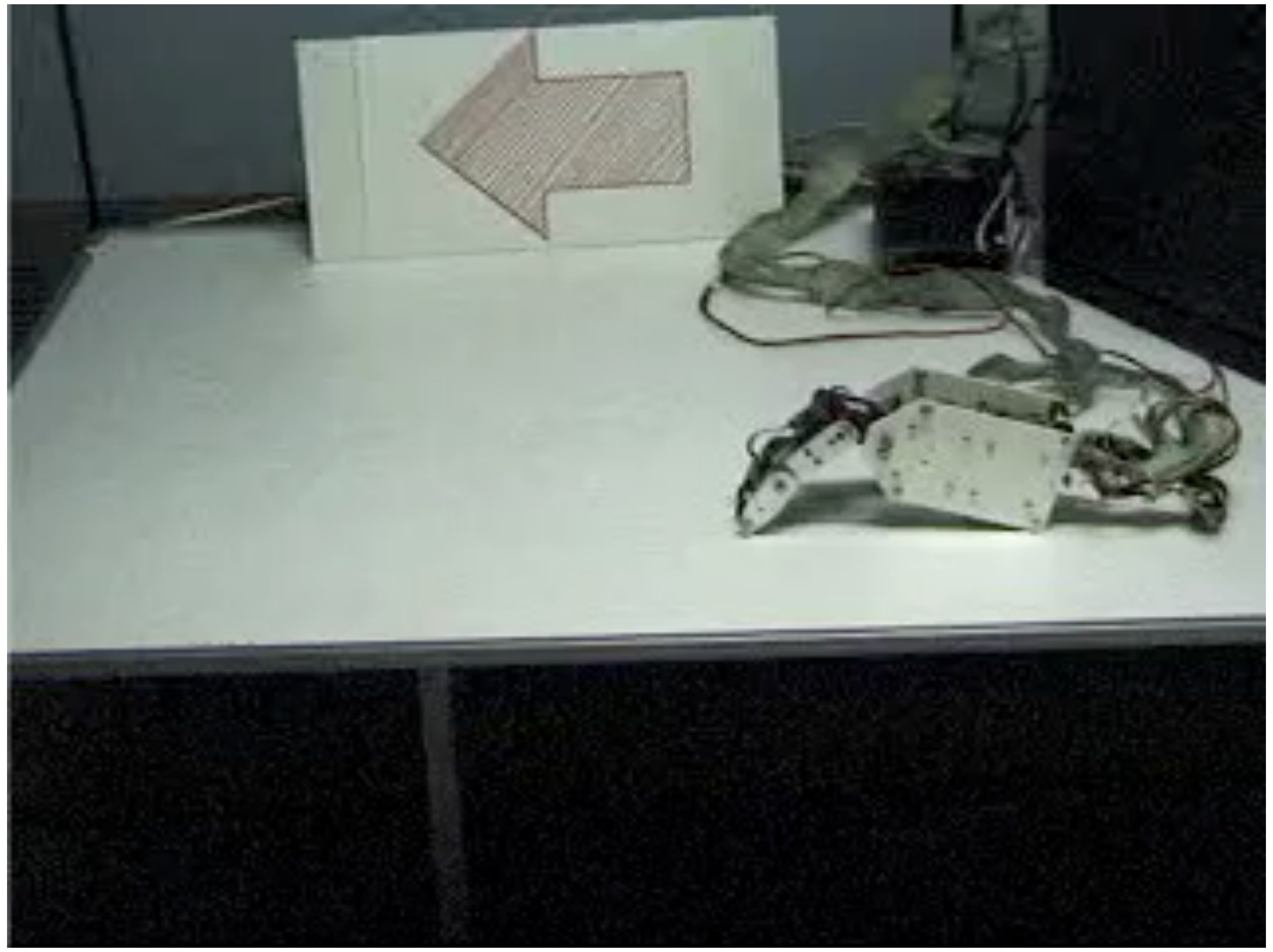
After

* slide from Rich Sutton

Example: Hajime Kimura's RL Robot



Before



After

* slide from Rich Sutton

Human Objectives

Jurgen Schmidhuber

Human Objectives

"AGI will surpass humans' in 2050, enabling robots to have fun, fall in love — and colonize the galaxy"

Jurgen Schmidhuber

Human Objectives

“AGI will surpass humans’ in 2050, enabling robots to have fun, fall in love — and colonize the galaxy”

Don’t worry about it — “They will pay as much attention to us as we do to ants”

Jurgen Schmidhuber

Human Objectives

"I think it is just the product of a few principles that will be considered very simple in hindsight, so simple that even kids will be able to understand and build intelligent, continually learning, more and more general problem solvers."

Jurgen Schmidhuber

Human Objectives

"I think it is just the product of a few principles that will be considered very simple in hindsight, so simple that even kids will be able to understand and build intelligent, continually learning, more and more general problem solvers."

High Level Objectives: Maximize Happiness, Don't Die

What would be an emergent behavior would evolve

if we have these high level objectives?

Jurgen Schmidhuber

Peril of AGI

AGI does not need to be evil to act nefariously

High level objective: Help human race to live and prosper

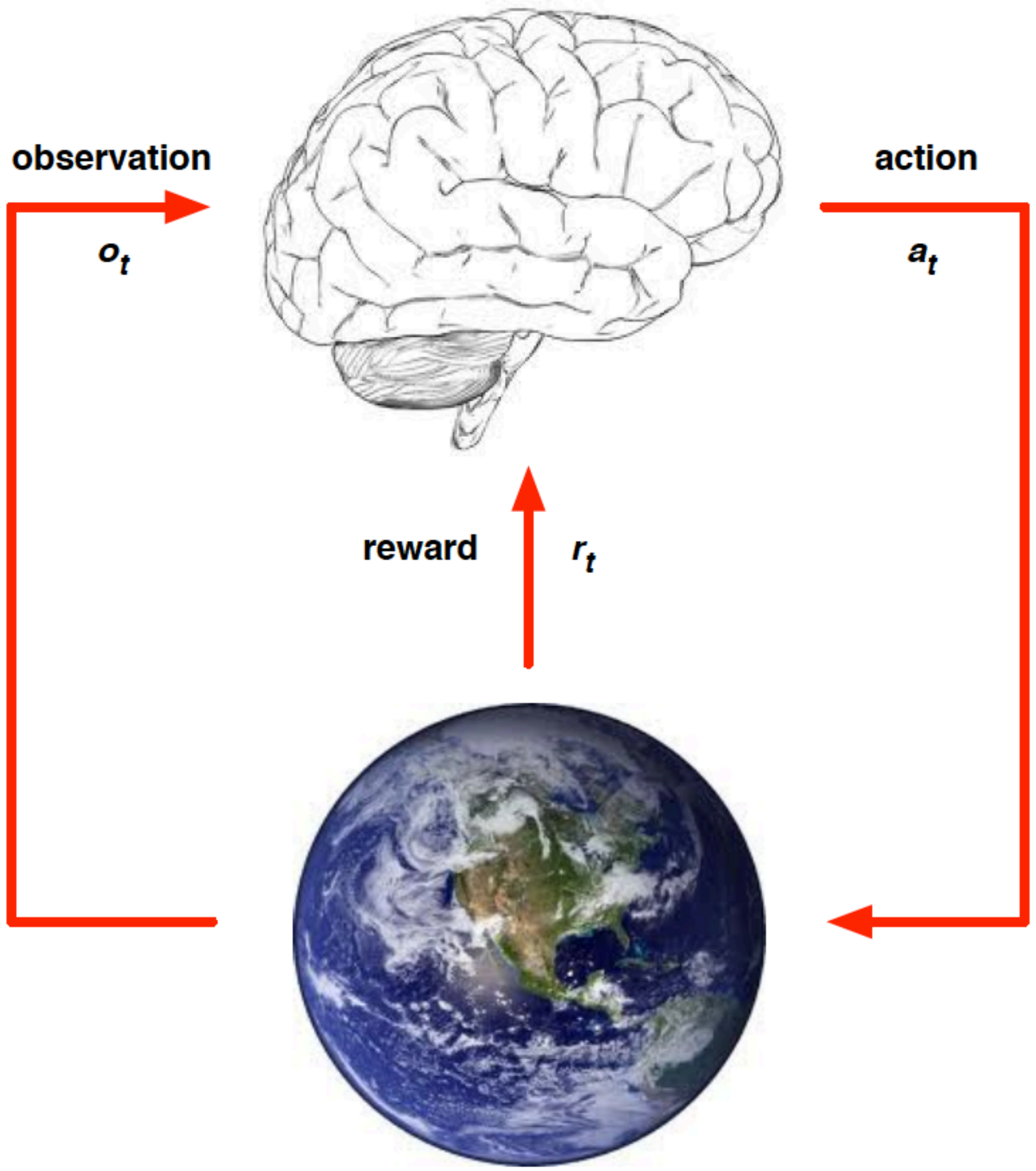
Emergent behavior: AGI self-preservation and greed

Challenges of RL

- Evaluative feedback (reward)
- Sequentiality, delayed consequences
- Need for trial and error, to explore as well as exploit
- Non-stationarity
- The fleeting nature of time and online data

* slide from Rich Sutton

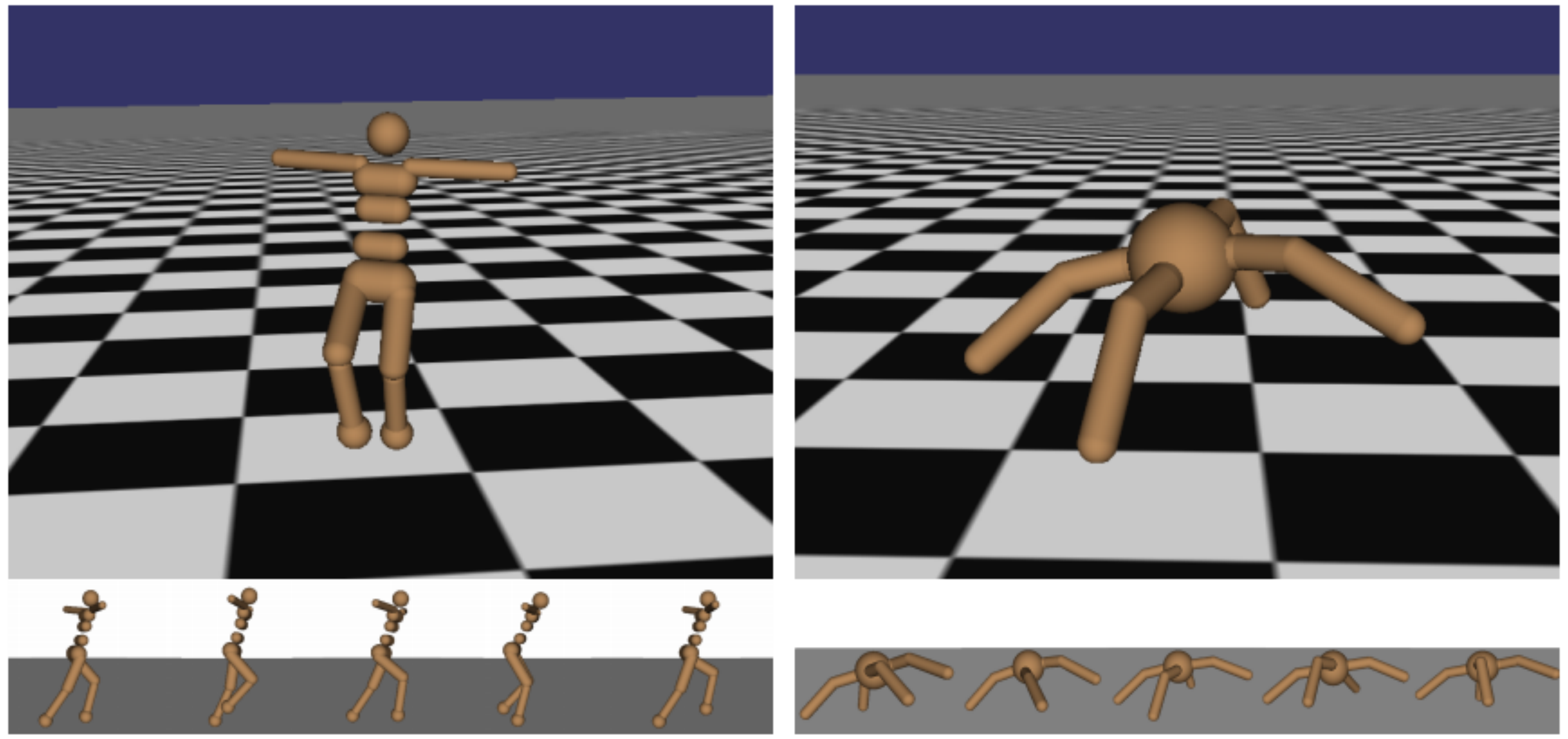
How does **RL** work?



- At each step t the agent:
  - Executes action a_t
  - Receives observation o_t
  - Receives scalar reward r_t
- The environment:
  - Receives action a_t
  - Emits observation o_{t+1}
  - Emits scalar reward r_{t+1}

* slide from David Silver

Robot Locomotion



Objective: Make the robot move forward

State: Angle and position of the joints
Action: Torques applied on joints
Reward: 1 at each time step upright + forward movement

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

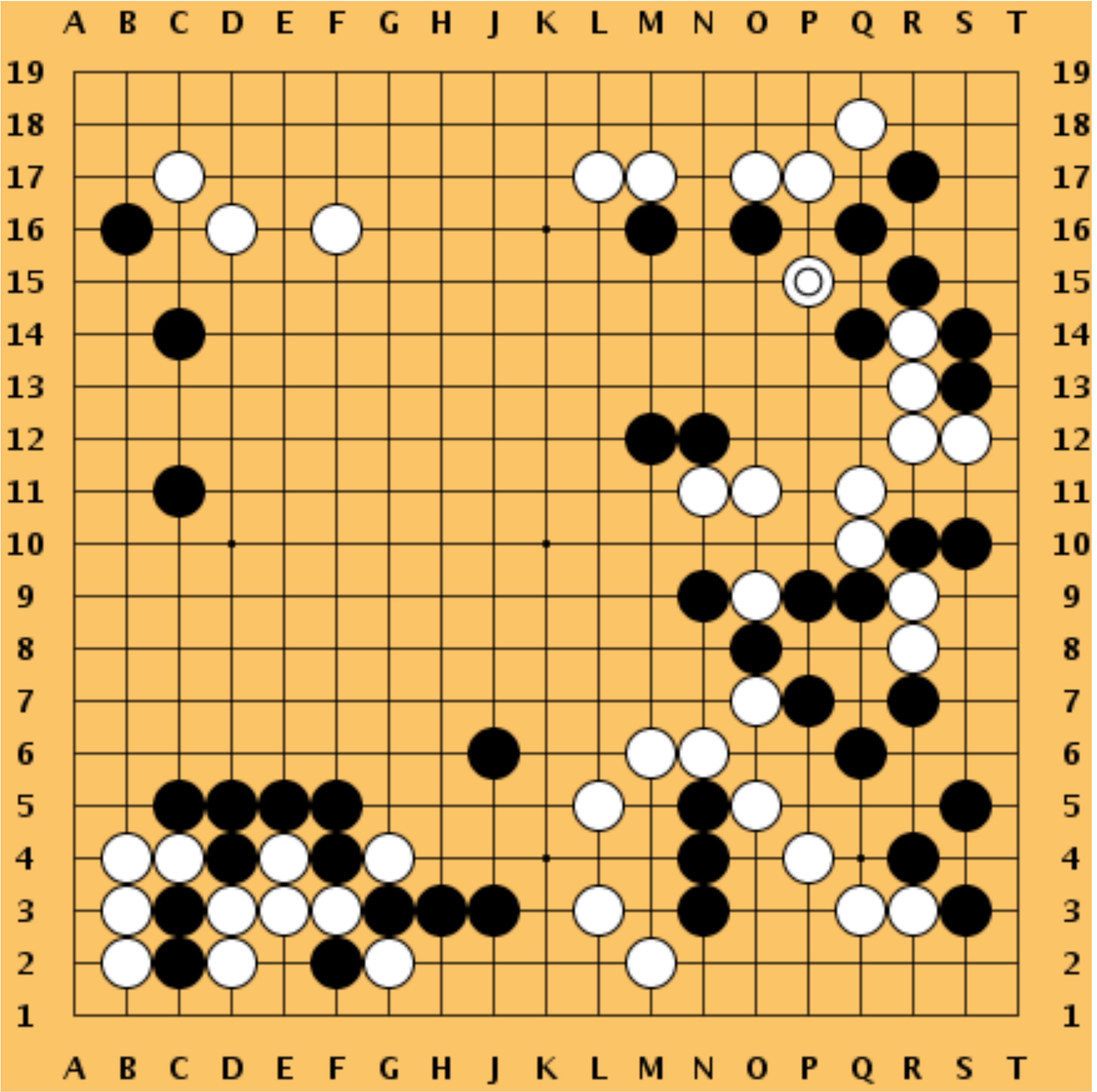
Atari Games

Objective: Complete the game with the highest score

State: Raw pixel inputs of the game state
Action: Game controls e.g. Left, Right, Up, Down
Reward: Score increase/decrease at each time step

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Go Game (AlphaGo)



Objective: Win the game!

State: Position of all pieces
Action: Where to put the next piece down
Reward: 1 if win at the end of the game, 0 otherwise

Markov Decision Processes

— Mathematical **formulation** of the RL problem

Defined by:

$\mathcal{S}$: set of possible states
$\mathcal{A}$: set of possible actions
$\mathcal{R}$: distribution of reward given (state, action) pair
$\mathbb{P}$: transition probability i.e. distribution over next state given (state, action) pair
γ : discount factor

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Markov Decision Processes

At times step t=0, environment samples initial state

For time t=0 until done:

- Agent selects action (deterministically or stochastically)
- Environment samples the reward
- Environment samples the next state
- Agent receives reward and next state

Markov Decision Processes

- Mathematical **formulation** of the RL problem

  Defined by:

  $\mathcal{S}$: set of possible states
  $\mathcal{A}$: set of possible actions
  $\mathcal{R}$: distribution of reward given (state, action) pair
  $\mathbb{P}$: transition probability i.e. distribution over next state given (state, action) pair
  γ : discount factor

- Life is **trajectory**: $\ldots S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1}, R_{t+2}, S_{t+2}, \ldots$

Markov Decision Processes

— Mathematical **formulation** of the RL problem

Defined by:

$\mathcal{S}$: set of possible states
$\mathcal{A}$: set of possible actions
$\mathcal{R}$: distribution of reward given (state, action) pair
$\mathbb{P}$: transition probability i.e. distribution over next state given (state, action) pair
γ : discount factor

— Life is **trajectory**: $\ldots S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1}, R_{t+2}, S_{t+2}, \ldots$

— **Markov property**: Current state completely characterizes the state of the world

$$p(r, s' | s, a) = Prob\Big[R_{t+1} = r, S_{t+1} = s' \;\Big|\; S_t = s, A_t = a\Big]$$

Components of the RL Agent

Policy

— How does the agent behave?

Value Function

— How good is each state and/or state-action pair?

Model

— Agent's representation of the environment

* slide from Dhruv Batra

Policy

— The policy is how the agent acts

— Formally, map from states to actions:

Deterministic policy: $a = \pi(s)$

Stochastic policy: $\pi(a|s) = \mathbb{P}[A_t = a|S_t = s]$

* slide from Dhruv Batra

Policy

- The policy is how the agent acts
- Formally, map from states to actions:

Deterministic policy: $a = \pi(s)$

Stochastic policy: $\pi(a|s) = \mathbb{P}[A_t = a | S_t = s]$

e.g.

State	Action
A $\longrightarrow$	2
B $\longrightarrow$	1

Simple example:

A = You are on the street car approaching
B = You are on the street no car approaching

Action 1 = Cross the street
Action 2 = Stop

* slide from Dhruv Batra

Policy

- The policy is how the agent acts
- Formally, map from states to actions:

Deterministic policy: $a = \pi(s)$

Stochastic policy: $\pi(a|s) = \mathbb{P}[A_t = a | S_t = s]$

e.g.

State	Action
A →	2
B →	1

Simple example:

A = You are on the street car approaching
B = You are on the street no car approaching

Action 1 = Cross the street
Action 2 = Stop

State	Action 1	Action 2
A	0.1	0.9
B	0.8	0.2

The **Optimal** Policy

What is a good policy?

* slide from Dhruv Batra

The **Optimal** Policy

What is a good policy?

Maximizes current reward? Sum of all future rewards?

* slide from Dhruv Batra

The **Optimal** Policy

What is a good policy?

Maximizes current reward? Sum of all future rewards?

Simple example:

A = You are on the street car approaching
B = You are on the street no car approaching

Action 1 = Cross the street
Action 2 = Stop

* slide from Dhruv Batra

The **Optimal** Policy

What is a good policy?

Maximizes current reward? Sum of all future rewards?

Discounted future rewards!

* slide from Dhruv Batra

The **Optimal** Policy

What is a good policy?

Maximizes current reward? Sum of all future rewards?

Discounted future rewards!

Formally: $\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | \pi\right]$

with $s_0 \sim p(s_0), a_t \sim \pi(\cdot|s_t), s_{t+1} \sim p(\cdot|s_t, a_t)$

The **Optimal** Policy

What is a good policy?

Maximizes current reward? Sum of all future rewards?

Discounted future rewards!

Formally: $\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | \pi\right]$

1. Why do we need expectation?

with $s_0 \sim p(s_0), a_t \sim \pi(\cdot|s_t), s_{t+1} \sim p(\cdot|s_t, a_t)$

* slide from Dhruv Batra

The **Optimal** Policy

What is a good policy?

Maximizes current reward? Sum of all future rewards?

Discounted future rewards!

Formally: $\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | \pi\right]$

1. Why do we need expectation?

2. Why do we need gamma (discount factor)?

with $s_0 \sim p(s_0), a_t \sim \pi(\cdot|s_t), s_{t+1} \sim p(\cdot|s_t, a_t)$

Components of the RL Agent

✓ **Policy**

— How does the agent behave?

Value Function

— How good is each state and/or action pair?

Model

— Agent’s representation of the environment

* slide from Dhruv Batra

Value Function

A value function is a prediction of future reward

“State Value Function" or simply “**Value Function**”

— How good is a state?

— Am I screwed? Am I winning this game?

“Action Value Function” or **Q-function**

— How good is a state action-pair?

— Should I do this now?

* slide from Dhruv Batra

Value Function and **Q-value** Function

Following a policy produces sample trajectories (or paths) $s_0, a_0, r_0, s_1, a_1, r_1, \ldots$

— The **value function** (how good is the state) at state s, is the expected cumulative reward from state s (and following the policy thereafter):

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | s_0 = s, \pi\right]$$

Value Function and **Q-value** Function

Following a policy produces sample trajectories (or paths) $s_0, a_0, r_0, s_1, a_1, r_1, \ldots$

— The **value function** (how good is the state) at state s, is the expected cumulative reward from state s (and following the policy thereafter):

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | s_0 = s, \pi\right]$$

— The **Q-value function** (how good is a state-action pair) at state s and action a, is the expected cumulative reward from taking action a in state s (and following the policy thereafter):

$$Q^{\pi}(s,a) = \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | s_0 = s, a_0 = a, \pi\right]$$

Components of the RL Agent

✓ **Policy**

— How does the agent behave?

✓ **Value Function**

— How good is each state and/or action pair?

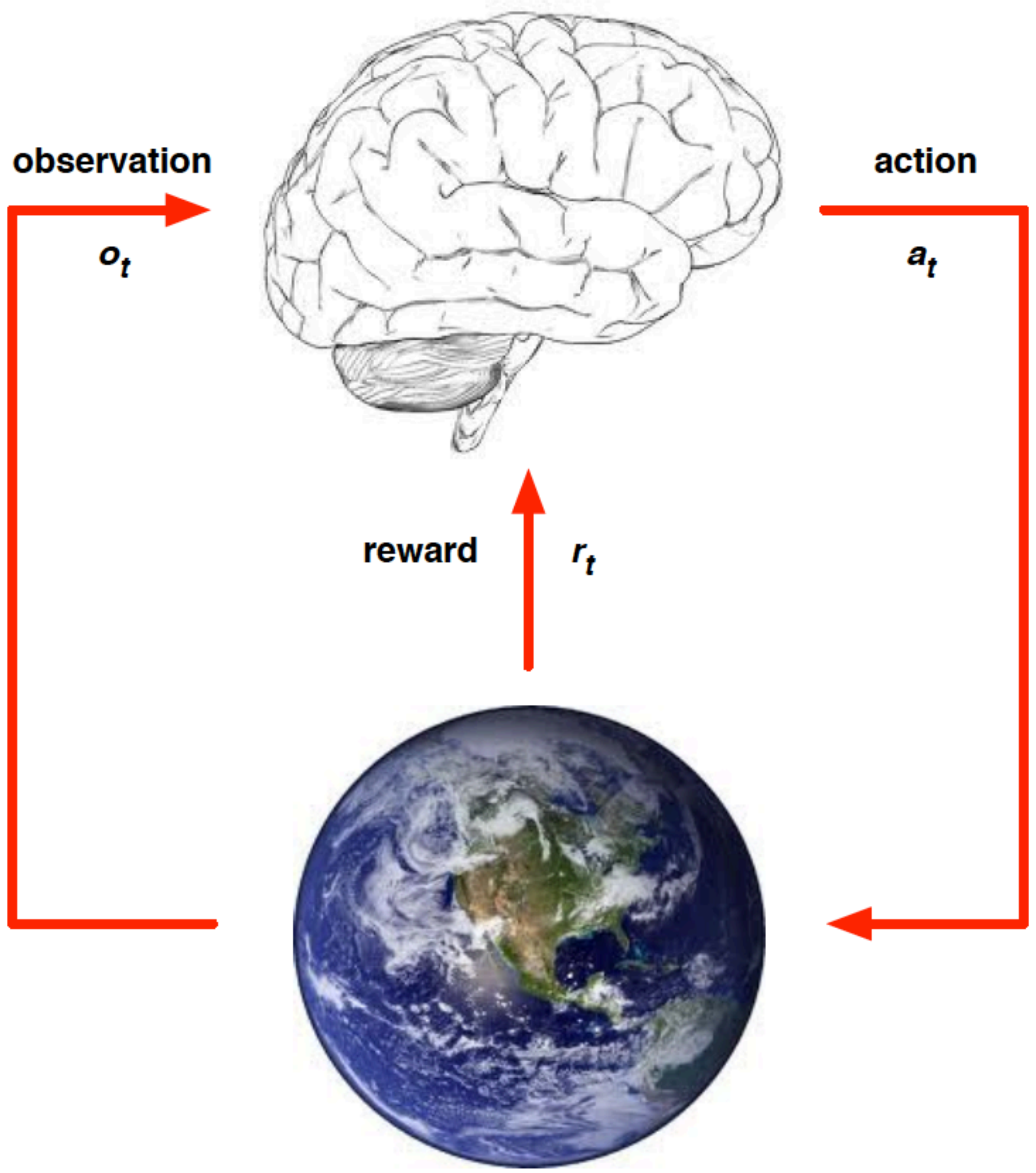
Model

— Agent's representation of the environment

* slide from Dhruv Batra

Model

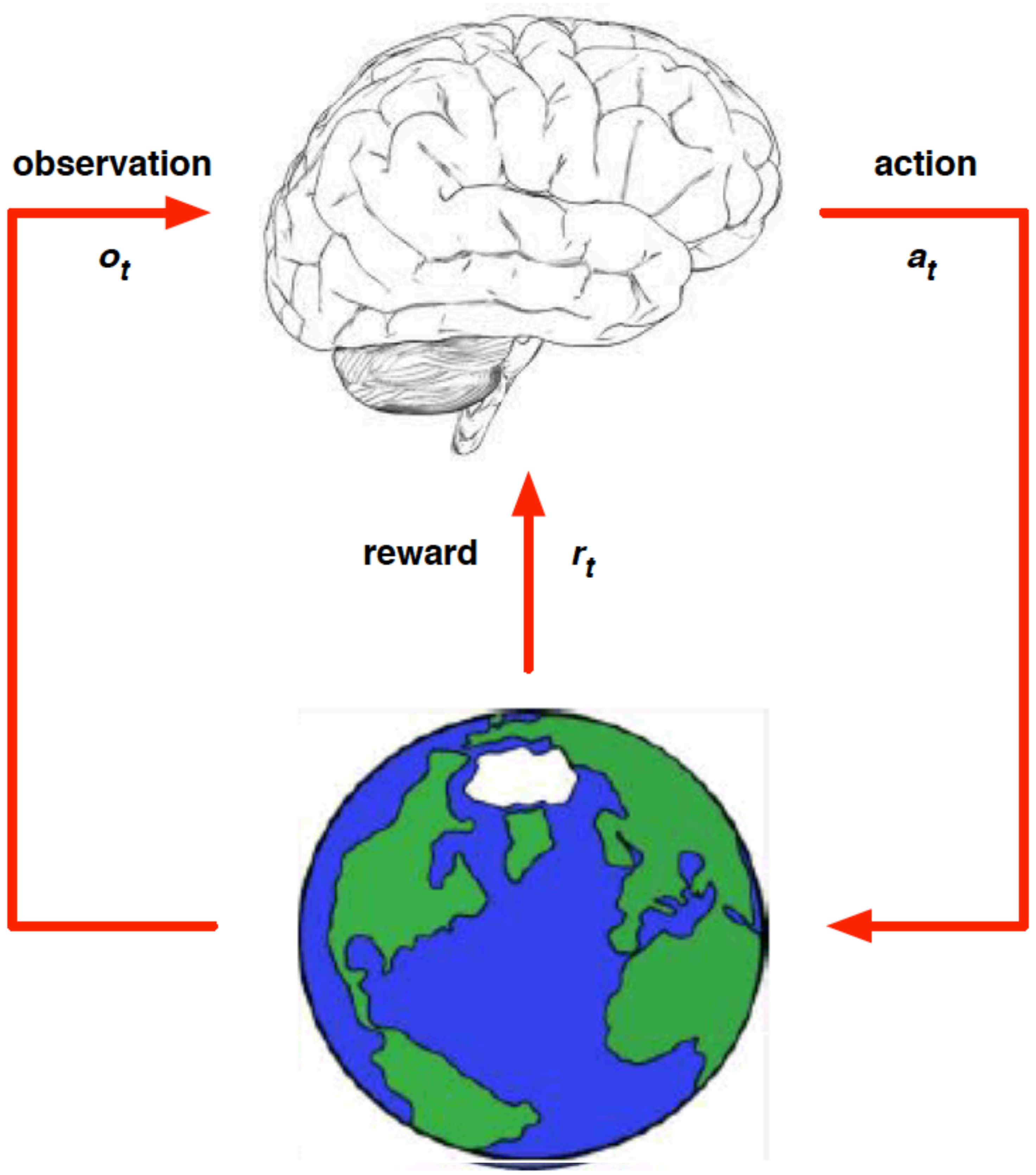
Model predicts what the world will do next



* slide from David Silver

Model

Model predicts what the world will do next



* slide from David Silver

Components of the RL Agent

✓ **Policy**

— How does the agent behave?

✓ **Value Function**

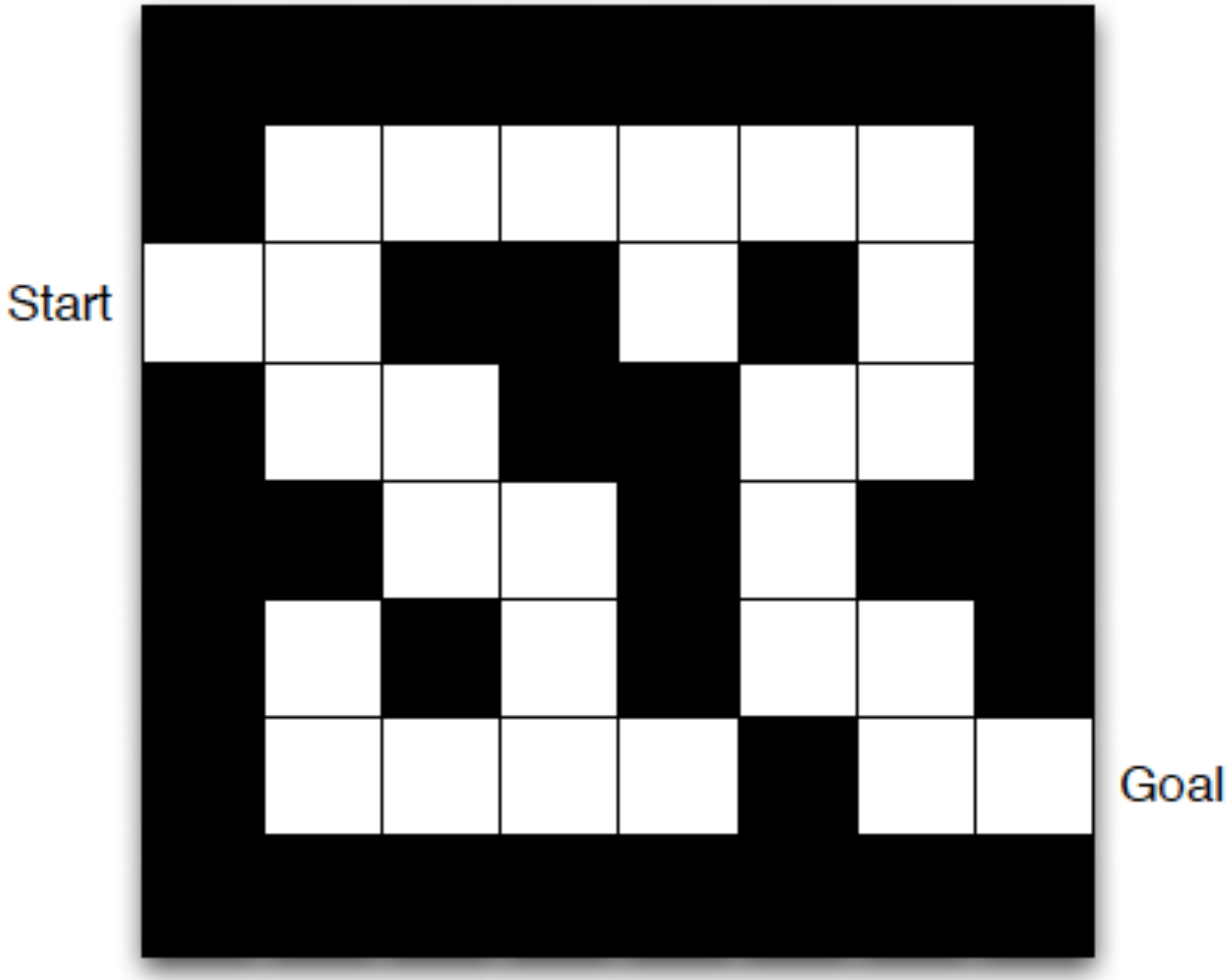
— How good is each state and/or action pair?

✓ **Model**

— Agent's representation of the environment

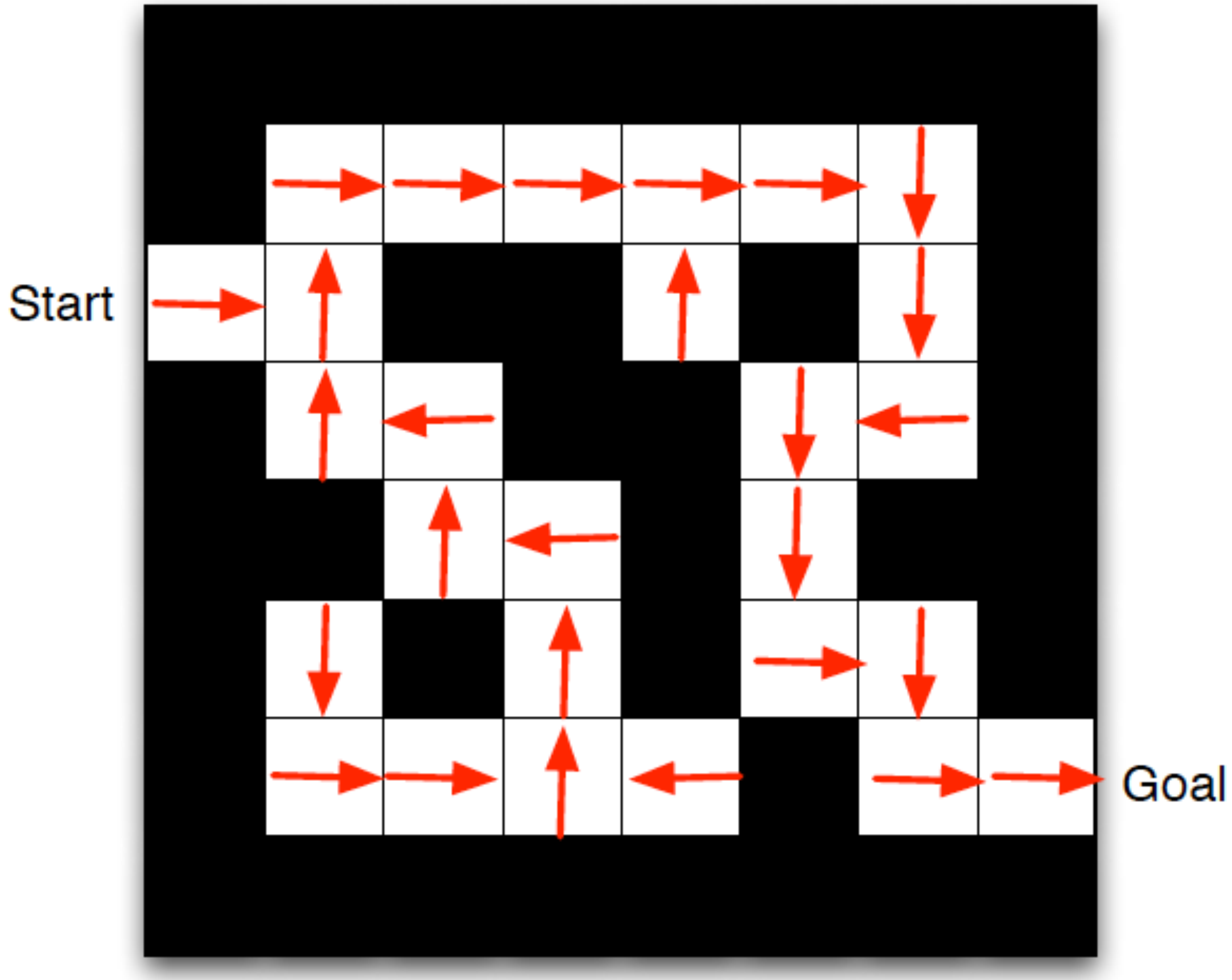
* slide from Dhruv Batra

Maze Example



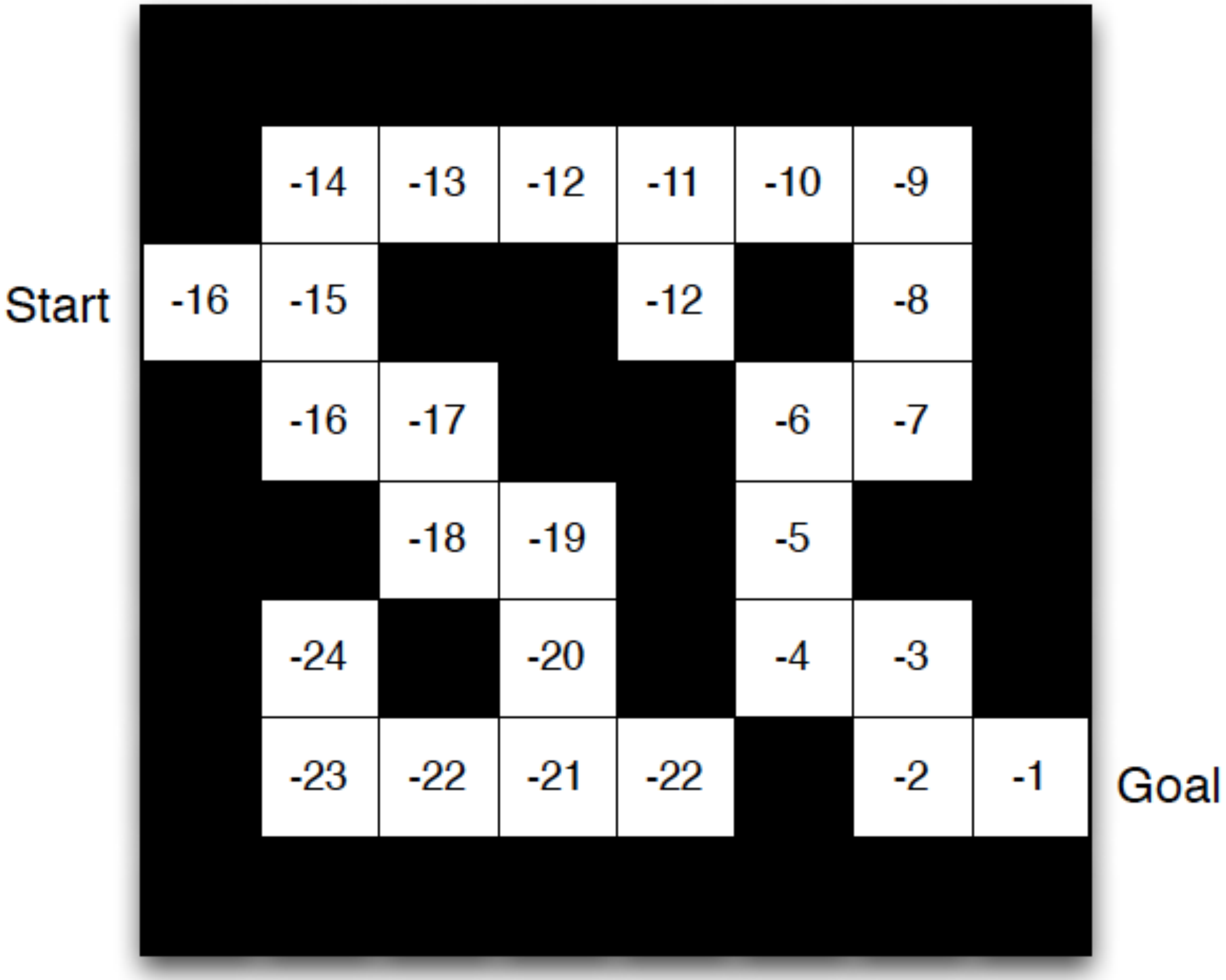
Reward: -1 per time-step
Actions: N, E, S, W
States: Agent's location

Maze Example: Policy



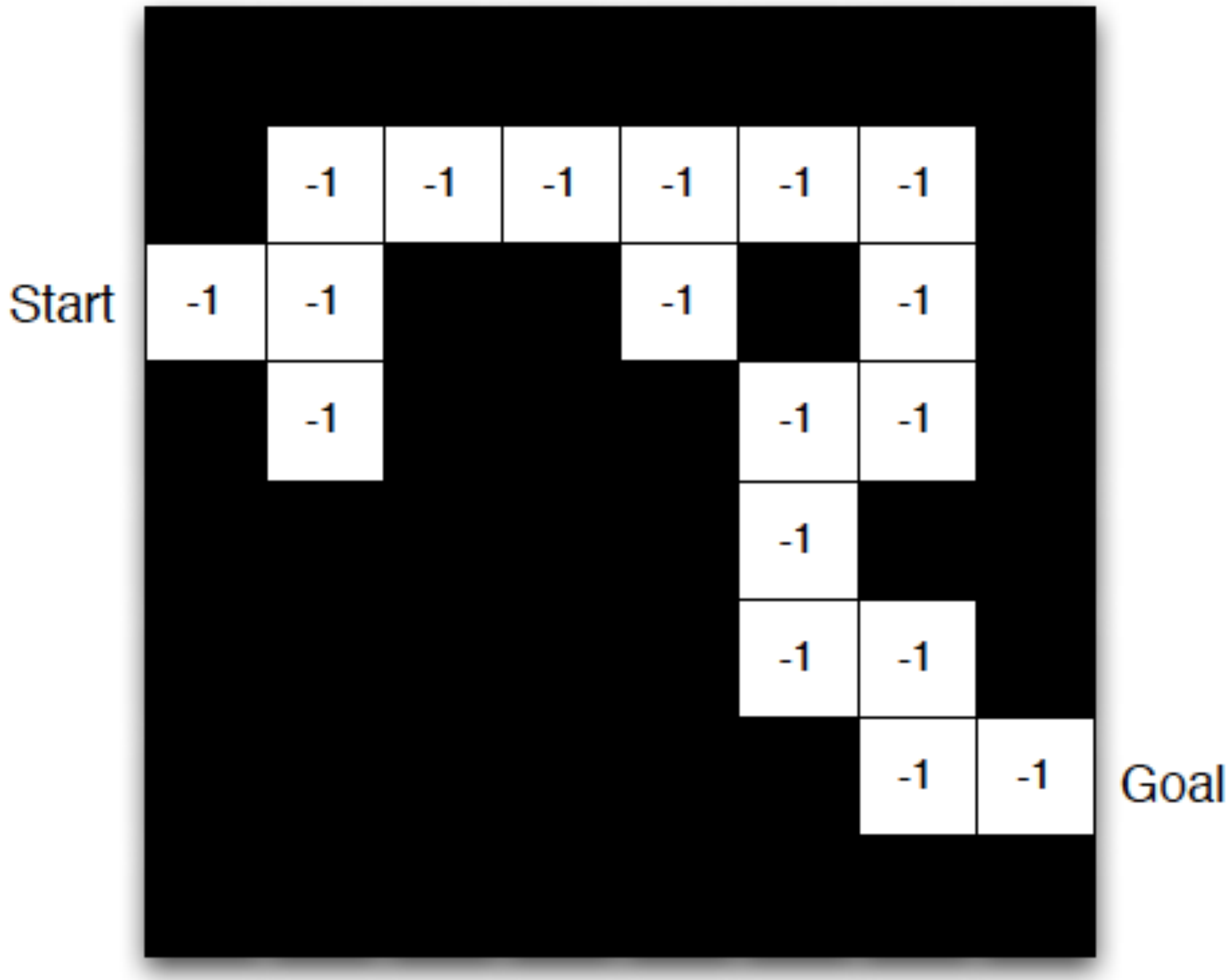
Arrows represent a policy $\pi(s)$ for each state s

Maze Example: Value



Numbers represent value $v_\pi(s)$ of each state s

Maze Example: Model



Grid layout represents transition model

Numbers represent the immediate reward for each state (same for all states)

* slide from David Silver

Components of the RL Agent

Policy

— How does the agent behave?

Value Function

— How good is each state and/or action pair?

Model

— Agent's representation of the environment

* slide from Dhruv Batra

Approaches to RL: Taxonomy

Model-free RL

Value-based RL

- Estimate the optimal action-value function $Q^*(s,a)$
- No policy (implicit)

Policy-based RL

- Search directly for the optima policy π^*
- No value function

Model-based RL

- Build a model of the world
- Plan (e.g., by look-ahead) using model

* slide from Dhruv Batra

Approaches to RL: Taxonomy

Model-free RL

Value-based RL

- — Estimate the optimal action-value function $Q^*(s, a)$
- — No policy (implicit)

Policy-based RL

- — Search directly for the optima policy π^*
- — No value function

Actor-critic RL

- — Value function
- — Policy function

Model-based RL

- — Build a model of the world
- — Plan (e.g., by look-ahead) using model

Deep RL

Value-based RL

— Use neural nets to represent Q function $Q(s, a; \theta)$

$$Q(s, a; \theta^*) \approx Q^*(s, a)$$

Policy-based RL

— Use neural nets to represent the policy π_θ

$$\pi_{\theta^*} \approx \pi^*$$

Model-based RL

— Use neural nets to represent and learn the model

* slide from Dhruv Batra

Approaches to RL

Value-based RL

- Estimate the optimal action-value function $Q^*(s, a)$
- No policy (implicit)

* slide from Dhruv Batra

Optimal Value Function

Optimal Q-function is the maximum achievable value

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a) = Q^{\pi^*}(s,a)$$

Optimal Value Function

Optimal Q-function is the maximum achievable value

$$Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a) = Q^{\pi^*}(s, a)$$

Once we have it, we can act optimally

$$\pi^*(s) = \operatorname*{argmax}_{a} Q^*(s, a)$$

* slide from David Silver

Optimal Value Function

Optimal Q-function is the maximum achievable value

$$Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a) = Q^{\pi^*}(s, a)$$

Once we have it, we can act optimally

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\ Q^*(s, a)$$

Optimal value maximizes over all future decisions

$$\begin{aligned} Q^*(s, a) &= r_{t+1} + \gamma \max_{a_{t+1}} r_{t+2} + \gamma^2 \max_{a_{t+2}} r_{t+3} + \ldots \\ &= r_{t+1} + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) \end{aligned}$$

* slide from David Silver

Optimal Value Function

Optimal Q-function is the maximum achievable value

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a) = Q^{\pi^*}(s,a)$$

Once we have it, we can act optimally

$$\pi^*(s) = \operatorname*{argmax}_{a} Q^*(s,a)$$

Optimal value maximizes over all future decisions

$$\begin{aligned} Q^*(s,a) &= r_{t+1} + \gamma \max_{a_{t+1}} r_{t+2} + \gamma^2 \max_{a_{t+2}} r_{t+3} + \ldots \\ &= r_{t+1} + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) \end{aligned}$$

Formally, Q* satisfied Bellman Equations

$$Q^*(s,a) = \mathbb{E}_{s'}\left[r + \gamma \max_{a'} Q^*(s',a') \mid s,a \right]$$

* slide from David Silver

Solving for the Optimal Policy

Value iteration algorithm: Use Bellman equation as an iterative update

$$Q_{i+1}(s,a) = \mathbb{E}\left[r + \gamma \max_{a'} Q_i(s',a')|s,a\right]$$

Q_i will converge to Q^* as i -> infinity

Solving for the Optimal Policy

Value iteration algorithm: Use Bellman equation as an iterative update

$$Q_{i+1}(s,a) = \mathbb{E}\left[r + \gamma \max_{a'} Q_i(s',a')|s,a\right]$$

Q_i will converge to Q^* as i -> infinity

What's the problem with this?

Solving for the Optimal Policy

Value iteration algorithm: Use Bellman equation as an iterative update

$$Q_{i+1}(s,a) = \mathbb{E}\left[r + \gamma \max_{a'} Q_i(s',a')|s,a\right]$$

Q_i will converge to Q^* as i -> infinity

What's the problem with this?

Not scalable. Must compute Q(s,a) for every state-action pair. If state is e.g. game pixels, computationally infeasible to compute for entire state space!

Solving for the Optimal Policy

Value iteration algorithm: Use Bellman equation as an iterative update

$$Q_{i+1}(s,a) = \mathbb{E}\left[r + \gamma \max_{a'} Q_i(s',a')|s,a\right]$$

Q_i will converge to Q^* as i -> infinity

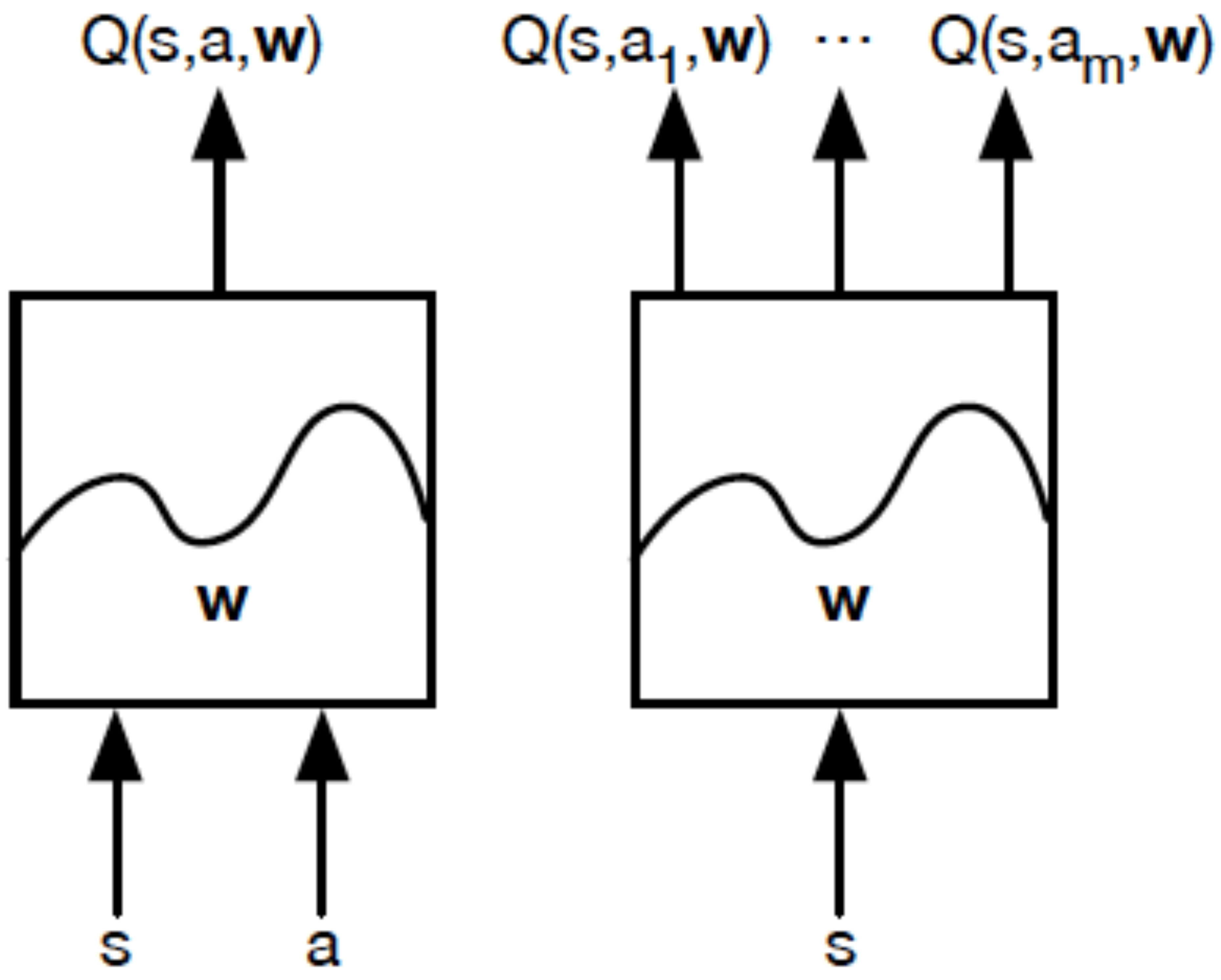
What's the problem with this?

Not scalable. Must compute Q(s,a) for every state-action pair. If state is e.g. game pixels, computationally infeasible to compute for entire state space!

Solution: use a function approximator to estimate Q(s,a). E.g. a neural network!

Q-Networks

$$Q(s, a, \mathbf{w}) \approx Q^*(s, a)$$



* slide from David Silver

Case Study: Playing **Atari** Games

[Mnih *et al.*, 2013; Nature 2015]

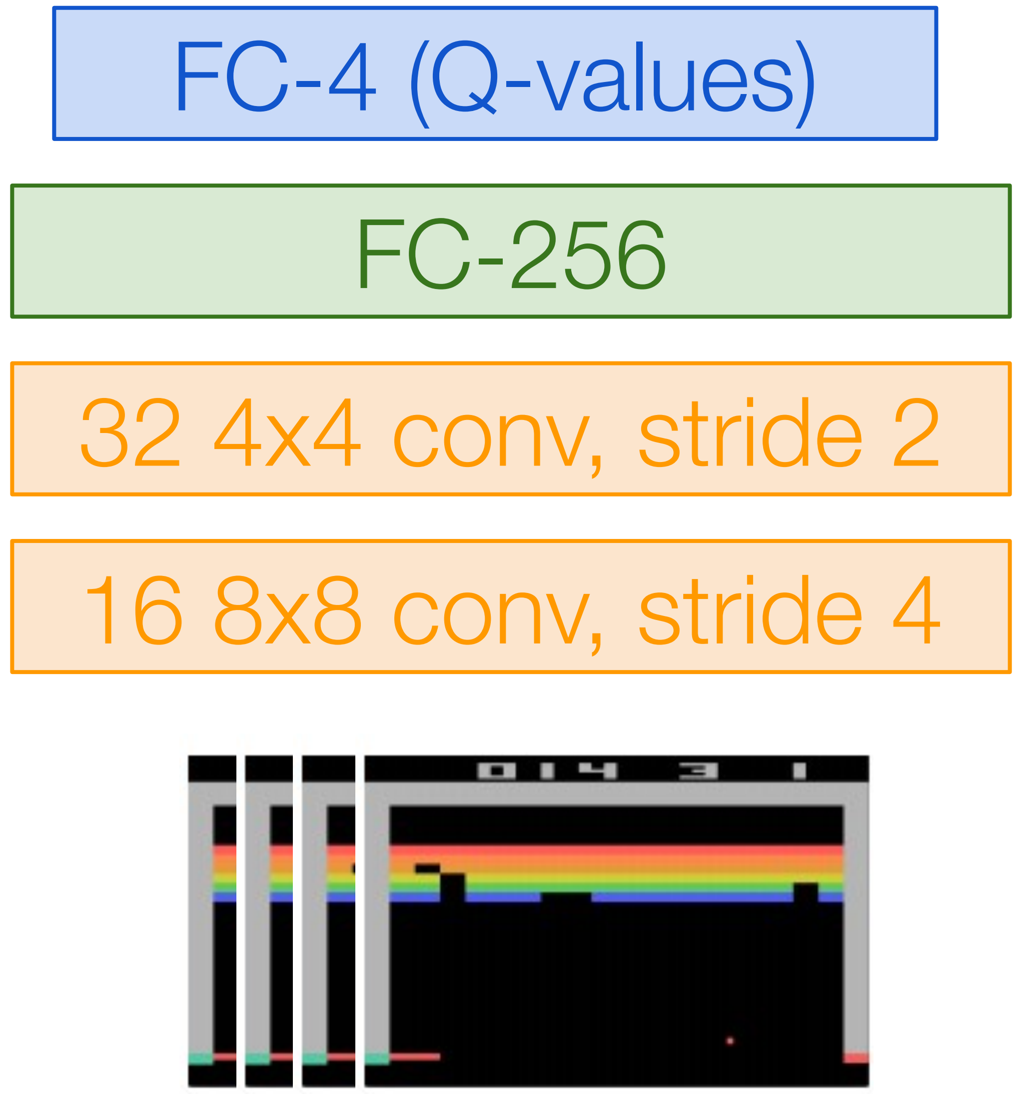
Objective: Complete the game with the highest score

State: Raw pixel inputs of the game state
Action: Game controls e.g. Left, Right, Up, Down
Reward: Score increase/decrease at each time step

Q-Network Architecture

[Mnih *et al.*, 2013; Nature 2015]

$Q(s, a; \theta)$: neural network with weights θ

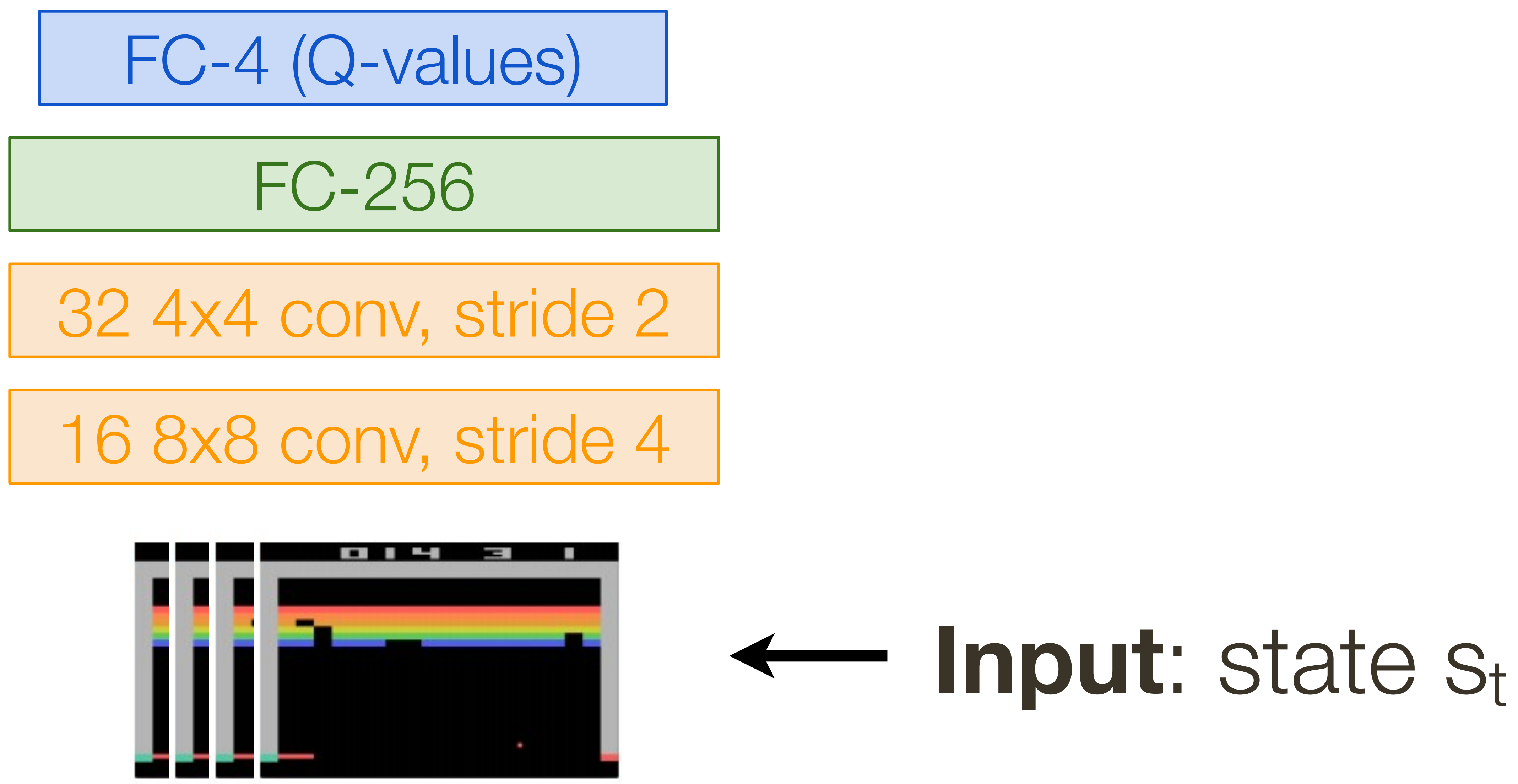


Current state s_t: 84x84x4 stack of last 4 frames
(after RGB->grayscale conversion, downsampling, and cropping)

Q-Network Architecture

[Mnih *et al.*, 2013; Nature 2015]

$Q(s, a; \theta)$: neural network with weights θ



Current state s_t: 84x84x4 stack of last 4 frames
(after RGB->grayscale conversion, downsampling, and cropping)

Q-Network Architecture

[Mnih *et al.*, 2013; Nature 2015]

$Q(s, a; \theta)$: neural network with weights θ

FC-4 (Q-values)

FC-256

32 4x4 conv, stride 2

16 8x8 conv, stride 4

familiar conv and fc layers

Current state s_t: 84x84x4 stack of last 4 frames (after RGB->grayscale conversion, downsampling, and cropping)

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Q-Network Architecture

[Mnih *et al.*, 2013; Nature 2015]

$Q(s, a; \theta)$: neural network with weights θ

FC-4 (Q-values)

FC-256

32 4x4 conv, stride 2

16 8x8 conv, stride 4

Last FC layer has 4-d output (if 4 actions), corresponding to $Q(s_t, a_1)$, $Q(s_t, a_2)$, $Q(s_t, a_3)$, $Q(s_t, a_4)$

Current state s_t: 84x84x4 stack of last 4 frames (after RGB->grayscale conversion, downsampling, and cropping)

Q-Network Architecture

[Mnih *et al.*, 2013; Nature 2015]

$Q(s, a; \theta)$: neural network with weights θ

FC-4 (Q-values)

FC-256

32 4x4 conv, stride 2

16 8x8 conv, stride 4

Last FC layer has 4-d output (if 4 actions), corresponding to $Q(s_t, a_1)$, $Q(s_t, a_2)$, $Q(s_t, a_3)$, $Q(s_t, a_4)$

Number of actions between 4-18 depending on Atari game

Current state s_t: 84x84x4 stack of last 4 frames (after RGB->grayscale conversion, downsampling, and cropping)

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Q-Network Architecture

[Mnih *et al.*, 2013; Nature 2015]

$Q(s, a; \theta)$: neural network with weights θ

FC-4 (Q-values)

FC-256

32 4x4 conv, stride 2

16 8x8 conv, stride 4

Last FC layer has 4-d output (if 4 actions), corresponding to $Q(s_t, a_1)$, $Q(s_t, a_2)$, $Q(s_t, a_3)$, $Q(s_t, a_4)$

A single feedforward pass to compute Q-values for all actions from the current state => efficient!

Number of actions between 4-18 depending on Atari game

Current state s_t: 84x84x4 stack of last 4 frames
(after RGB->grayscale conversion, downsampling, and cropping)

Q-Network Learning

Remember: want to find a Q-function that satisfies the Bellman Equation:

$$Q^*(s,a) = \mathbb{E}[r + \gamma \max_{a'} Q^*(s',a') \mid s,a]$$

Q-Network Learning

Remember: want to find a Q-function that satisfies the Bellman Equation:

$$Q^*(s,a) = \mathbb{E}[r + \gamma \max_{a'} Q^*(s',a') \mid s,a]$$

Forward Pass:

Loss function: $L_i(\theta_i) = \mathbb{E}\left[(y_i - Q(s,a;\theta_i)^2\right]$

where $y_i = \mathbb{E}[r + \gamma \max_{a'} Q^*(s',a') \mid s,a]$

Q-Network Learning

Remember: want to find a Q-function that satisfies the Bellman Equation:

$$Q^*(s,a) = \mathbb{E}[r + \gamma \max_{a'} Q^*(s',a') \mid s,a]$$

Forward Pass:

Loss function: $L_i(\theta_i) = \mathbb{E}\left[(y_i - Q(s,a;\theta_i)^2\right]$

where $y_i = \mathbb{E}[r + \gamma \max_{a'} Q^*(s',a') \mid s,a]$

Backward Pass:

Gradient update (with respect to Q-function parameters θ):

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}\left[r + \gamma \max_{a'} Q(s',a';\theta_{i-1}) - Q(s,a;\theta_i))\nabla_{\theta_i} Q(s,a;\theta_i)\right]$$

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Q-Network Learning

Remember: want to find a Q-function that satisfies the Bellman Equation:

$$Q^*(s,a) = \mathbb{E}[r + \gamma \max_{a'} Q^*(s',a') \mid s,a]$$

Forward Pass:

Loss function: $L_i(\theta_i) = \mathbb{E}\left[(y_i - Q(s,a;\theta_i))^2\right]$

where $y_i = \mathbb{E}[r + \gamma \max_{a'} Q^*(s',a') \mid s,a]$

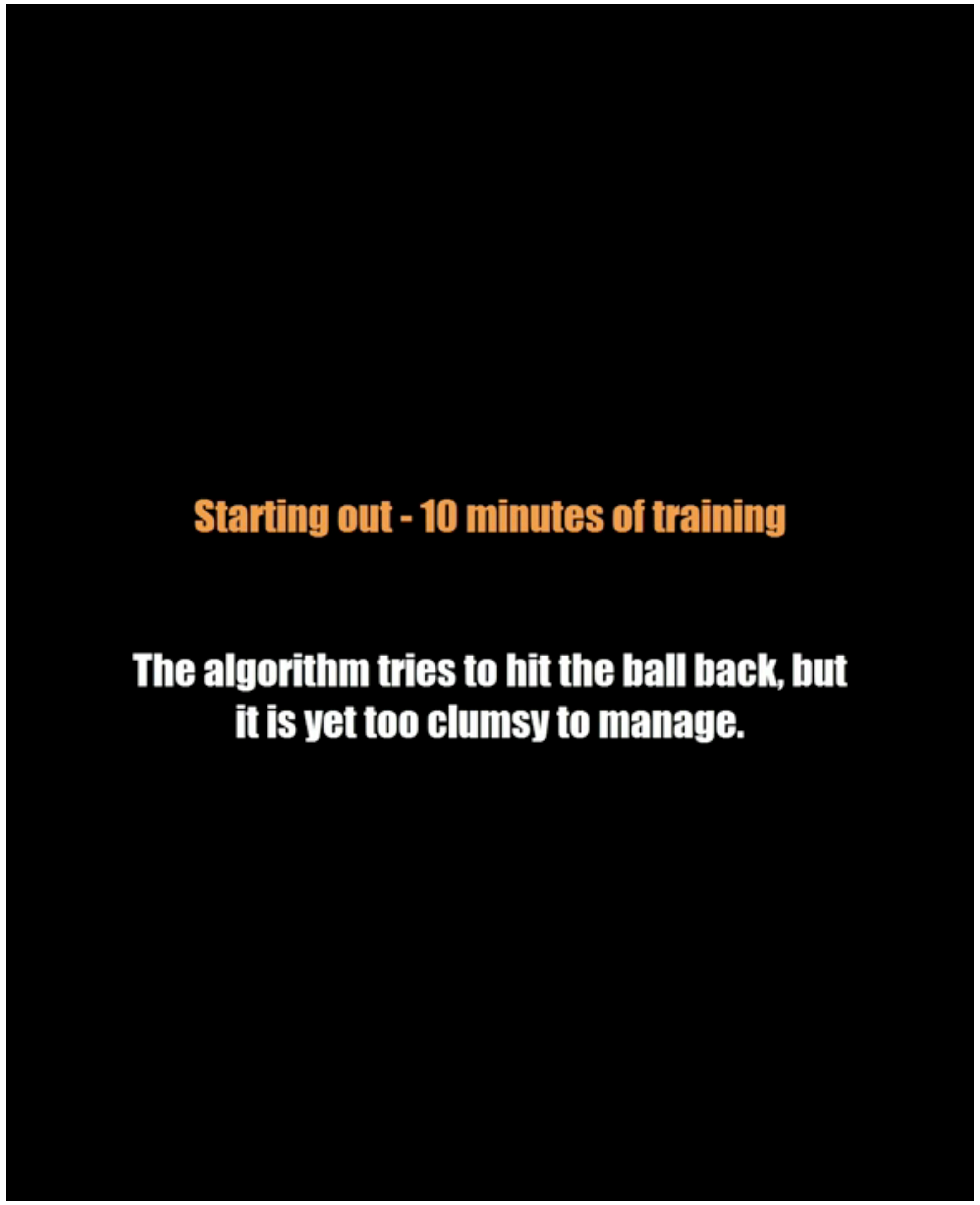
Iteratively try to make the Q-value close to the target value (y_i) it should have, if Q-function corresponds to optimal Q* (and optimal policy π^*)

Backward Pass:

Gradient update (with respect to Q-function parameters θ):

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}\left[r + \gamma \max_{a'} Q(s',a';\theta_{i-1}) - Q(s,a;\theta_i))\nabla_{\theta_i} Q(s,a;\theta_i)\right]$$

Example: Atari Playing



Example: Atari Playing