Computer Vision:
Graph Neural Networks

Siheng Chen 陈思衡

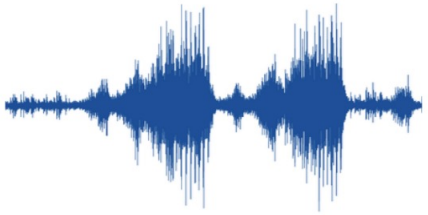
Outline

- Graph neural network basics
- GNN for visual physics
- GCN for multi-person behavior prediction

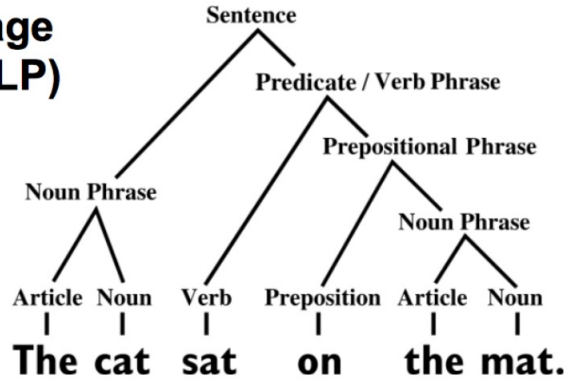
Traditional Neural Networks

IMAGENET

Speech data

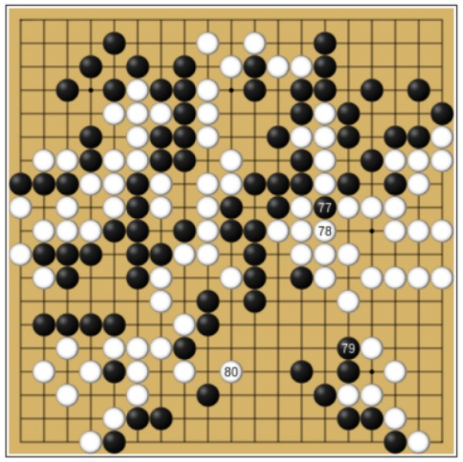


Natural language processing (NLP)



...

Grid games



Deep neural nets that exploit:

- translation equivariance (weight sharing)
- hierarchical compositionality

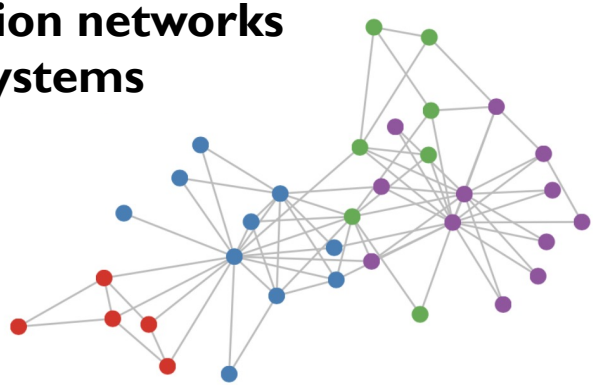
https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

Graph-Structured Data

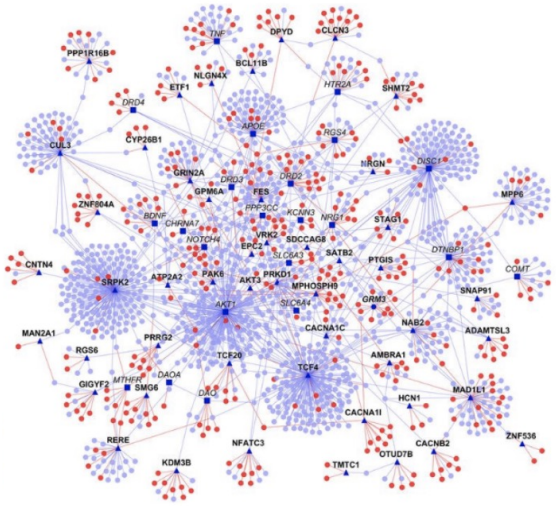
A lot of real-world data does not "live" on grids

Standard CNN and RNN architectures don't work on these data

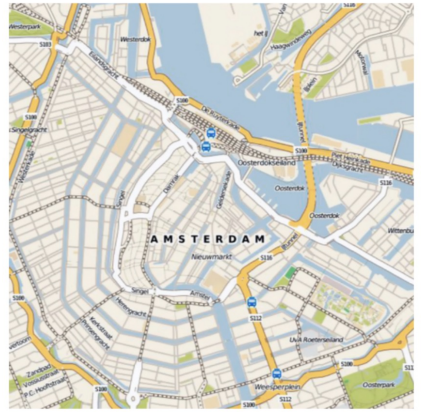
Social networks
Citation networks
Communication networks
Multi-agent systems



Proteins interaction networks

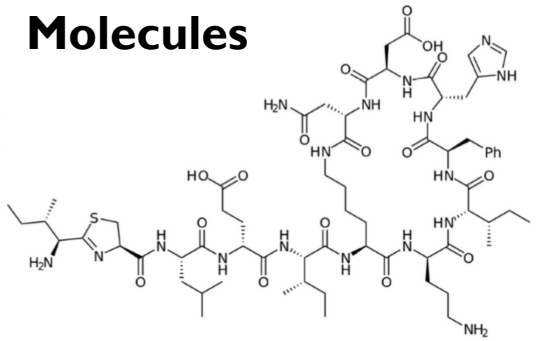


Road maps



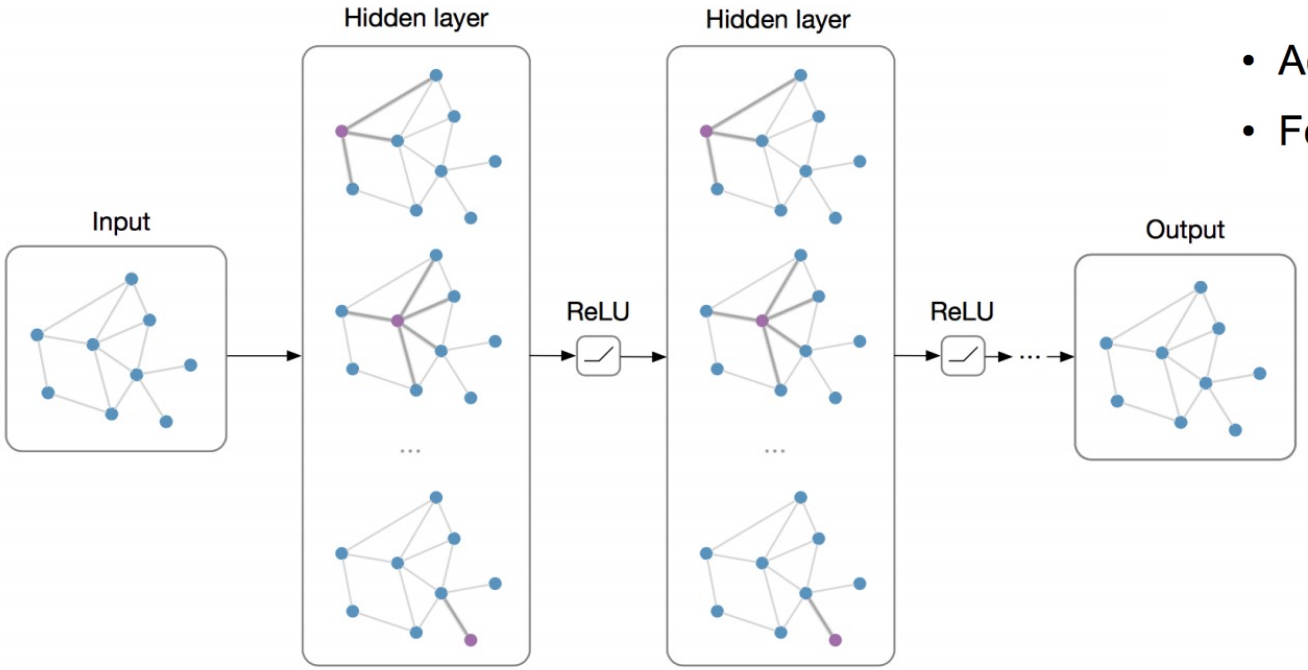
Knowledge graphs

Molecules



https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

Graph Neural Networks (GNNs)



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

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

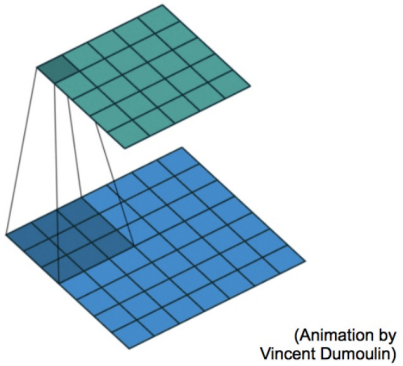
Main Idea: Pass messages between pairs of nodes and agglomerate

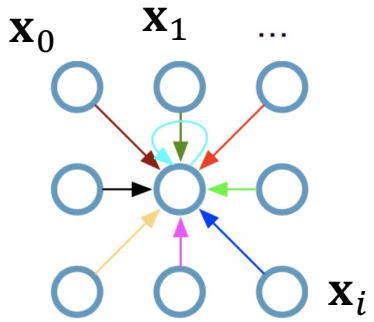
Alternative Interpretation: Pass messages between nodes to refine node (and possibly edge) representations

https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

Recap: Convolutional Neural Networks (CNNs) on Grids

Single CNN layer with 3×3 filter





Update for a single pixel:

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

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

Full update:

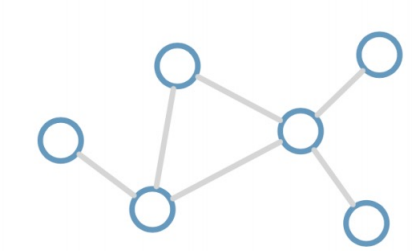
- $\mathbf{x}_4^{(l+1)} = \sigma(\mathbf{W}_0^l\mathbf{x}_0^l + \mathbf{W}_1^l\mathbf{x}_1^l + \cdots + \mathbf{W}_8^l\mathbf{x}_8^l)$

https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

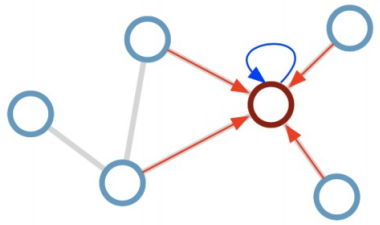
Graph Convolutional Networks (GCNs)

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)

Update rule:

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

Matrix form

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

$\mathcal{N}_i$: neighbor indices

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

$\mathbf{W} \in \mathbb{R}^{d^{(l+1)} \times d^{(l)}}$: Weight matrix

$\tilde{\mathbf{A}} \in \mathbb{R}^{n \times n}$: Normalized adjacency matrix

$\mathbf{X}^{(l)} \in \mathbb{R}^{n \times d^{(l)}}$: Node feature vector in layer l

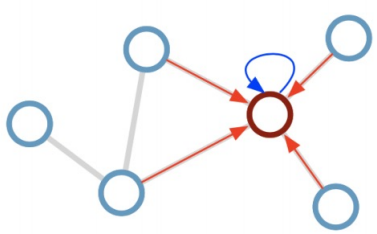
https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

Graph Convolutional Networks (GCNs)

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

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

feature aggregation



Properties for feature aggregation :

- Penalizes the feature difference between neighboring nodes
- low-pass filter

$$\mathbf{X}_{\text{out}} = \tilde{\mathbf{A}}\mathbf{X}_{\text{in}}$$

Interpreted as one gradient descent step for the **Laplacian smoothing problem :**

Make graph smoother

$$\underset{\mathbf{X}\in\mathbb{R}^{n\times d}}{\arg\min}\ \mathcal{L}_1(\mathbf{X}) := \frac{1}{2}\mathrm{tr}\left(\mathbf{X}^\top(\mathbf{I}-\tilde{\mathbf{A}})\mathbf{X}\right) = \frac{1}{2}\sum_{(v_i,v_j)\in\mathcal{E}}\left\|\frac{\mathbf{X}_i}{\sqrt{d_i+1}} - \frac{\mathbf{X}_j}{\sqrt{d_j+1}}\right\|_2^2$$

One gradient descent step :

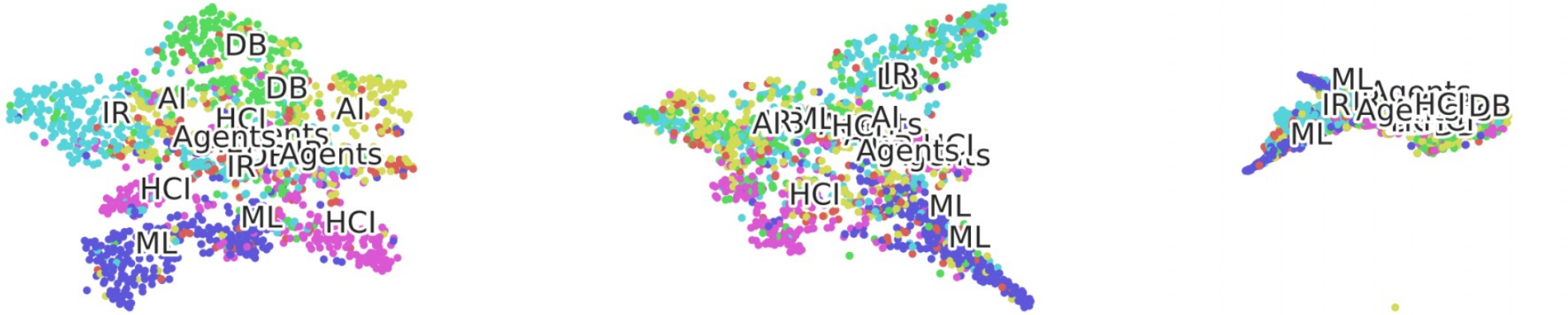
$$\mathbf{X}_{\text{out}} = \mathbf{X}_{\text{in}} - (\mathbf{I}-\tilde{\mathbf{A}})\mathbf{X}_{\text{in}} = \tilde{\mathbf{A}}\mathbf{X}_{\text{in}}$$

Graph Convolutional Networks (GCNs)

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

GCN Make graph smoother

Citation network: Node feature visualization with t-SNE



2 layers

5 layers

7 layers

2 layers: each cluster gets smoother
7 layers: the whole graph gets smoother (over smoothing)

Graph Neural Networks with Attention (GAT)

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

GCN $\quad \mathbf{X}^{(l+1)} = \sigma\left(\widetilde{\mathbf{A}}\mathbf{X}^{(l)}\mathbf{W}\right)$

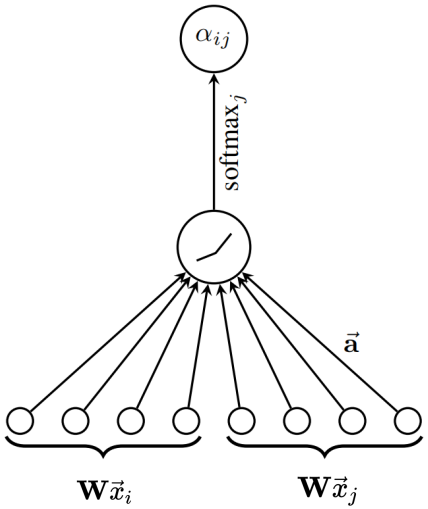
General form of GAT $\quad \mathbf{X}^{(l+1)} = \sigma\left((\boldsymbol{\alpha} \circ \widetilde{\mathbf{A}})\mathbf{X}^{(l)}\mathbf{W}\right)$

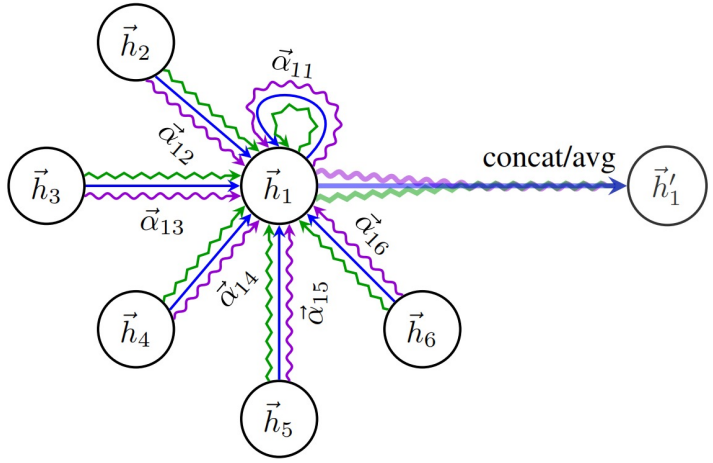
$\boldsymbol{\alpha}$: trainable attention matrix

$\widetilde{\mathbf{A}}$: prior adjacency mask

Main idea: trainable adjacency coefficient (attention on neighbors)

- Attention coefficient α_{ij} from node i to j
- Averaged multi-head-attention aggregation





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

Properties for GAT :

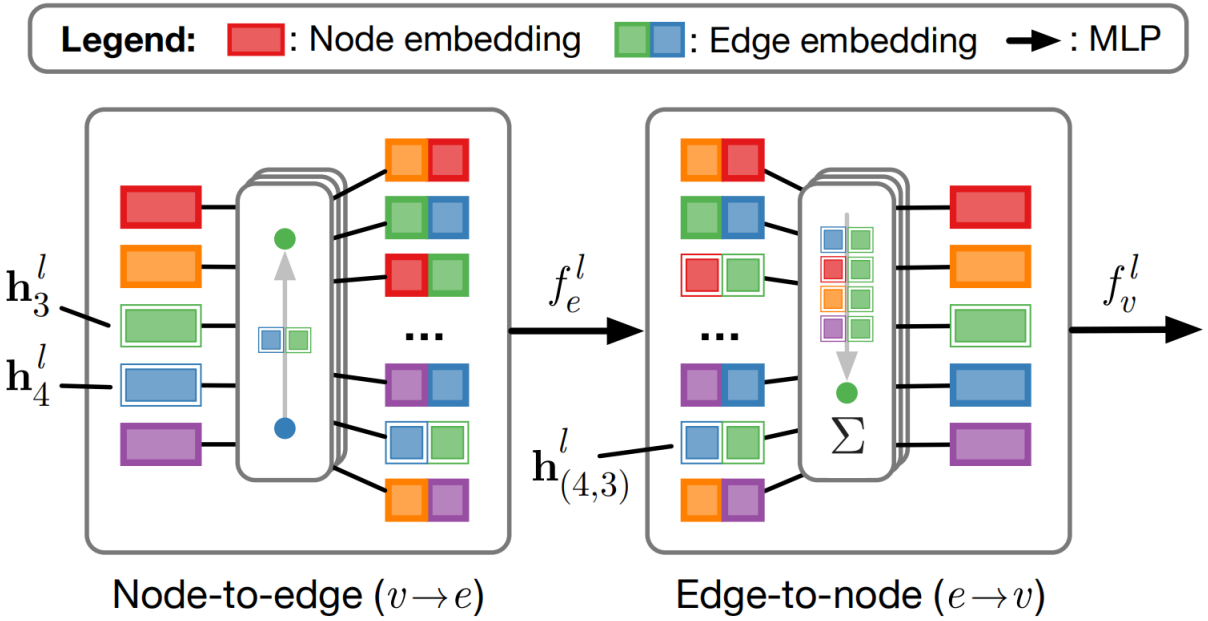
- Higher model capacity than GCN
- Efficient Computationally

  (parallelized across edges and nodes)
- Can be more difficult to optimize

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

GNNs with Edge Embeddings

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



Pros:

- Support edge features
- More expressive
- A general form of message passing

Cons:

- Need to store intermediate edge-based activations

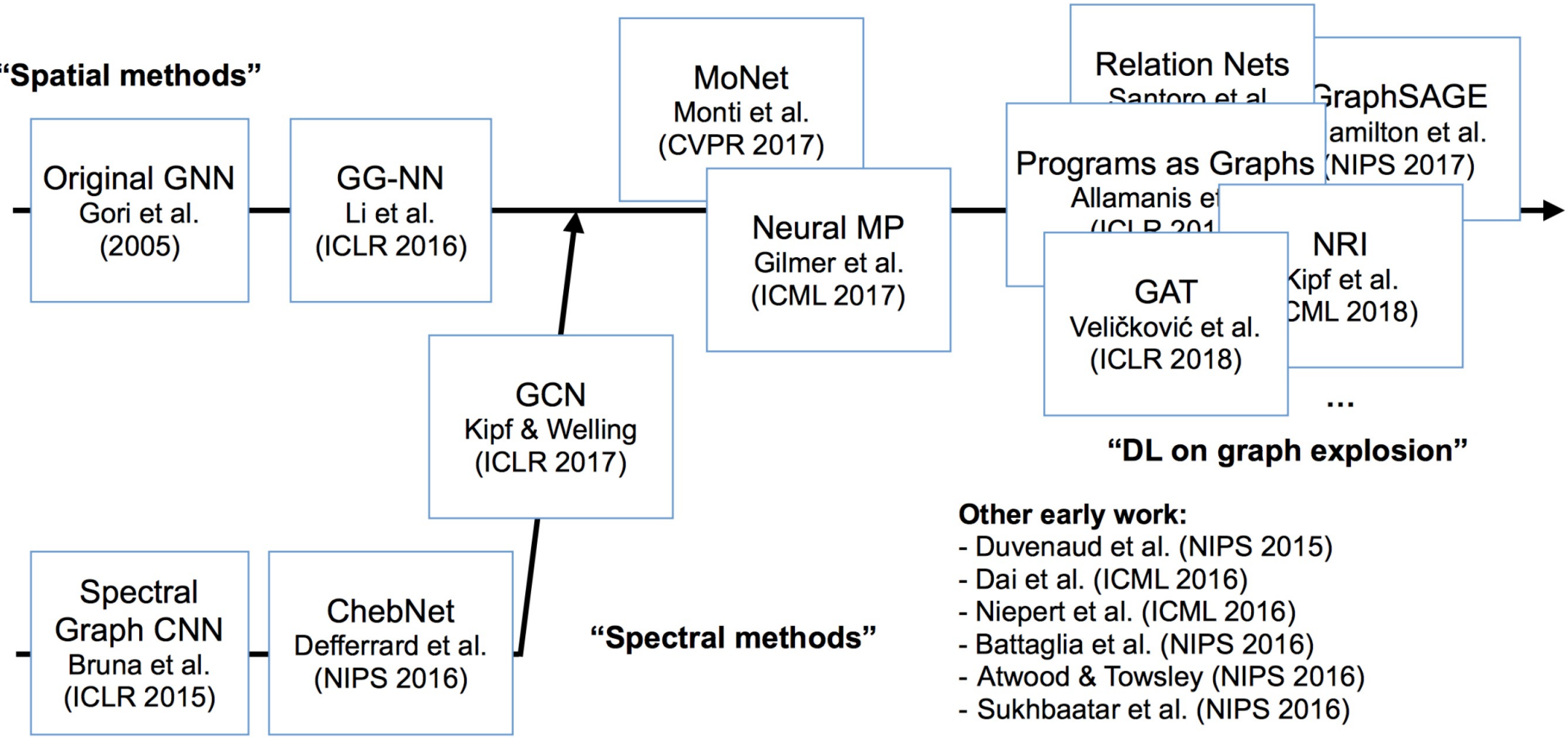
➢ In practice limited to small graphs

Formally:

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

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

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)

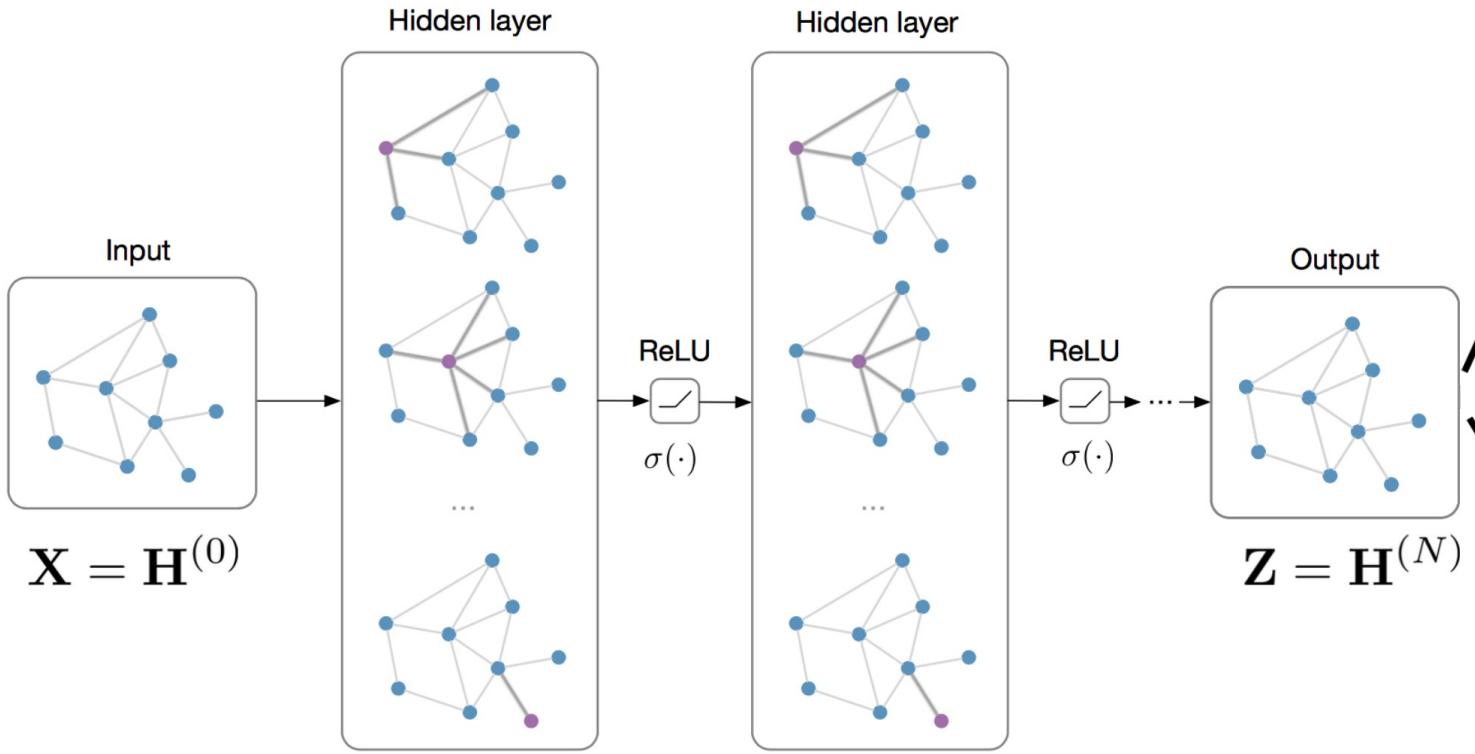
https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

How do we use GNN / GCN for real problems?

https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

Classification and Link Prediction with GNNs / GCNs

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



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)

Link prediction:

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

Kipf & Welling (NIPS BDL 2016)

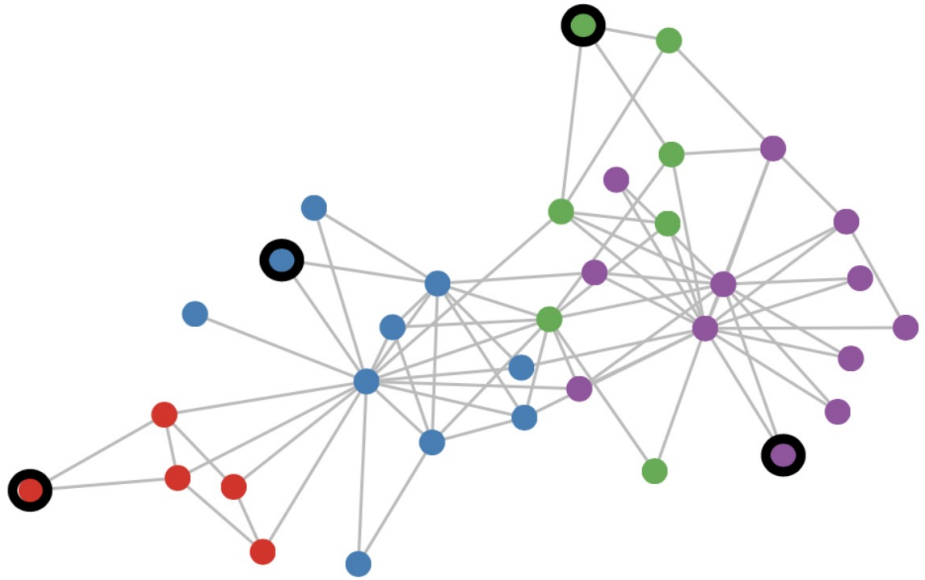
"Graph Auto-Encoders"

https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

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

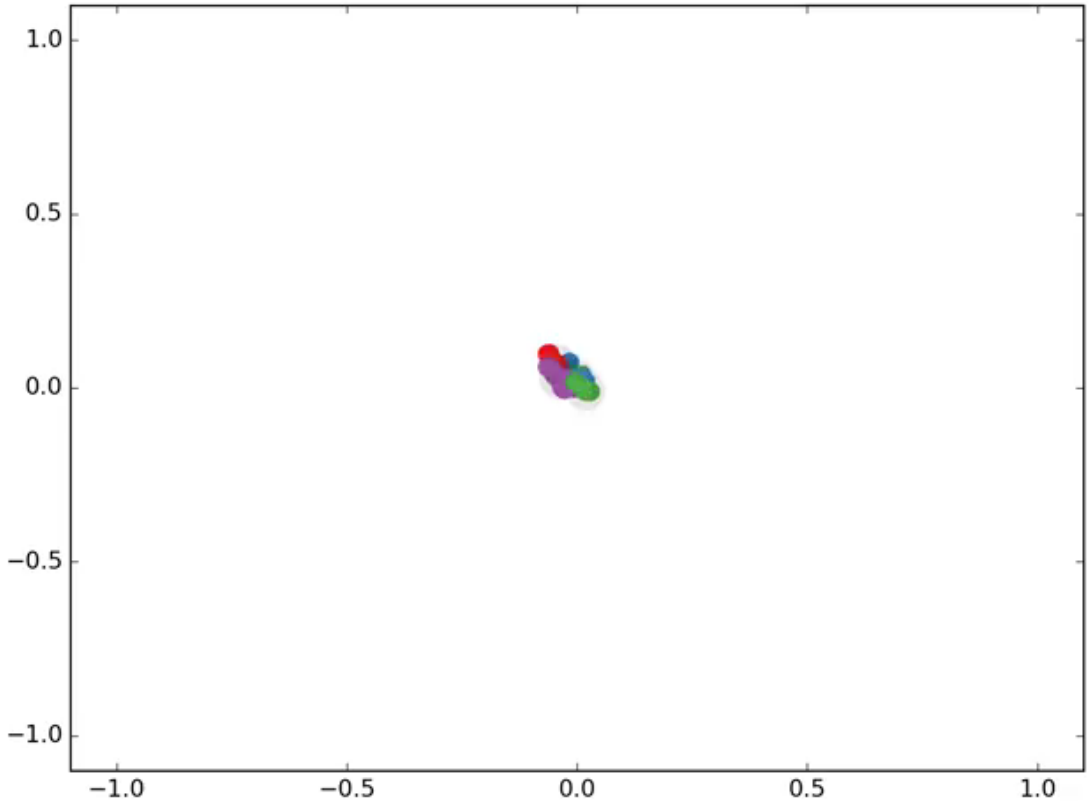
$\mathbf{Y}$ label matrix

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

https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

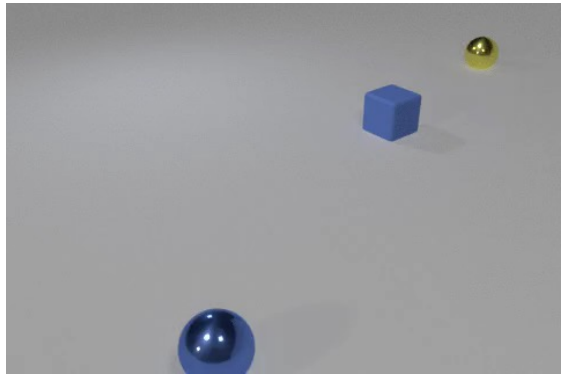
Semi-supervised Classification on Graphs

- Latent space dynamics for 300 training iterations
- Single label per class
- Labeled nodes are highlighted.

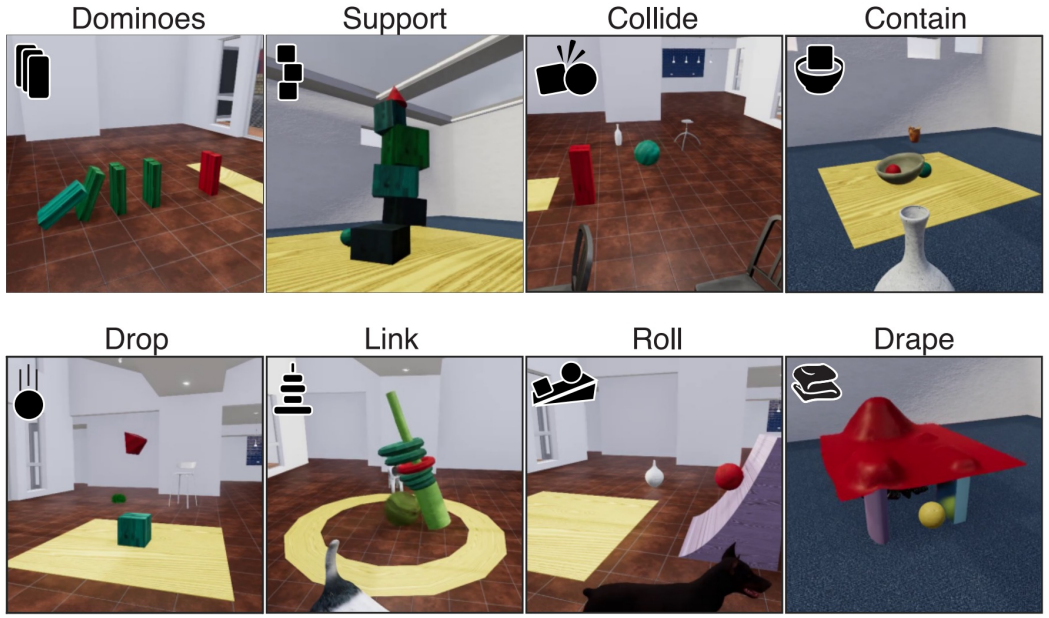


https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

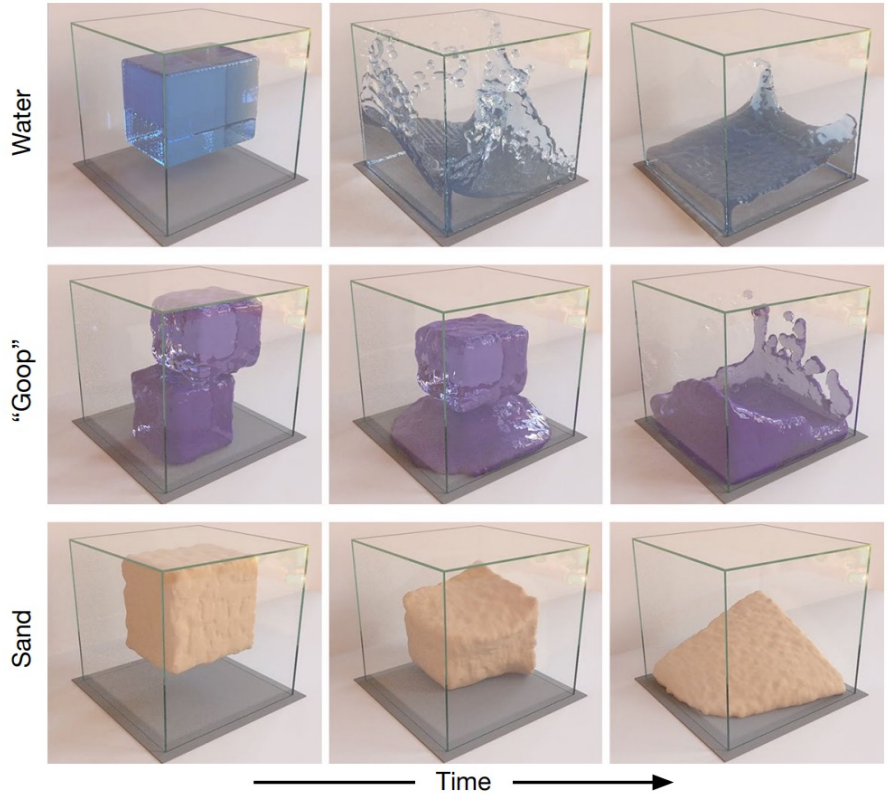
GNN for physical model



Chen et al, Comphy dataset



Bear, Elias et al, Physion dataset

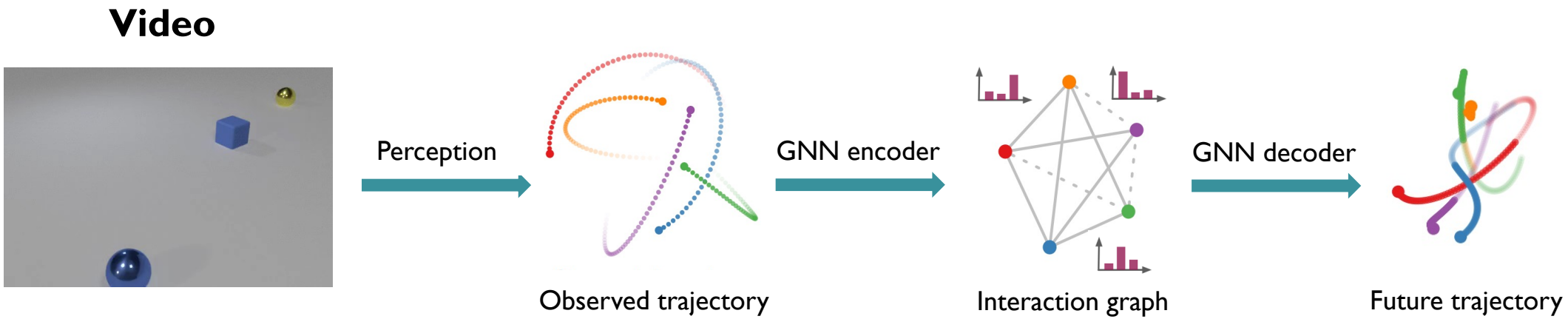


Alvaro et al, Learning to Simulate Complex Physics with Graph Networks

- Scene: dense or sparse physical objects
- Model: GNN to capture objects interaction

GNN for relation inference

- What's the relation between objects? (same charged/opposite charged/no charge)
- The future trajectory of objects?

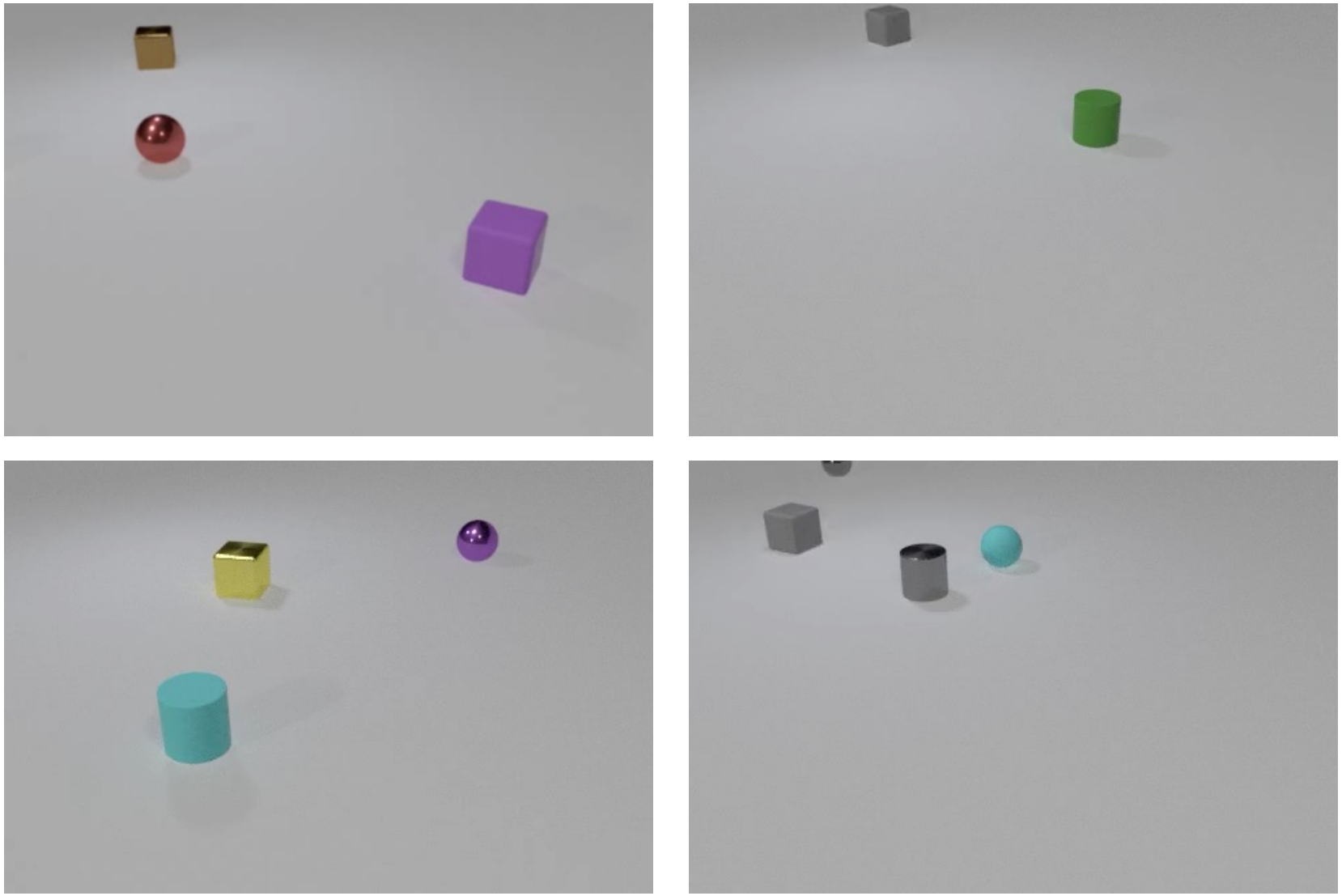


Task framework:

- Given video observation
- Video → trajectory
- Infer pair-wise relation of objects
- Predict future trajectory based on history trajectory and inferred relation

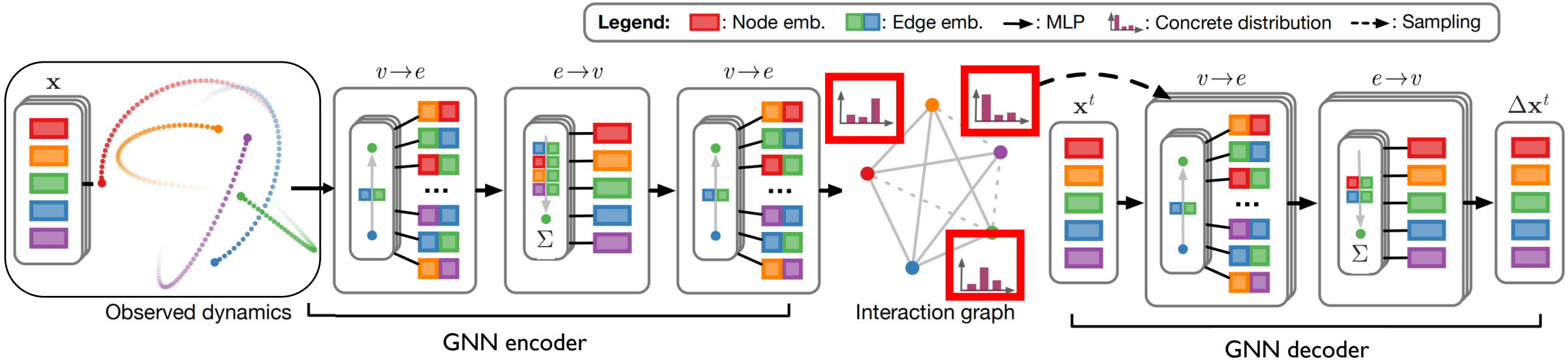
https://www.cs.ubc.ca/~lsigal/532S_2018W2/Lecture18a.pdf

GNN for relation inference



GNN for relation inference

Kipf et al. Neural relational inference for interacting systems. ICML2018



$\mathbf{X} \in \mathbb{R}^{T \times N \times d}$

T: time steps
N: num of objects
d: motion feature dimension

$$\mathbf{h}_j^1 = f_{\text{emb}}(\mathbf{x}_j)$$
$$v \to e: \quad \mathbf{h}_{(i,j)}^1 = f_e^1([\mathbf{h}_i^1, \mathbf{h}_j^1])$$
$$e \to v: \quad \mathbf{h}_j^2 = f_v^1\left(\sum_{i \neq j} \mathbf{h}_{(i,j)}^1\right)$$
$$v \to e: \quad \mathbf{h}_{(i,j)}^2 = f_e^2([\mathbf{h}_i^2, \mathbf{h}_j^2])$$

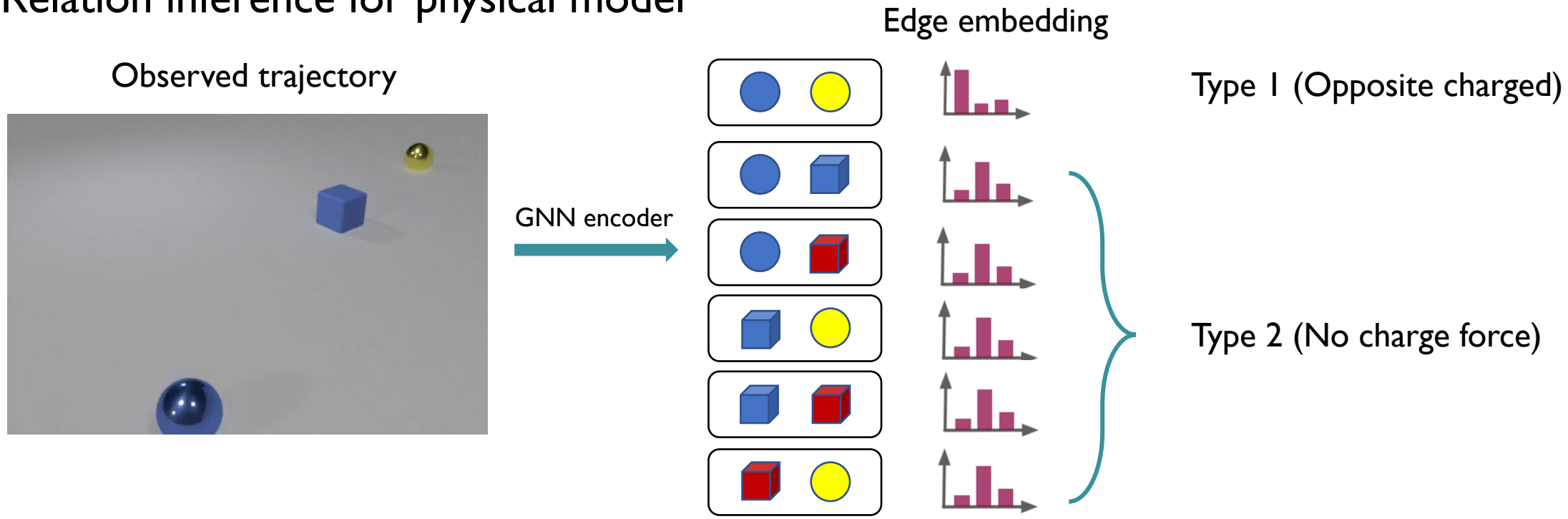
$$v \to e: \quad \tilde{\mathbf{h}}_{(i,j)}^t = \sum_k z_{ij,k} \tilde{f}_e^k([\mathbf{x}_i^t, \mathbf{x}_j^t])$$
$$e \to v: \quad \boldsymbol{\mu}_j^{t+1} = \mathbf{x}_j^t + \tilde{f}_v\left(\sum_{i \neq j} \tilde{\mathbf{h}}_{(i,j)}^t\right)$$
$$p(\mathbf{x}_j^{t+1} | \mathbf{x}^t, \mathbf{z}) = \mathcal{N}(\boldsymbol{\mu}_j^{t+1}, \sigma^2 \mathbf{I})$$

- GNN encodes edge embeddings for relation inference (can be supervised or unsupervised)
- Different relationship types correspond to different interaction modules in the GNN decoder
- GNN decodes node embeddings for trajectory prediction (supervised)

GNN for relation inference

Kipf et al. Neural relational inference for interacting systems. ICML2018

Relation inference for physical model



- The charge property of objects is expressed by the charge relationship between objects
- Different relationship types reflect different types of forces
- Other types of forces such as springs and light rods can also be considered
- Relations are constant

GNN for relation inference

Kipf et al. Neural relational inference for interacting systems. ICML2018

Accuracy (in %) of unsupervised interaction recovery

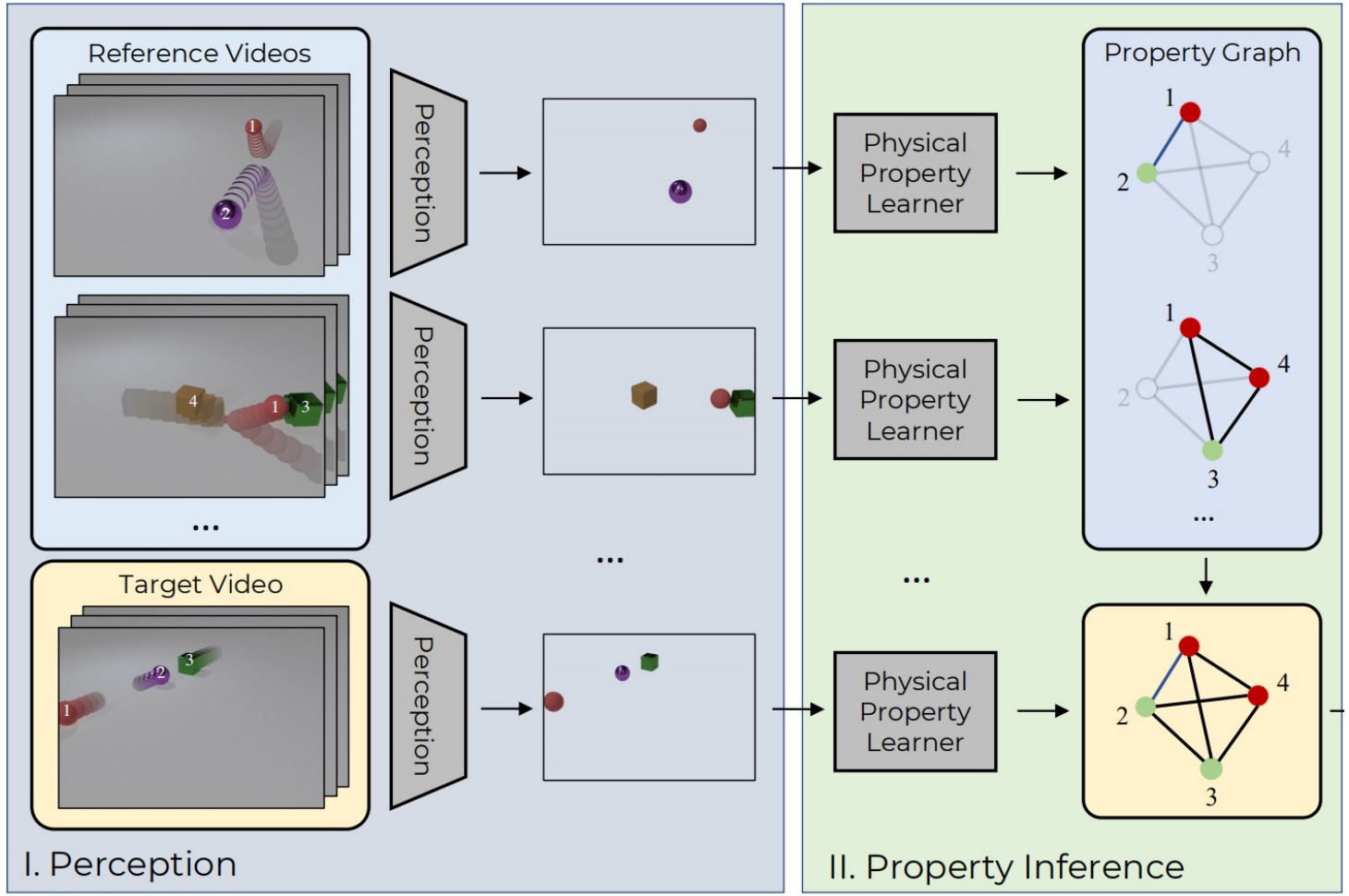
Model	Springs	Charged	Kuramoto
	5 objects		
Corr. (path)	52.4±0.0	55.8±0.0	62.8±0.0
Corr. (LSTM)	52.7±0.9	54.2±2.0	54.4±0.5
NRI (sim.)	**99.8**±0.0	59.6±0.8	–
NRI (learned)	**99.9**±0.0	**82.1**±0.6	**96.0**±0.1
Supervised	99.9±0.0	95.0±0.3	99.7±0.0
	10 objects		
Corr. (path)	50.4±0.0	51.4±0.0	59.3±0.0
Corr. (LSTM)	54.9±1.0	52.7±0.2	56.2±0.7
NRI (sim.)	**98.2**±0.0	53.7±0.8	–
NRI (learned)	**98.4**±0.0	**70.8**±0.4	**75.7**±0.3
Supervised	98.8±0.0	94.6±0.2	97.1±0.1

- Recovers the ground-truth interaction graph with high accuracy on most tasks.
- Unsupervised method of NRI is comparable to the supervised benchmark

GNN for relation inference

Chen et al. COMPHY: Compositional physical reasoning of objects and events from videos. ICLR 2022

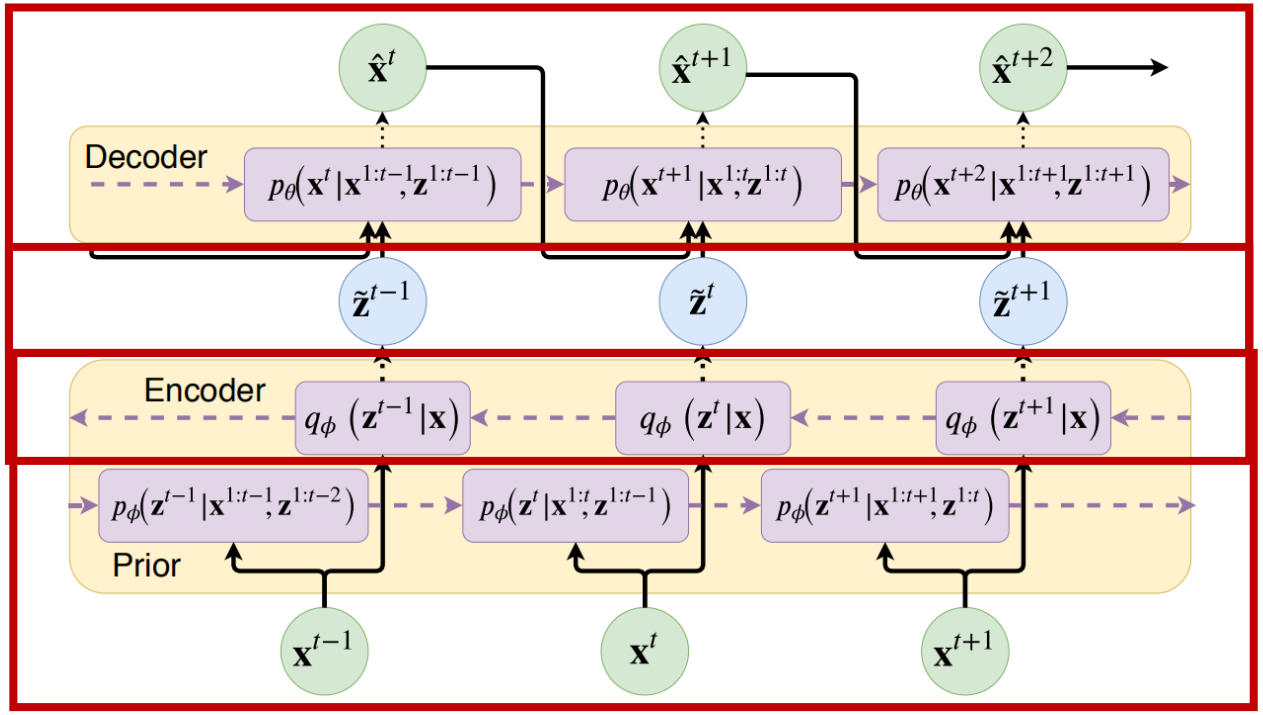
- Latest dataset COMPHY provides 4 reference videos for each target video.
- The object in the reference video has appeared in the target video.
- Property graph is expected to be better inferred with reference videos.



GNN for Dynamic Neural Relational Inference

Graber, Schwing. Dynamic Neural Relational Inference. CVPR 2020

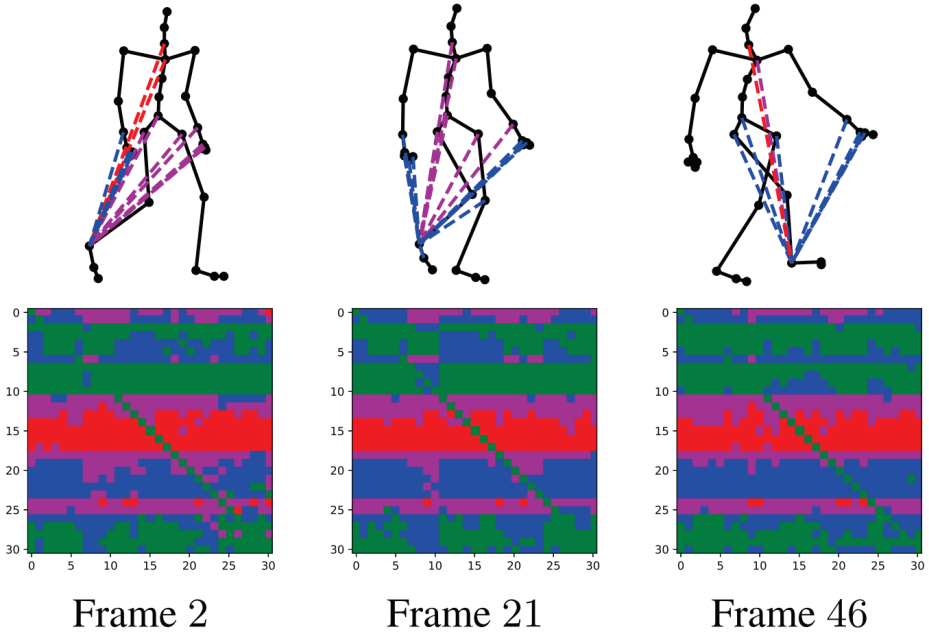
- Produce edge embeddings at every time step
- LSTM encodes the past and future history of entity relations
- Approximate posterior is computed as a function of both the past and future.
- Edge variables are sampled from the approximate posterior
  (to select edge models for the decoder)
- The decoder evolves a hidden state using this GNN and the previous predictions
- Predicts the state of the entities at the next time step.



GNN for Dynamic Neural Relational Inference

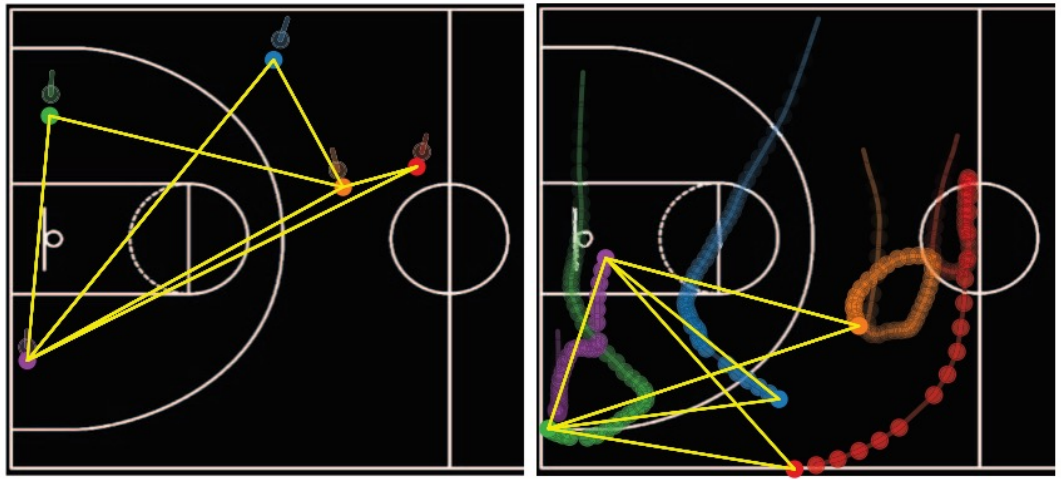
Graber, Schwing. Dynamic Neural Relational Inference. CVPR 2020

Motion prediction



Different motion stages imply different motion relationships of joint points

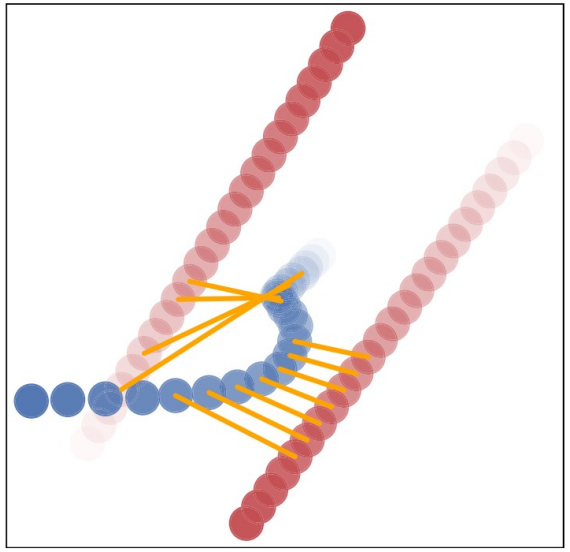
NBA basketball game dataset



The players' coordination changes over time

GNN for Dynamic Neural Relational Inference

Graber, Schwing. Dynamic Neural Relational Inference. CVPR 2020

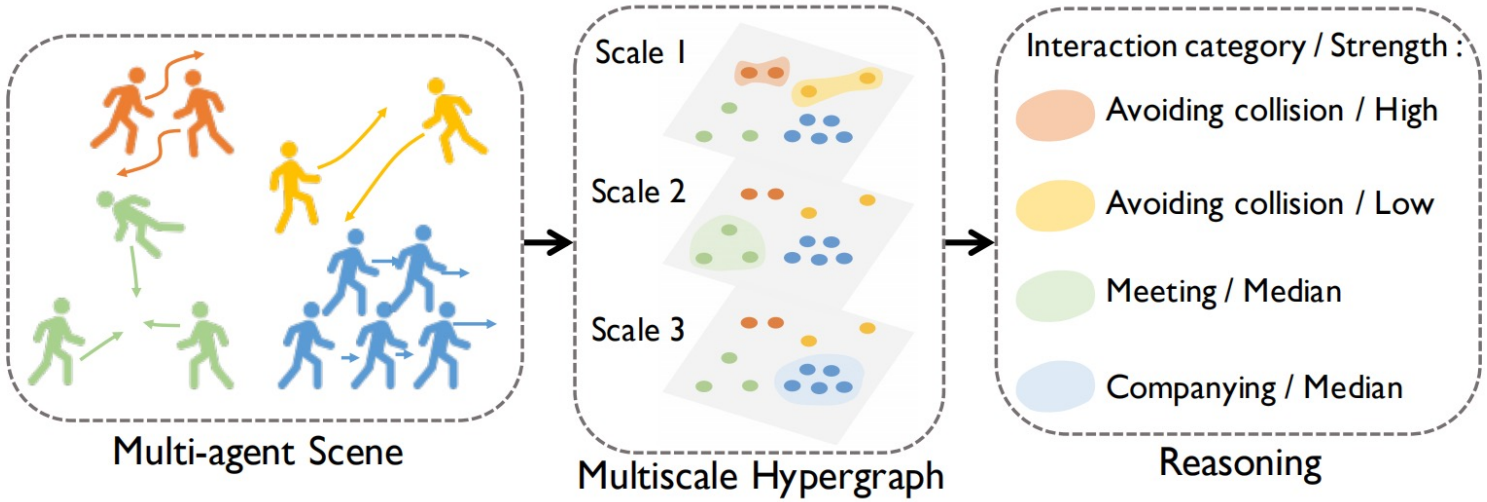


Dynamic relation in physical dataset

- The first two (red) move with a constant velocity in some direction.
- The third (blue) is initialized with a random velocity but is additionally "pushed" away by the other particles whenever the distance separating them is less than 1.(Such high-frequency function is not easy to fit in static NRI)
- Dynamic relationships reflect dynamic forces

GNN for hyper-relational Inference

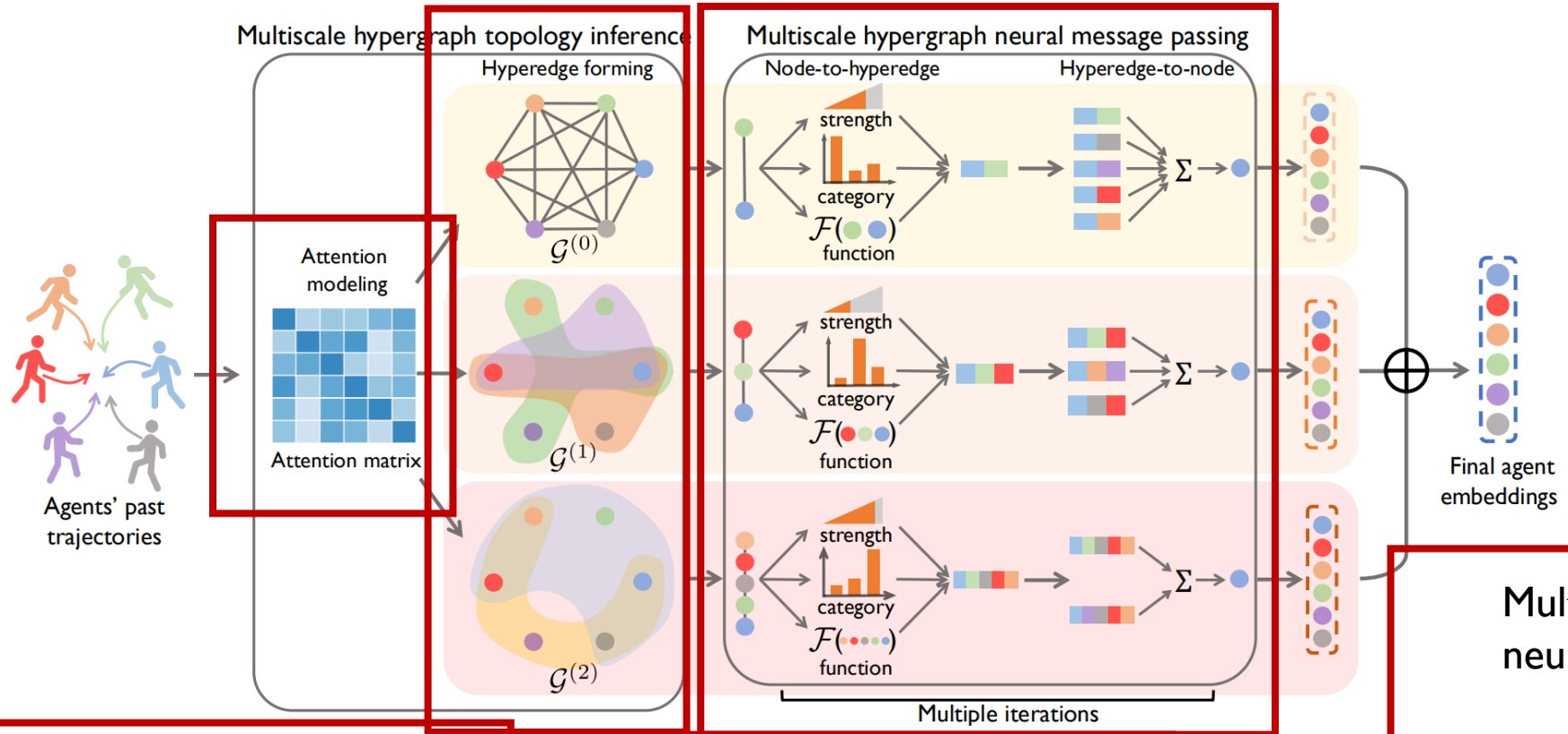
Xu, Li, Ni, Zhang, Chen. GroupNet. CVPR 2022



- Captures both pair-wise and group-wise interactions among multiple agents from their trajectories.
- Infers the interaction category and strength for each interaction group.

GNN for hyper-relational Inference

Xu, Li, Ni, Zhang, Chen. GroupNet. CVPR 2022



Attention modeling

$$\mathbf{A}_{i,j} = \mathbf{q}_i^\top \mathbf{q}_j / (\|\mathbf{q}_i\|_2 \|\mathbf{q}_j\|_2)$$

Hyperedge forming

$$e_i^{(s)} = \arg\max_{\Omega \subseteq \mathcal{V}} \|\mathbf{A}_{\Omega,\Omega}\|_{1,1},$$

$$\text{s.t. } |\Omega| = K^{(s)},\ v_i \in \Omega,\ i = 1, \ldots, N$$

Multiscale **hypergraph** neural message passing

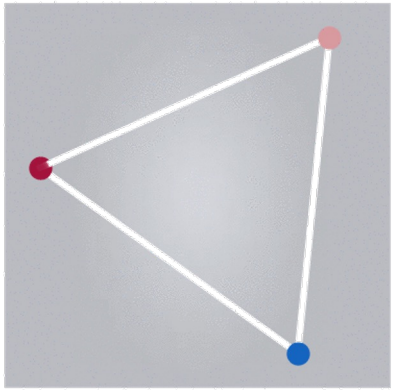
v→e $$\mathbf{e}_i = r_i \sum_{\ell=1}^{L} c_{i,\ell} \mathcal{F}_\ell \Big(\sum_{v_j \in e_i} \mathbf{v}_j \Big) \in \mathbb{R}^d$$

e→v $$\mathbf{v}_i \leftarrow f_v \big([\mathbf{v}_i, \sum_{e_j \in \mathcal{E}_i} \mathbf{e}_j] \big) \in \mathbb{R}^d$$

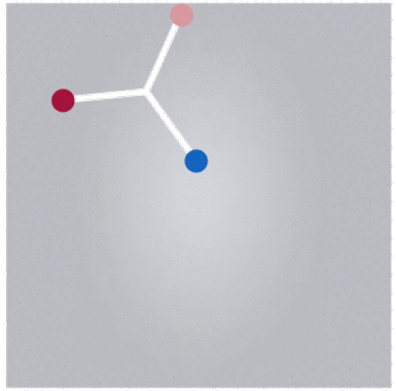
GNN for hyper-relational Inference

Xu, Li, Ni, Zhang, Chen. GroupNet. CVPR 2022

Free



Spring

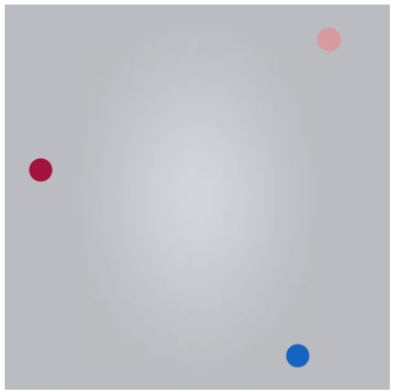


Light bar

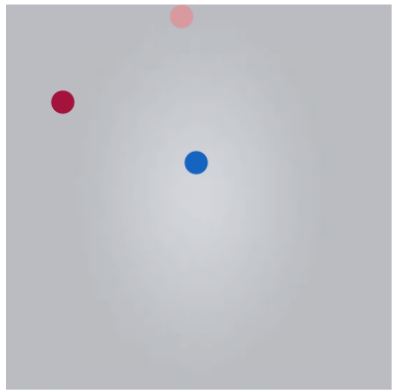
GNN for hyper-relational Inference

Xu, Li, Ni, Zhang, Chen. GroupNet. CVPR 2022

Free

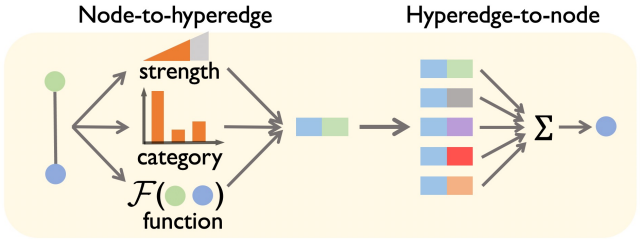


Spring



Light bar

Task: Given the trajectories **only**, recognize the interaction categories during prediction



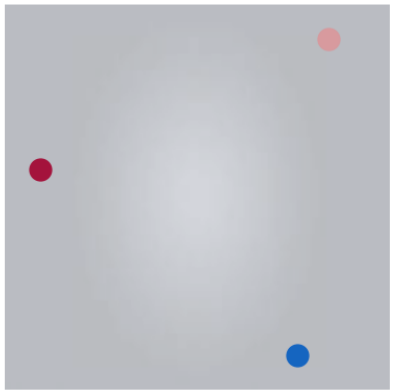
$$\mathbf{e}_i = r_i \sum_{\ell=1}^{L} c_{i,\ell} \mathcal{F}_\ell \Big(\sum_{v_j \in e_i} \mathbf{v}_j \Big) \in \mathbb{R}^d,$$

category

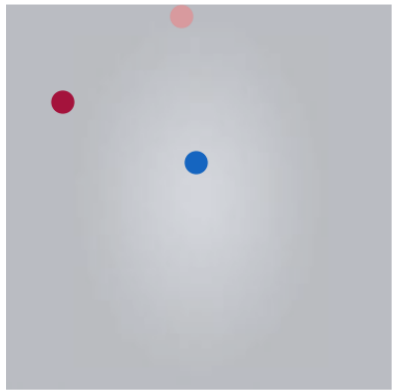
GNN for hyper-relational Inference

Xu, Li, Ni, Zhang, Chen. GroupNet. CVPR 2022

Free



Spring



Light bar

Task: Given the trajectories **only**, recognize the interaction categories during prediction

Table 1: Comparison of accuracy (mean ± std in %) of on interaction category recognition.

	Corr. (path)	Corr. (LSTM)	NRI	**Ours**	Supervised
2-types	78.3±0.0	75.3 ± 2.5	89.7 ± 3.5	**99.3** ± 0.3	99.5 ± 0.2
3-types	–	–	64.9 ± 2.3	**91.5** ± 2.4	98.7 ± 0.7

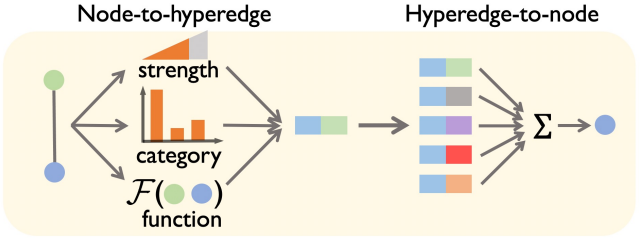
GNN for hyper-relational Inference

Xu, Li, Ni, Zhang, Chen. GroupNet. CVPR 2022

Low charge

High charge

Strength-Charge Curve



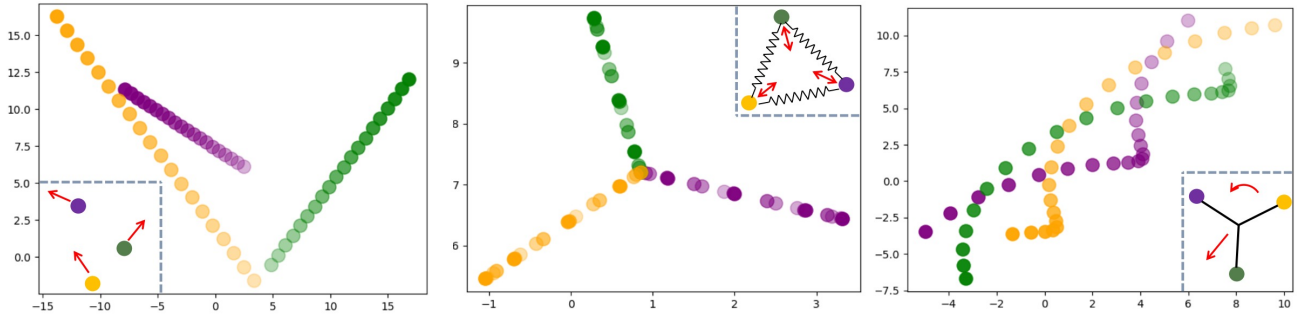
$$\mathbf{e}_i = r_i \sum_{\ell=1}^{L} c_{i,\ell} \mathcal{F}_\ell \Big(\sum_{v_j \in e_i} \mathbf{v}_j \Big) \in \mathbb{R}^d,$$

strength

GNN for hyper-relational Inference

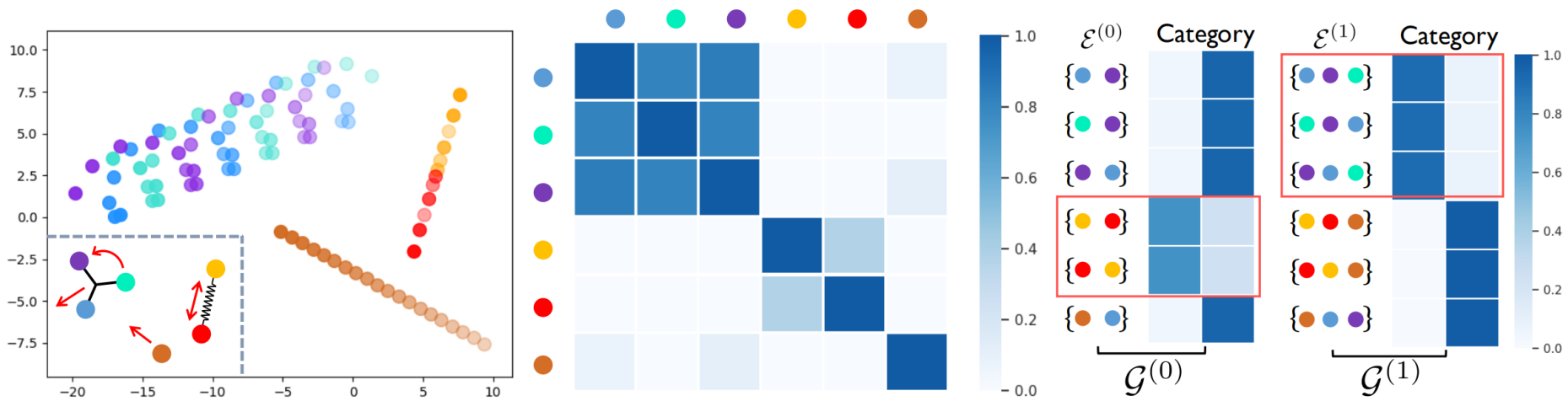
Xu, Li, Ni, Zhang, Chen. GroupNet. CVPR 2022

Trajectory samples under three interaction categories.



(a) Free (b) Spring (c) Light bar

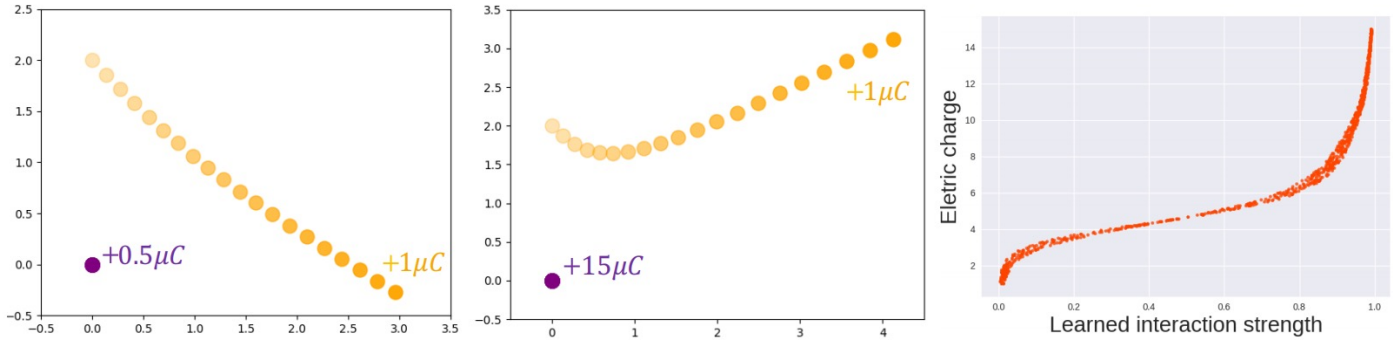
- The particle trajectories contain a three-group with a light bar, a two-group with a spring and an individual particle.
- The heatmap of the learnt affinity matrix via affinity modeling.
- The red box represents the three-group and two-group we inferred.



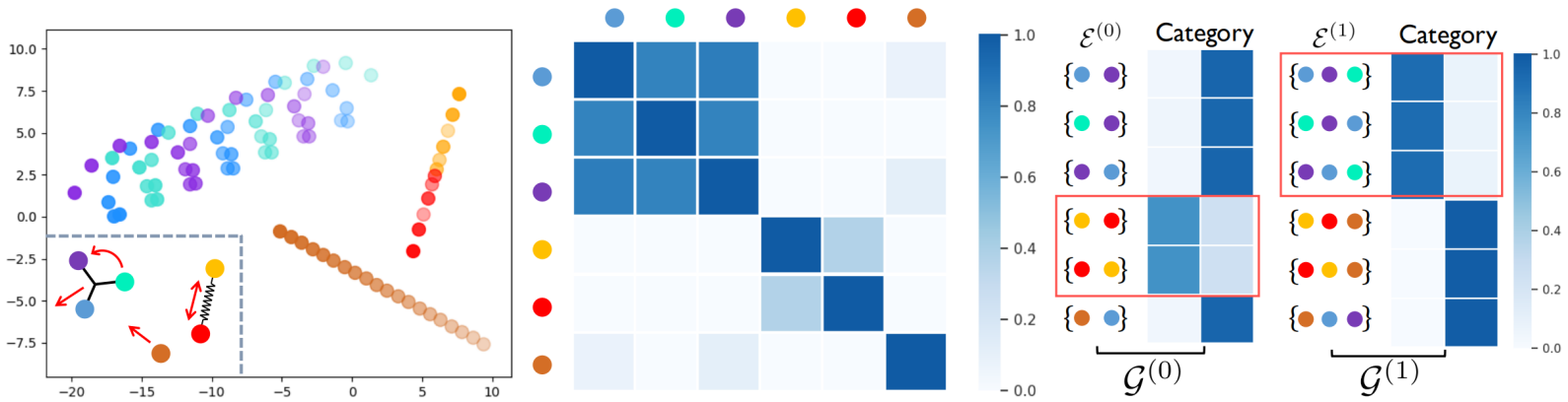
GNN for hyper-relational Inference

Xu, Li, Ni, Zhang, Chen. GroupNet. CVPR 2022

- Both category and strength of interaction can be captured



(a) Low charge (b) High charge (c) Relation curve

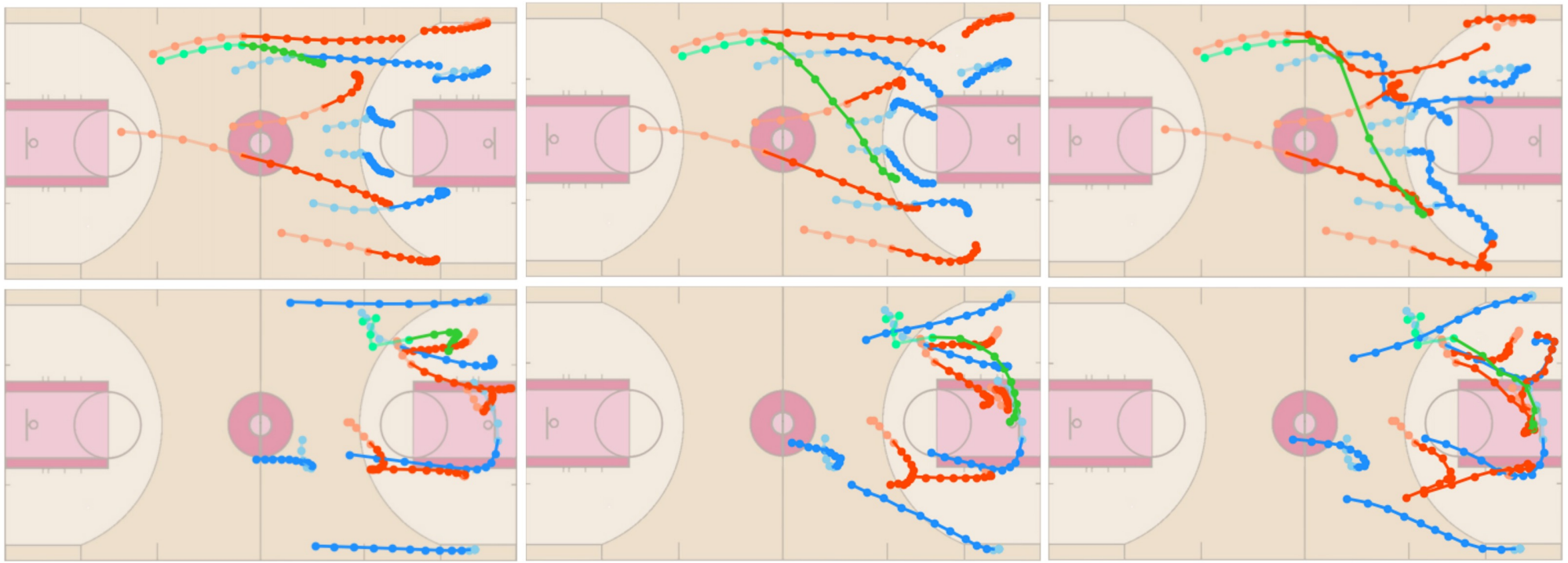


GNN for hyper-relational Inference

Xu, Li, Ni, Zhang, Chen. GroupNet. CVPR 2022

Qualitative results on the NBA dataset

- The best trajectory among 20 predictions
- Red/blue——two teams
- Green ——basketball
- Light color ——past trajectory



(a) NMMP (b) GroupNet (c) GT

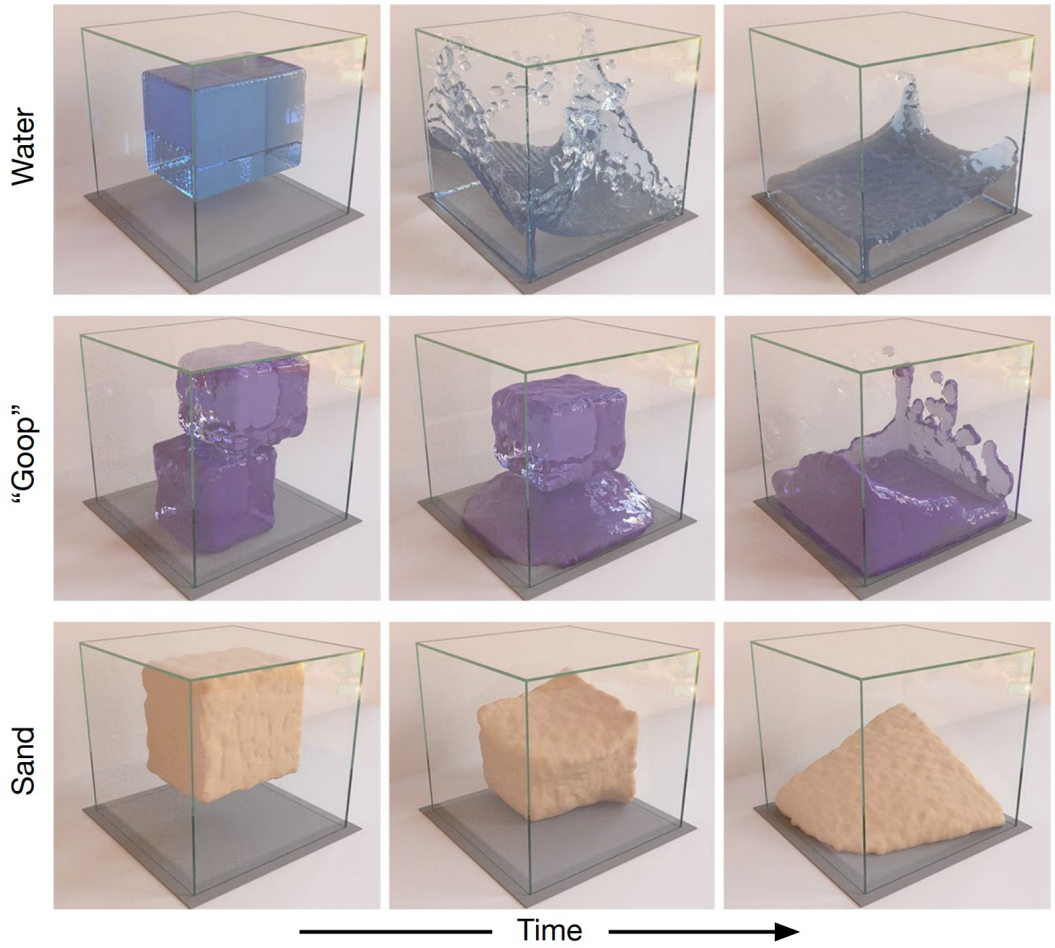
GNN for complex Physics

Alvaro et al. Learning to Simulate Complex Physics with Graph Networks. ICML 2020

GNN for a wide variety of challenging physical domains:

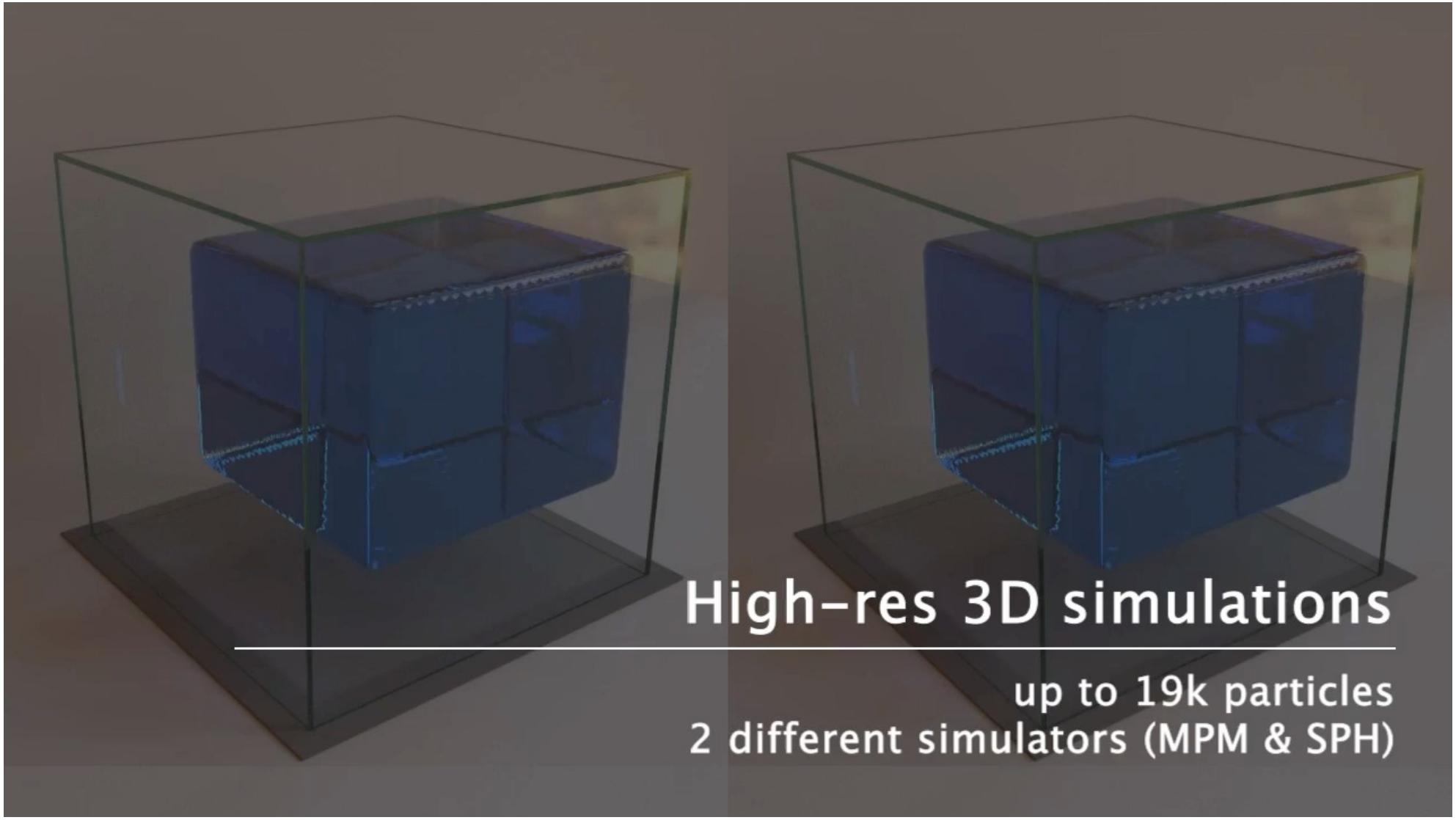
- Fluids
- Rigid solids
- deformable materials

Physical model——thousands of particles
Particles——nodes in a graph,
Dynamics——message-passing for prediction



GNN for complex Physics

Alvaro et al. Learning to Simulate Complex Physics with Graph Networks. ICML 2020

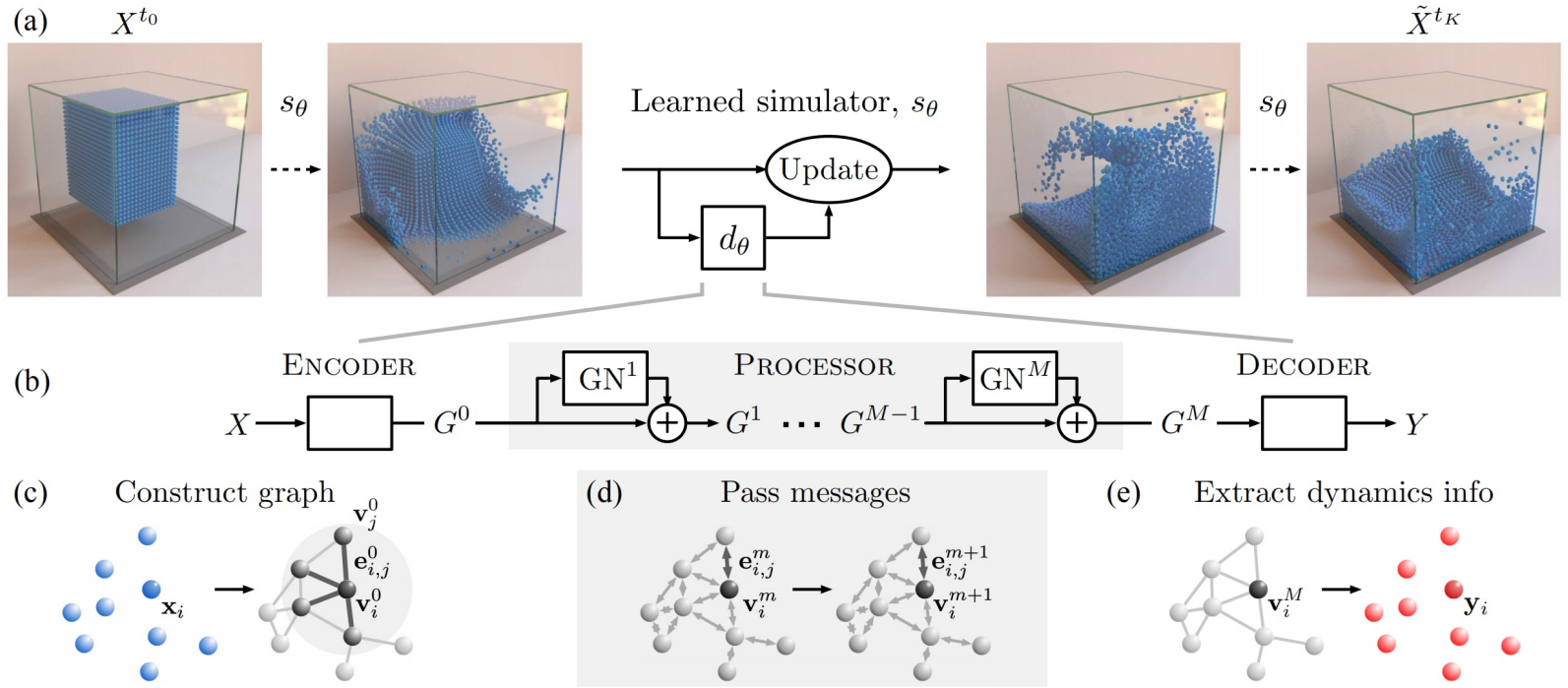


GNN for complex Physics

Alvaro et al. Learning to Simulate Complex Physics with Graph Networks. ICML 2020

Encode-process-decode scheme:

- Encoder: constructs latent graph G from input state X
- Processor: performs M rounds of learned message-passing over the latent graphs $G^1, G^2, \ldots, G^M$
- Decoder: extracts dynamics information from the final latent graph G^M



Add edges between particles within a "connectivity radius" (local interactions prior)

Multi-step message-passing is critical for complex physics (especially for rigid object)

Future position and velocity are updated

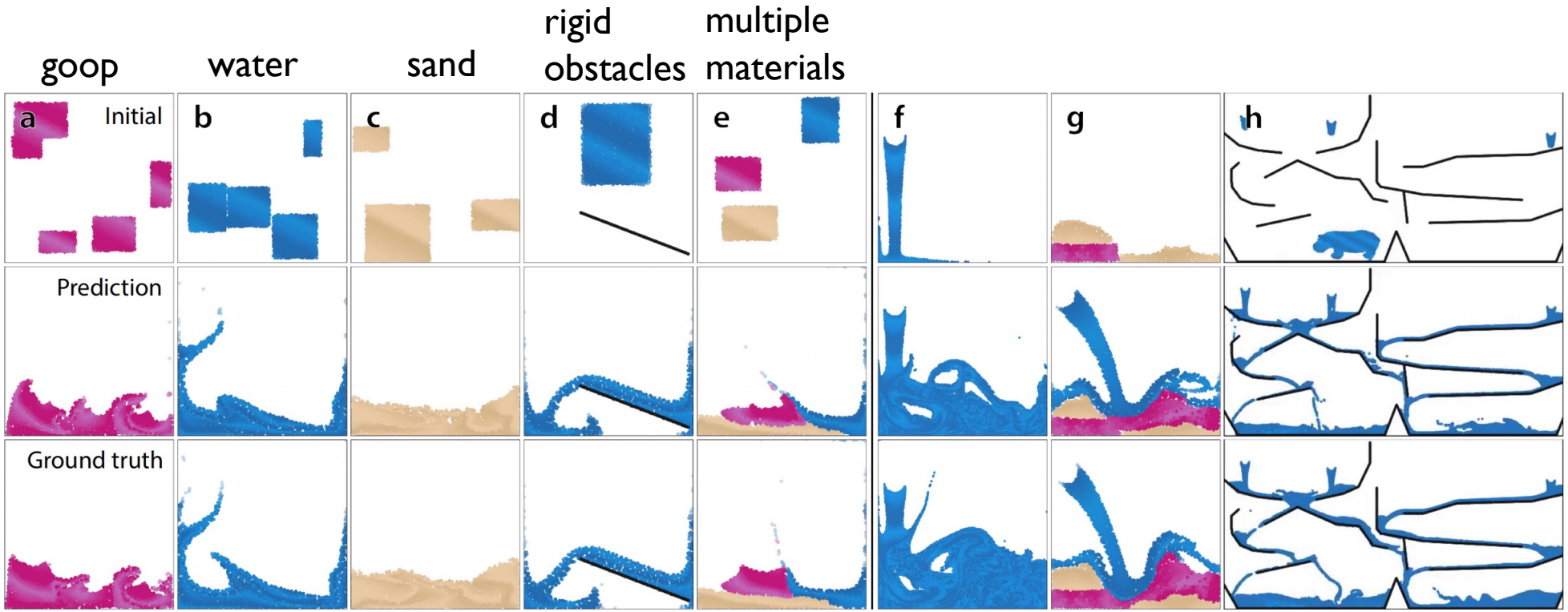
GNN for complex Physics

Alvaro et al. Learning to Simulate Complex Physics with Graph Networks. ICML 2020

Simulate many materials

Pre-trained models on several out-of-distribution:

- Water ramps→ high-res turbulence (f)
- Multi-material interactions → Multi-material interactions +unseen objects (g)
- generalizing on significantly larger domains (h)



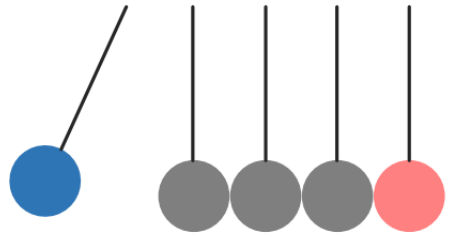
Hierarchical modeling for dynamic particle interaction

Li et al. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. ICLR 2019

Modeling of the long-range dependence via hierarchical modeling

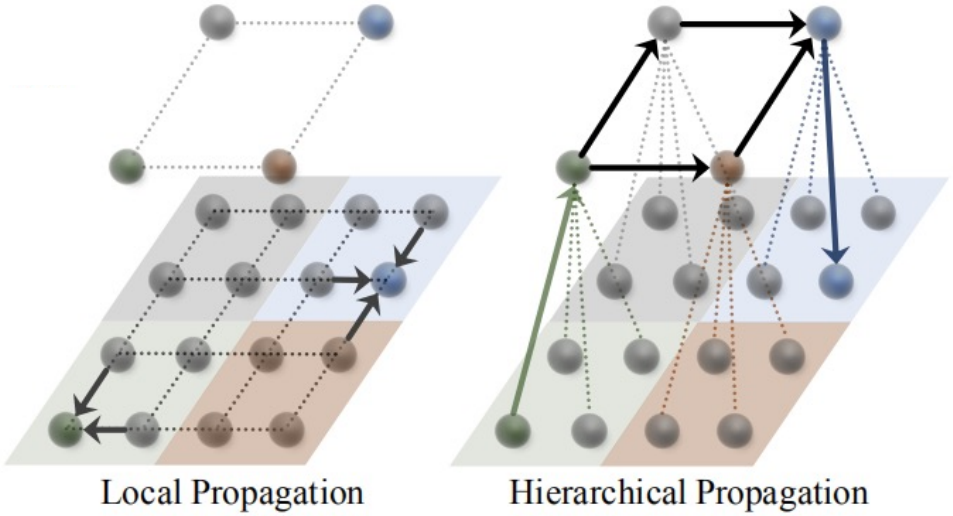
- A object consists of particles (leaf).
- Cluster the particles into several non-overlapping clusters.
- A new root particle for each cluster.

For handling instantaneous force propagation (long-range dependence)

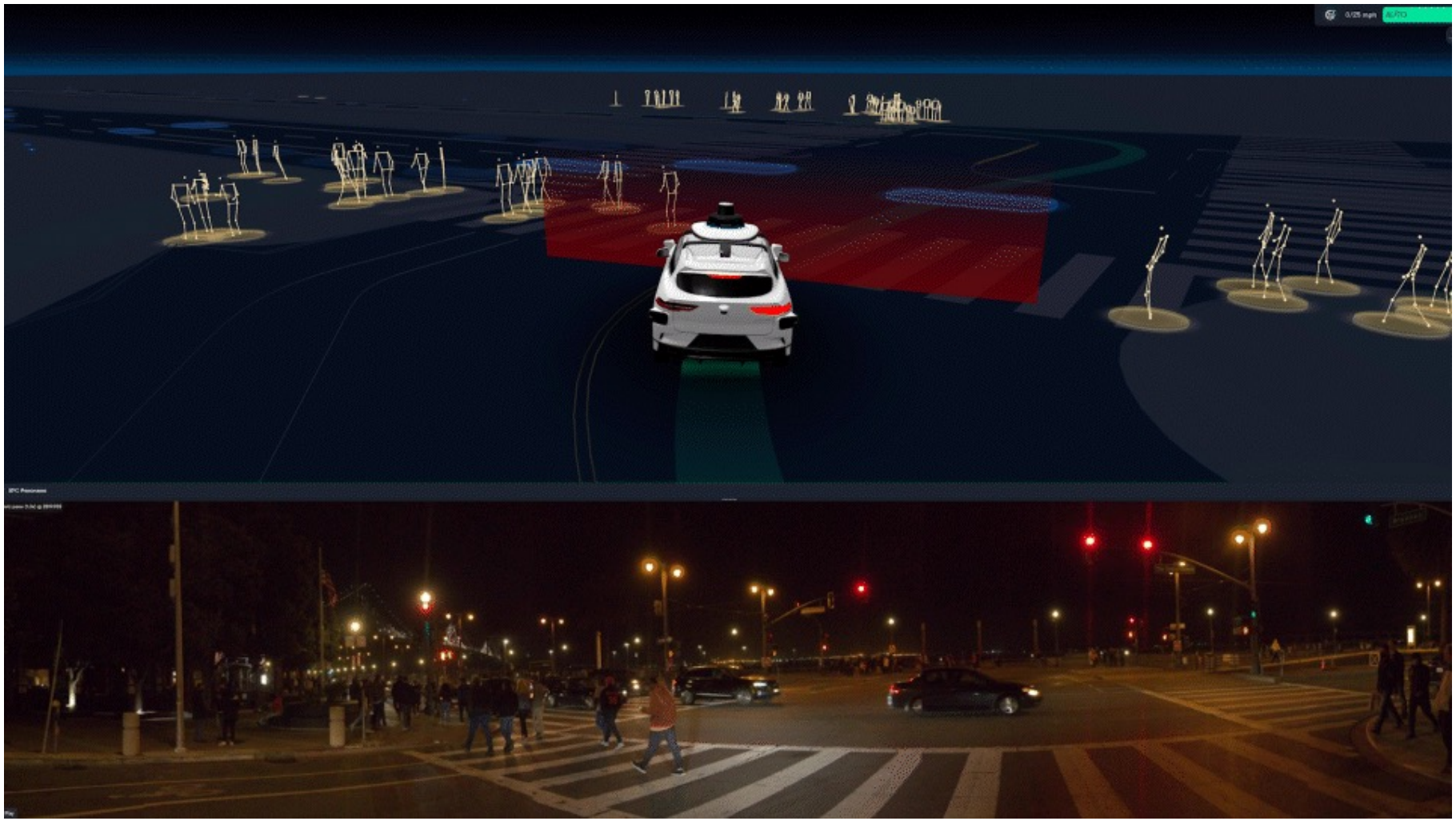


Multi-stage propagation

- Leaf → Leaf
- Leaf → Root
- Root → Root
- Root → Leaf

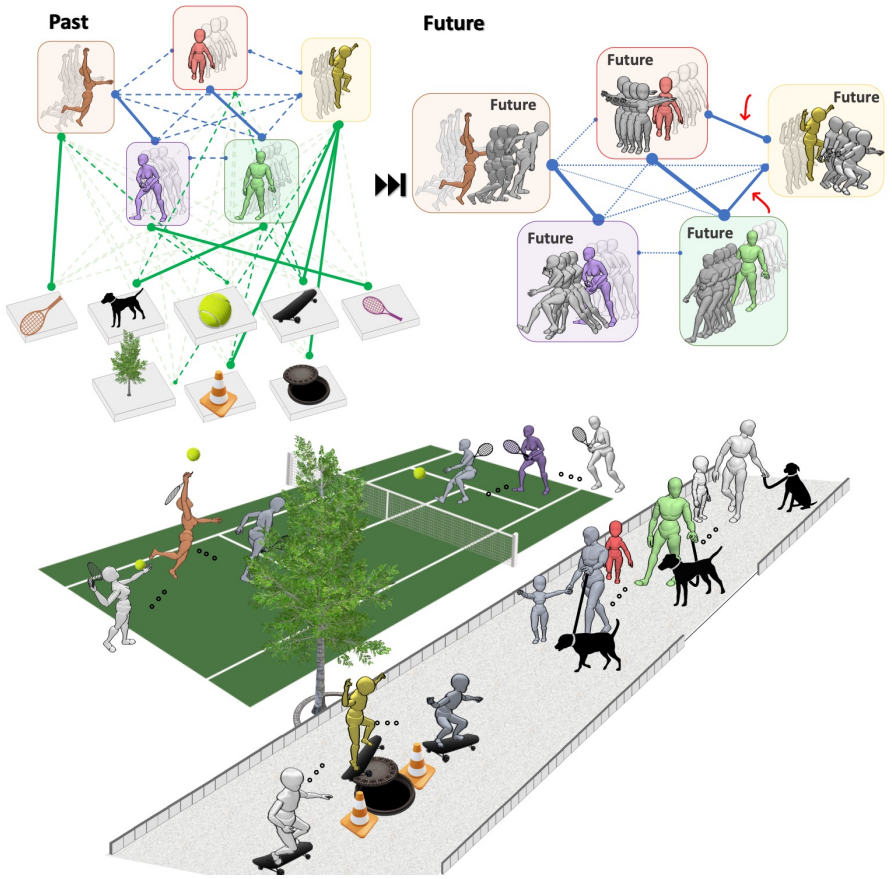


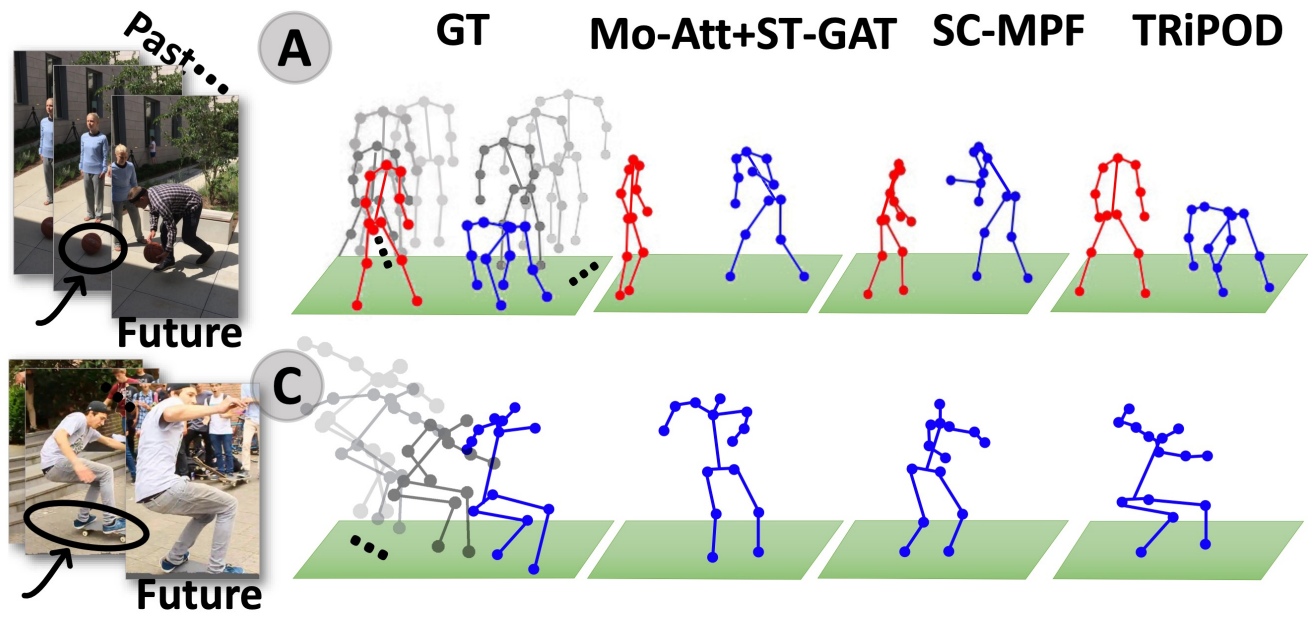
GCN for Multi-Person Behavior Prediction



https://blog.waymo.com/2022/02/utilizing-key-point-and-pose-estimation.html

GCN for Multi-Person Behavior Prediction

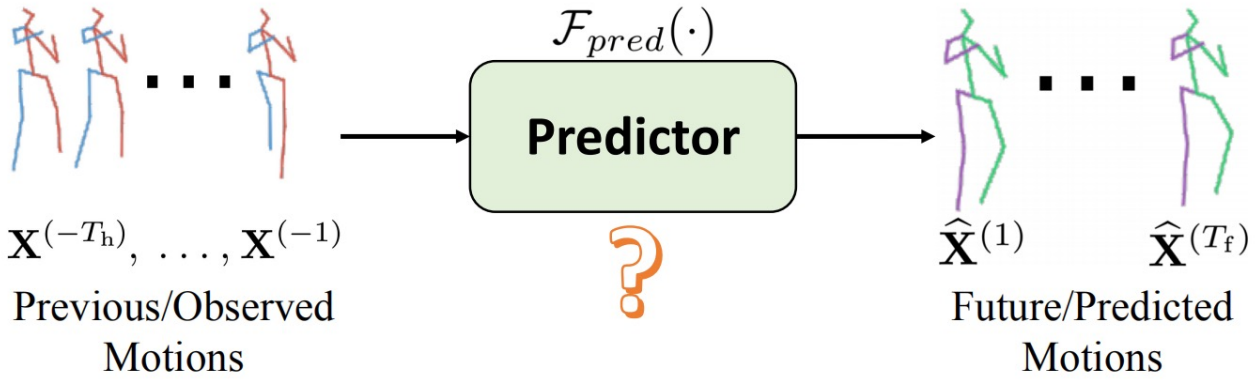




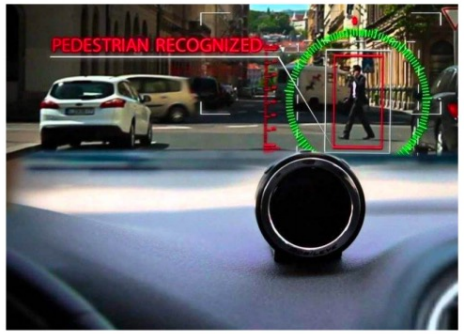
Vida Adeli, Mahsa Ehsanpour, Ian Reid, Juan Carlos Niebles,Silvio Savarese, Ehsan Adeli, Hamid Rezatofighi, TRiPOD: Human Trajectory and Pose Dynamics Forecasting in the Wild, ICCV 2021

GCN for Skeleton based Human Motion Prediction

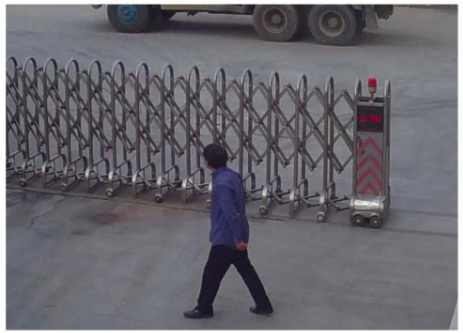
What is 3D skeleton-based human motion prediction?




Human-machine Interaction

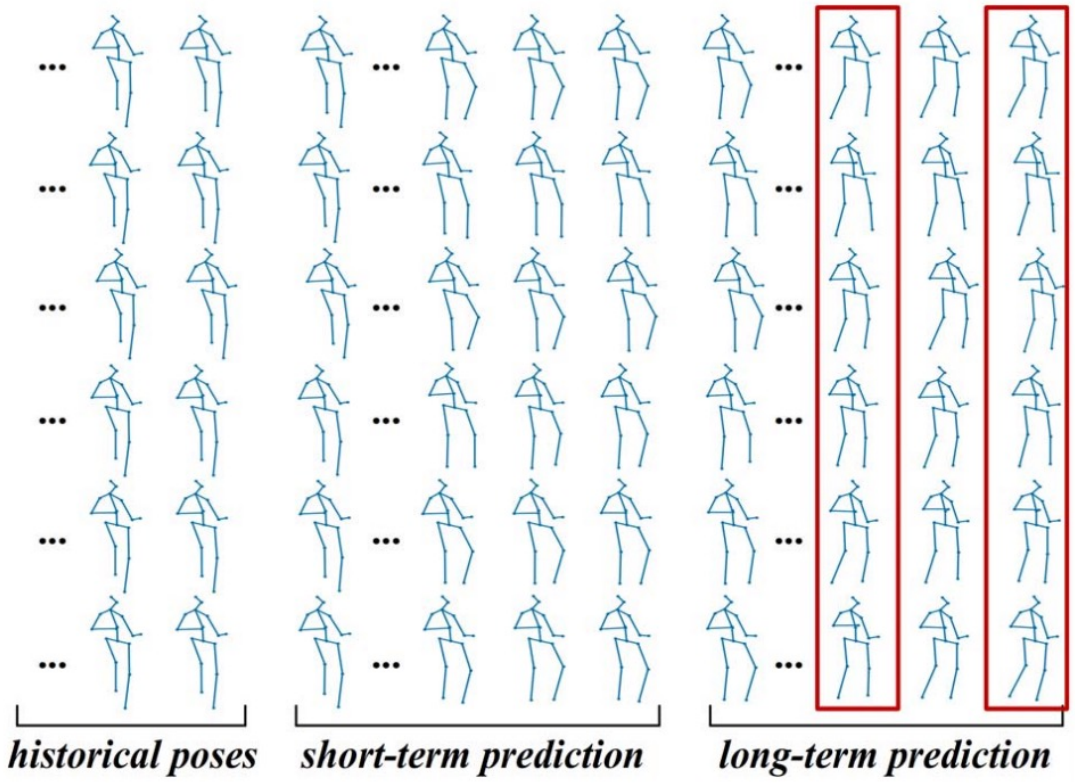


Automatic driving


Tracking and Surveillance

GCN for Skeleton based Human Motion Prediction

Short-term/long-term motion prediction



Varies of motion type (H3.6M dataset):

- Walking
- Eating
- Smoking
- Discussion
- Directions
- Greeting
- Phoning
- Posing
- Purchase
- Sitting
- …

Temporal modeling:

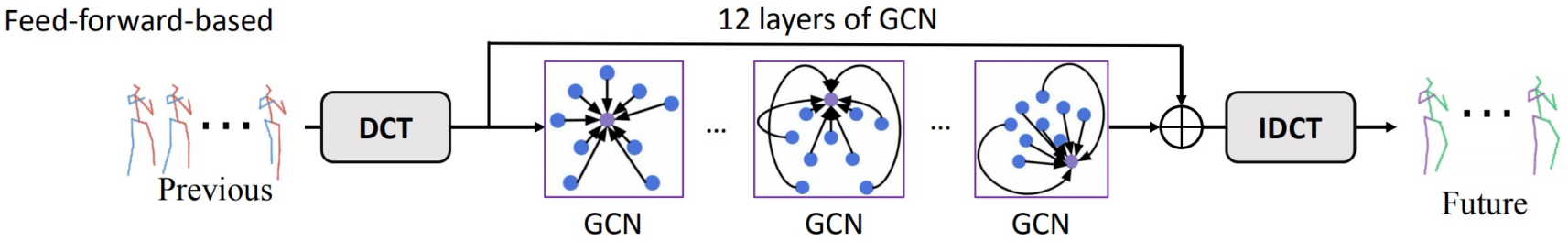
- Time convolution
- DCT coding

Skeleton modeling:

- Graph convolution network

GCN for Skeleton based Human Motion Prediction

Cui et al. Learning Dynamic Relationships for 3D Human Motion Prediction. CVPR 2020

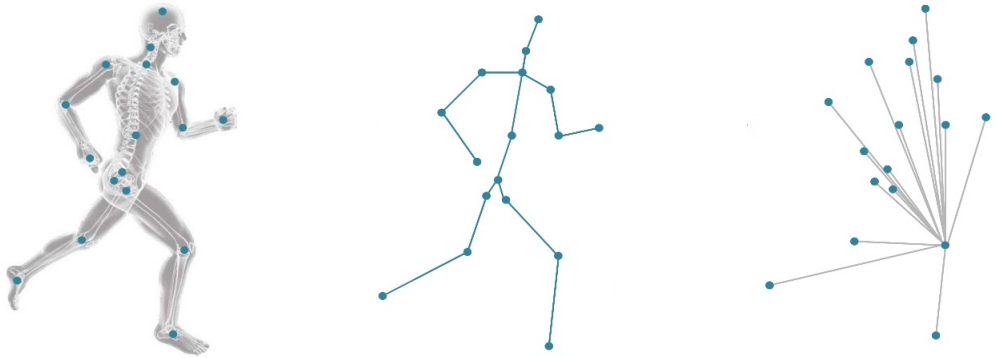


Temporal modeling:
- Time convolution
- DCT coding

Skeleton modeling:
- Graph convolution network

Graph convolution: $\boldsymbol{F}^{(l+1)} = g(\boldsymbol{F}^{(l)}) = \sigma[(\boldsymbol{A}_p^{(l)} \circ \boldsymbol{M} + \boldsymbol{Q}^{(l)})\boldsymbol{F}^{(l)}\boldsymbol{W}^{(l)}],$

partially constrained flexible

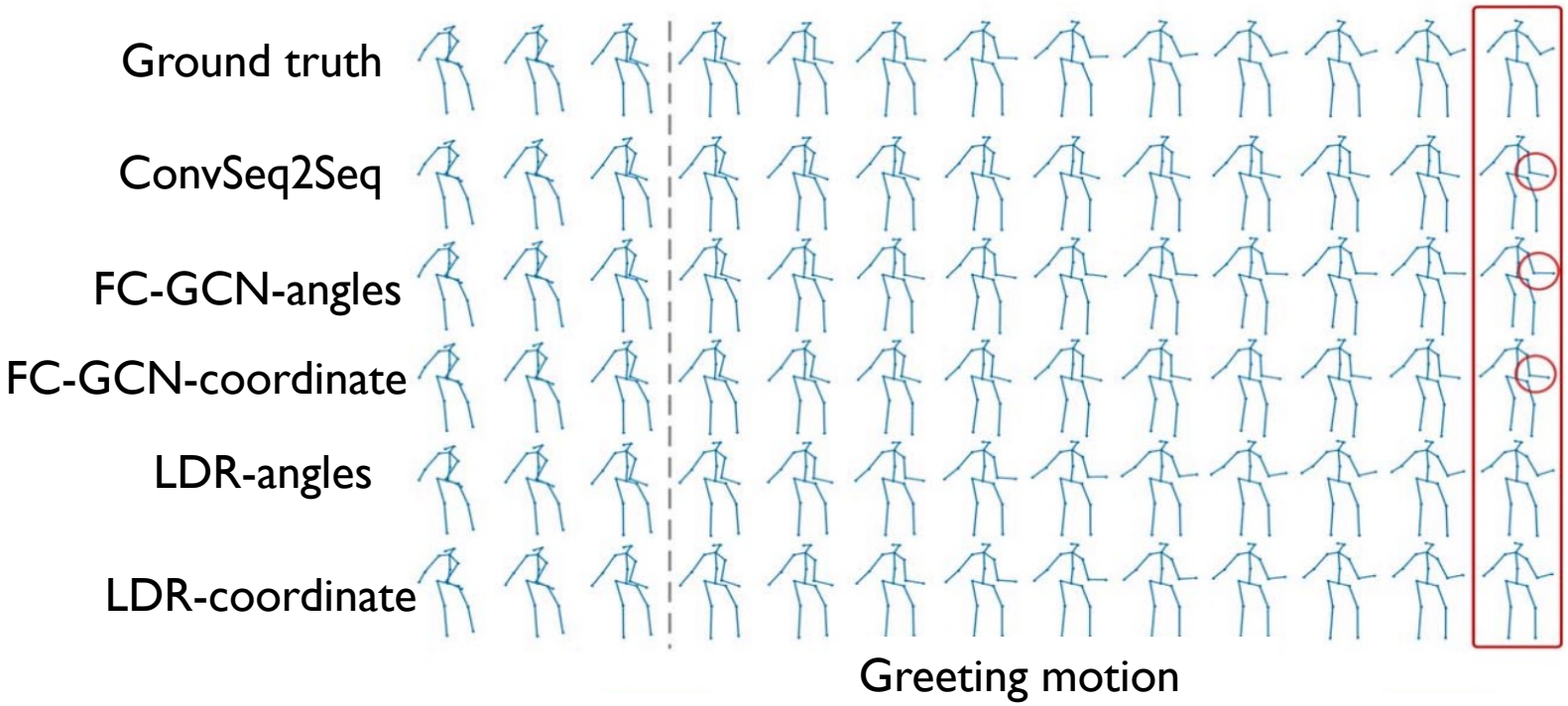


$\boldsymbol{F}^{(l)}$: feature in layer l
$\boldsymbol{M}$: skeleton mask matrix
$\boldsymbol{W}$: learnable weight matrix

Over-smoothing
- Deep low-pass filtering
- Possible consequence:
  The speed of each joint point tends to be consistent, and the posture does not change

GCN for Human Motion Prediction

Cui et al. Learning Dynamic Relationships for 3D Human Motion Prediction. CVPR 2020



Motion position coding:

- 3D coordinate (x,y,z)
- Angle of each joint

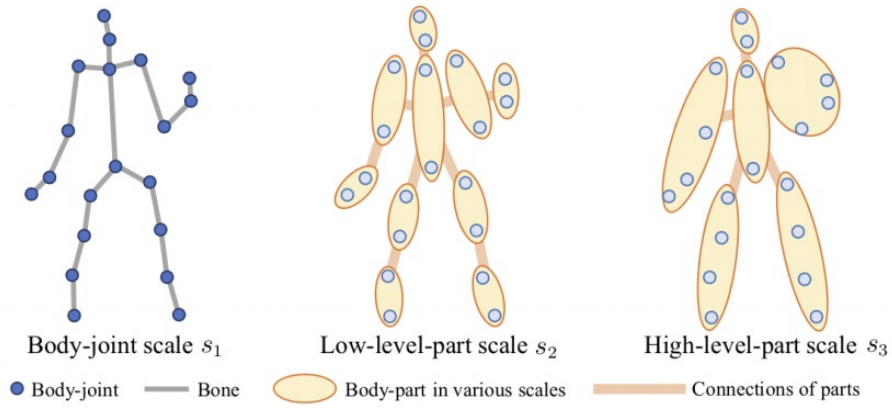
The red box represents the prediction result of the last frame (400 ms)

The red circle represents the wrong part of the model

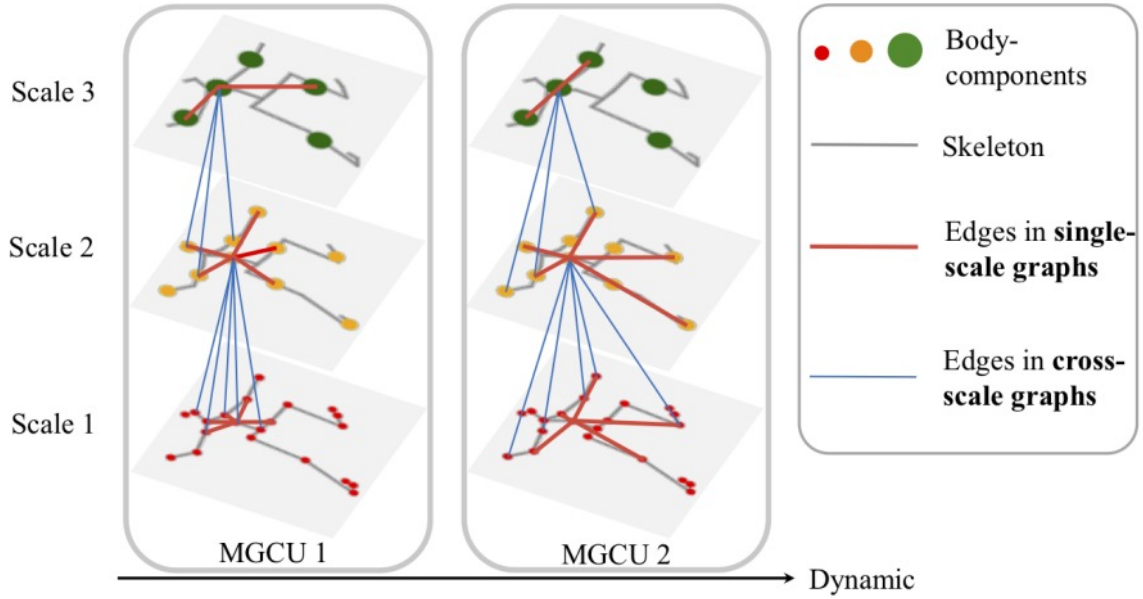
Multiscale GCN model for Human Motion Prediction

Li, Chen, et al. Dynamic Multiscale Graph Neural Networks for 3D skeleton-based Human Motion Prediction. CVPR 2020 (oral)

Three body scales



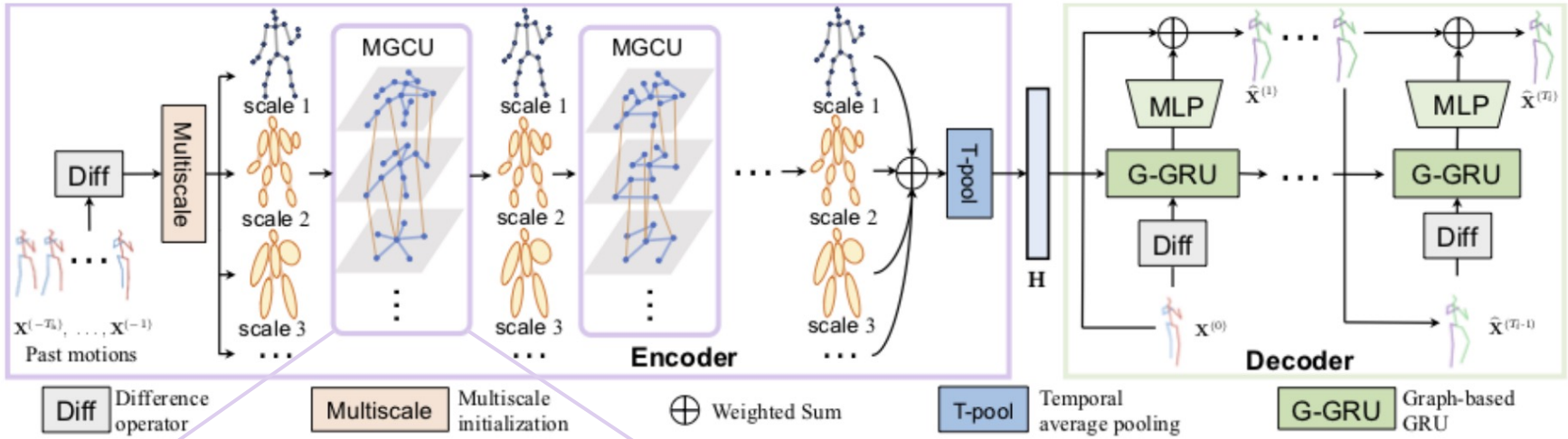
Capture relations across scales

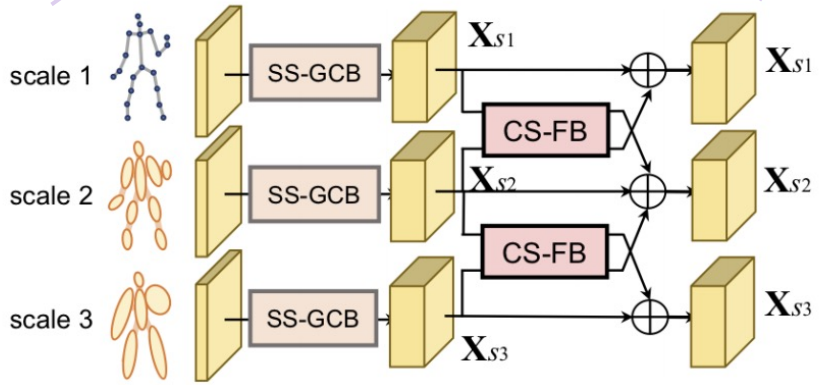


Examples: local relations Distant relations

Multiscale GCN model for Human Motion Prediction

Li, Chen, et al. Dynamic Multiscale Graph Neural Networks for 3D skeleton-based Human Motion Prediction. CVPR 2020 (oral)





single-scale graph convolution

cross-scale fusion blocks

encoder-decoder framework

- Encoder: multiscale graph computation
- Decoder: graph-based GRU

Multiscale GCN model for Human Motion Prediction

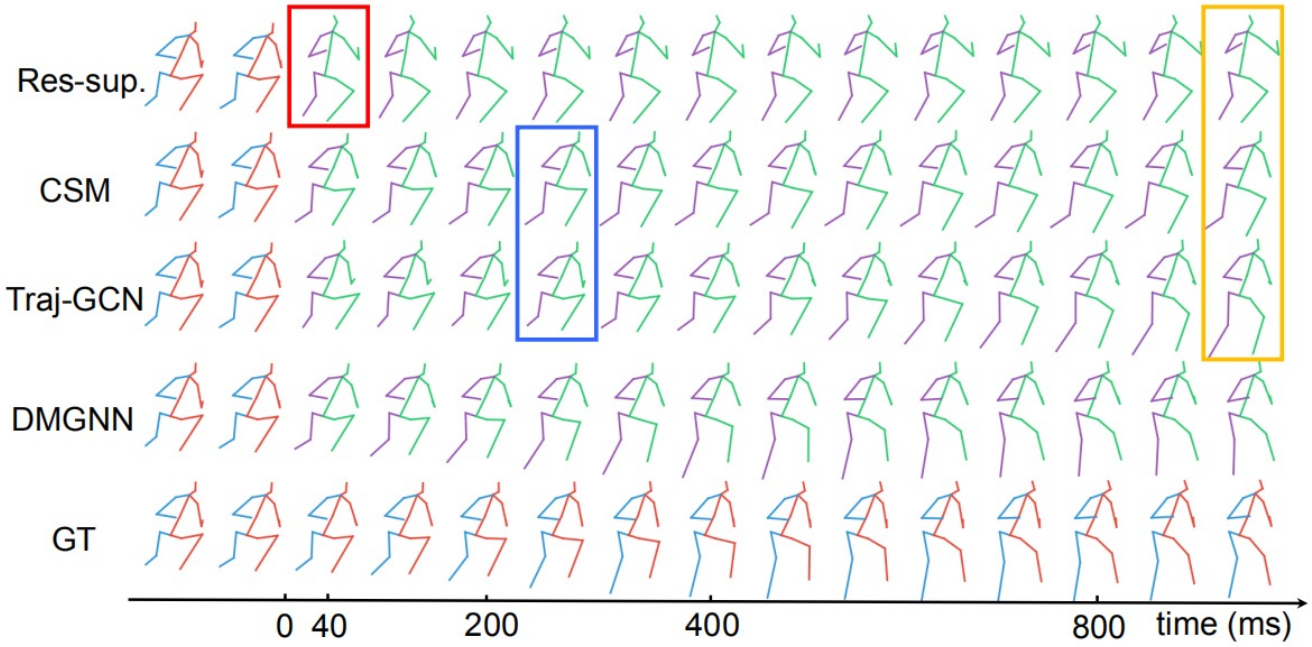
Li, Chen, et al. Dynamic Multiscale Graph Neural Networks for 3D skeleton-based Human Motion Prediction. CVPR 2020 (oral)

Large discontinuity between the last observed pose the first predicted one

Large errors after the 280th ms

Large posture errors in long-term

Qualitative comparison on the action 'Taking Photo' of H3.6M for both short and long-term prediction.



1000 ms with the frame interval of 80 ms.

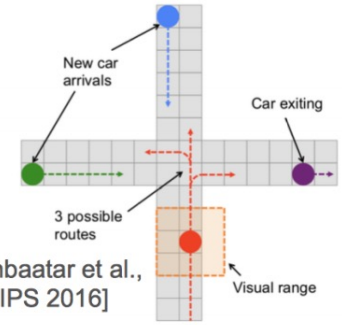
Conclusions

- **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)

Next lecture: Motion & tracking

Thank you!

sihengc@sjtu.edu.cn