Continual Graph Learning (CGL)





Xikun ZHANG Dongjin SONG Yushan JIANG Zijie PAN Dacheng TAO

THE UNIVERSITY OF SYDNEY

UCONN
UNIVERSITY OF CONNECTICUT

Agenda

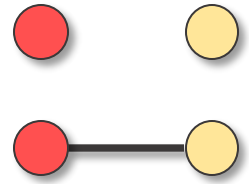
- Background on Graph Representation Learning (10 min)
- Motivation of Continual Graph Learning (10 min)
- Problem setup of CL and CGL (20 min)
- CL techniques & CGL techniques (40 min)
- Evaluation Metrics & CGL benchmarks (20 min)
- Future directions (10 min)

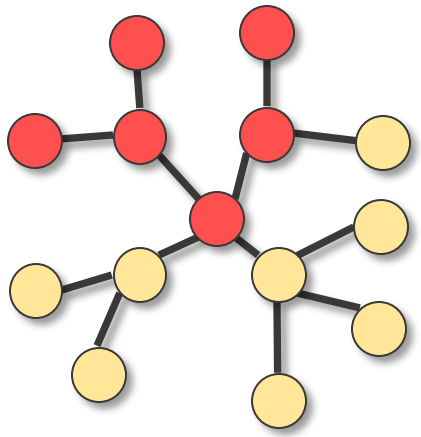
Agenda

- Background on Graph Representation Learning
- Motivation of Continual Graph Learning
- Problem setup of CL and CGL
- CL techniques & CGL techniques
- Evaluation Metrics & CGL benchmarks
- Future directions

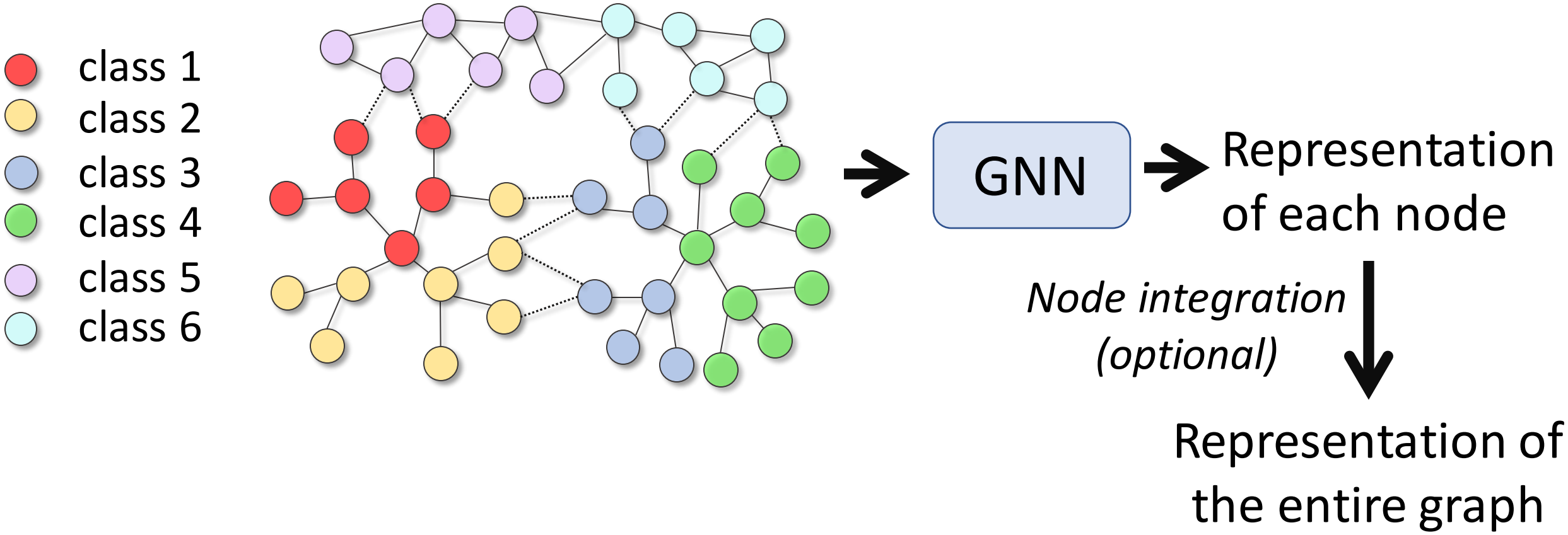
Graphs consist of

- Nodes (or vertices)
- Edges (or links) connecting nodes



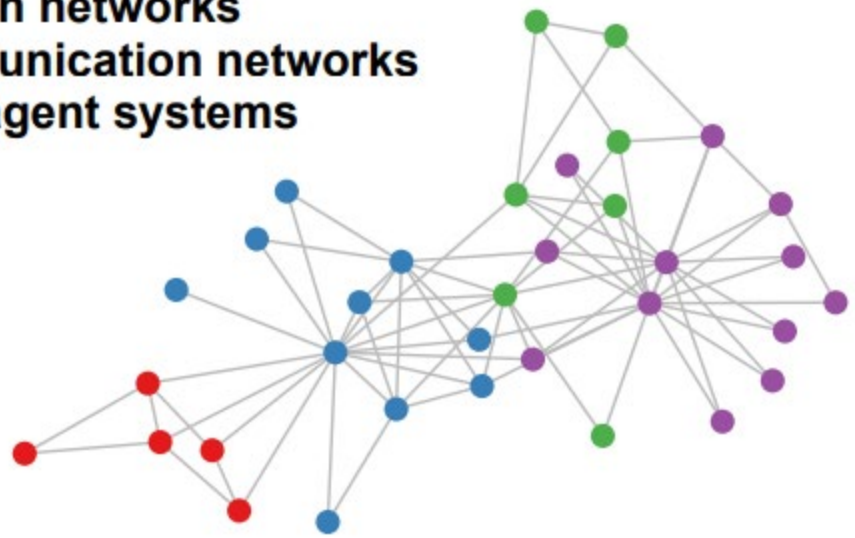


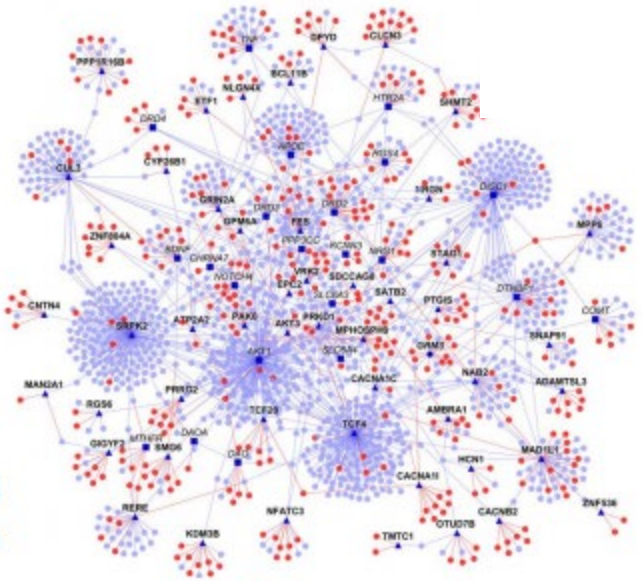
Node representation learning via Graph Neural Networks (GNNs)



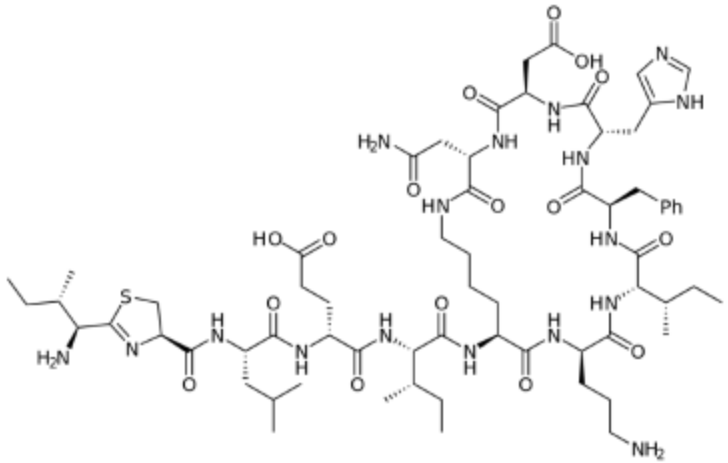
Graph data are pervasive in our world

Social networks
Citation networks
Communication networks
Multi-agent systems





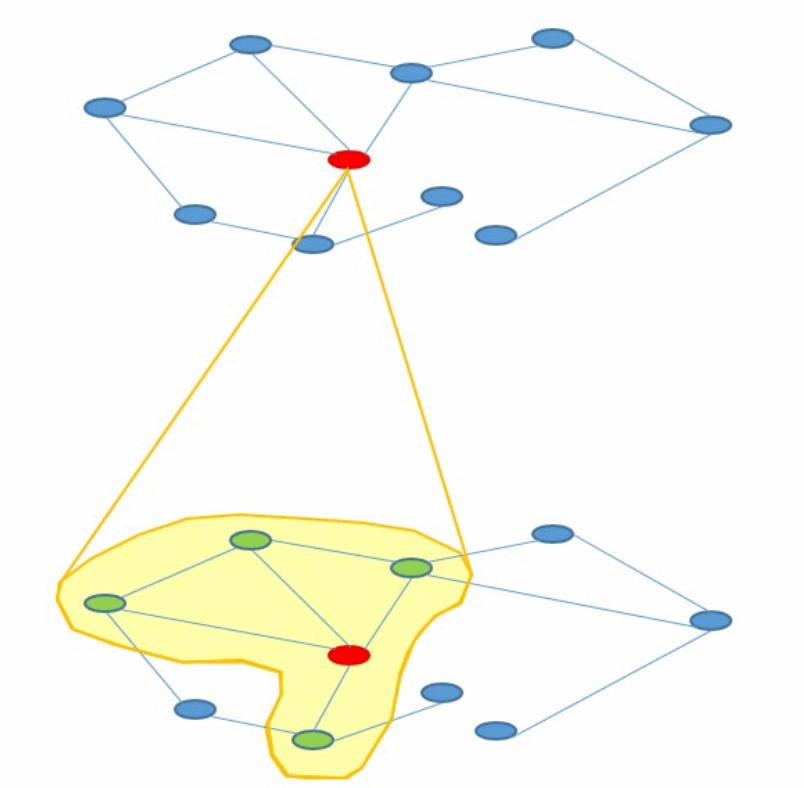
Protein interaction networks



Molecules

Left: https://dilinikarunarathna.medium.com/introduction-to-graph-convolutional-networks-gcn-2235ed69875d
Middle: https://www.genengnews.com/insights/protein-protein-interactions-get-a-new-groove-on/
Right: https://sl.wikipedia.org/wiki/Bacitracin

Message Passing Neural Network (MPNN)



Message function

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw})$$

Neighbours of *v*

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1})$$

Node update function

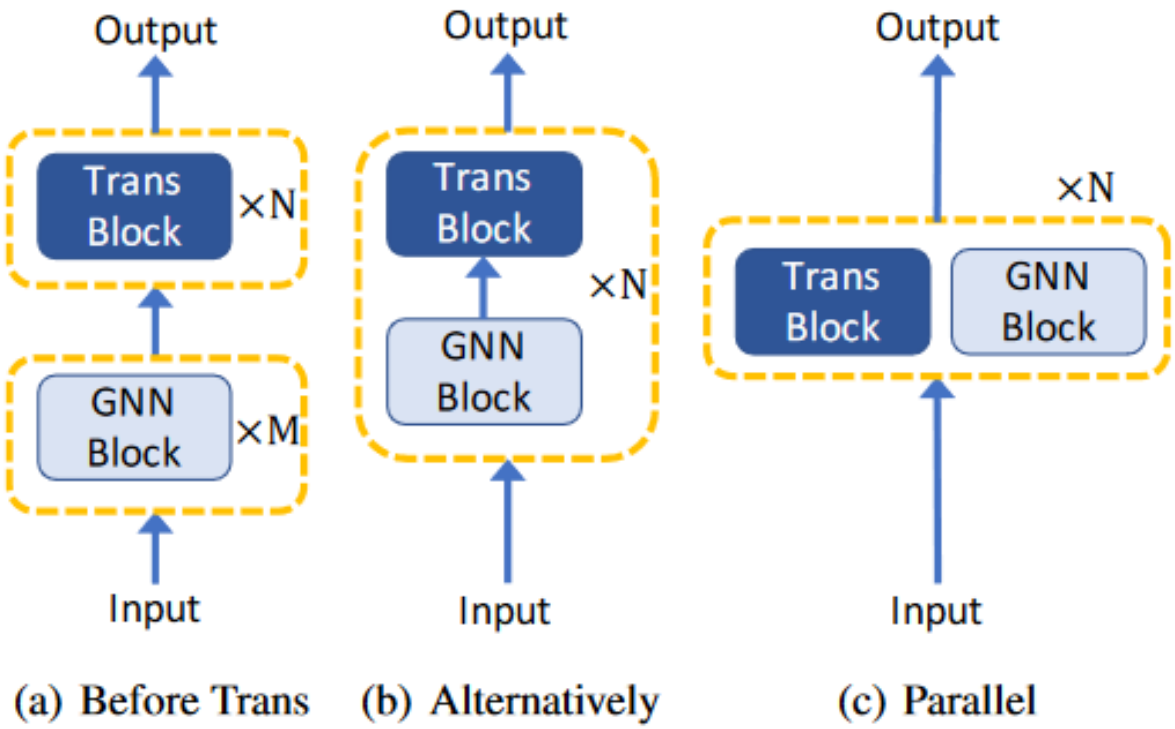
$$\hat{y} = R(\{h_v^T \mid v \in G\}).$$

Readout function

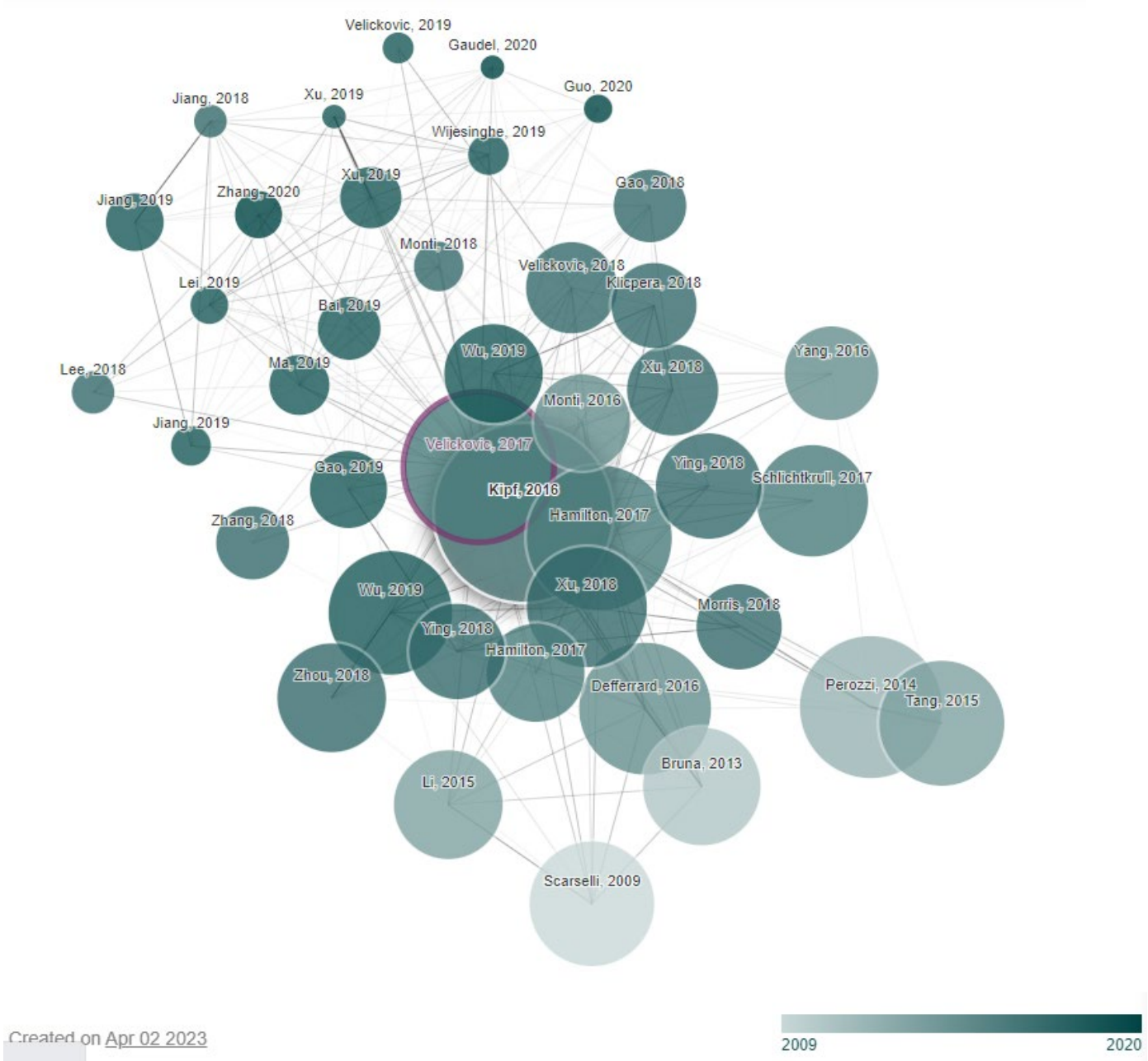
Gilmer, Justin, et al. "Neural message passing for quantum chemistry." International conference on machine learning. PMLR, 2017.

Graph transformers

- GNNs as Auxiliary Modules in Transformer
- Improved Positional Embeddings from Graphs
- Improved Attention Matrices from Graphs



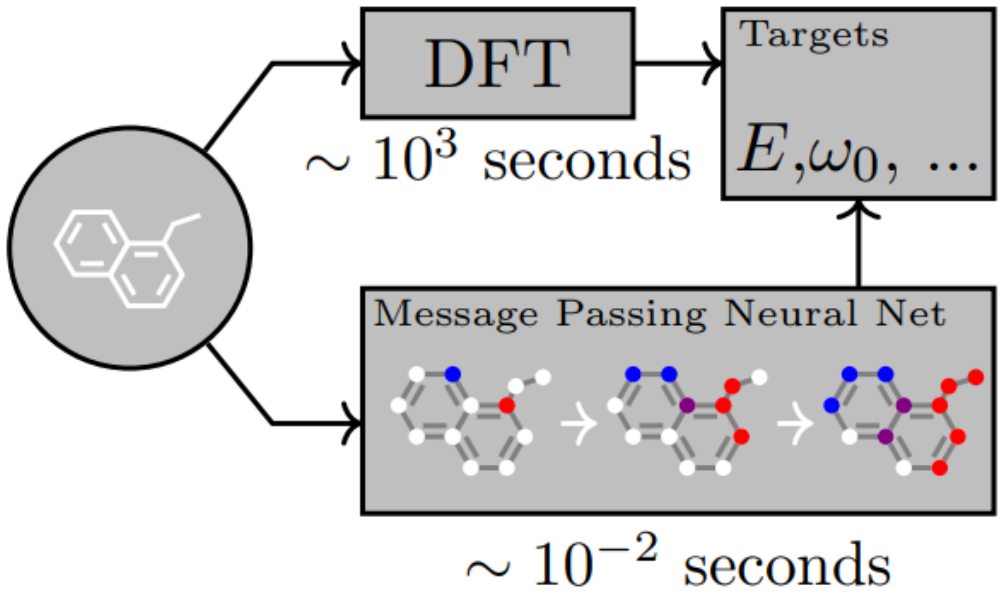
Min, Erxue, et al. "Transformer for graphs: An overview from architecture perspective." arXiv preprint arXiv:2202.08455 (2022).



Popular node-level task: Node classification on Citation network

https://www.connectedpapers.com/

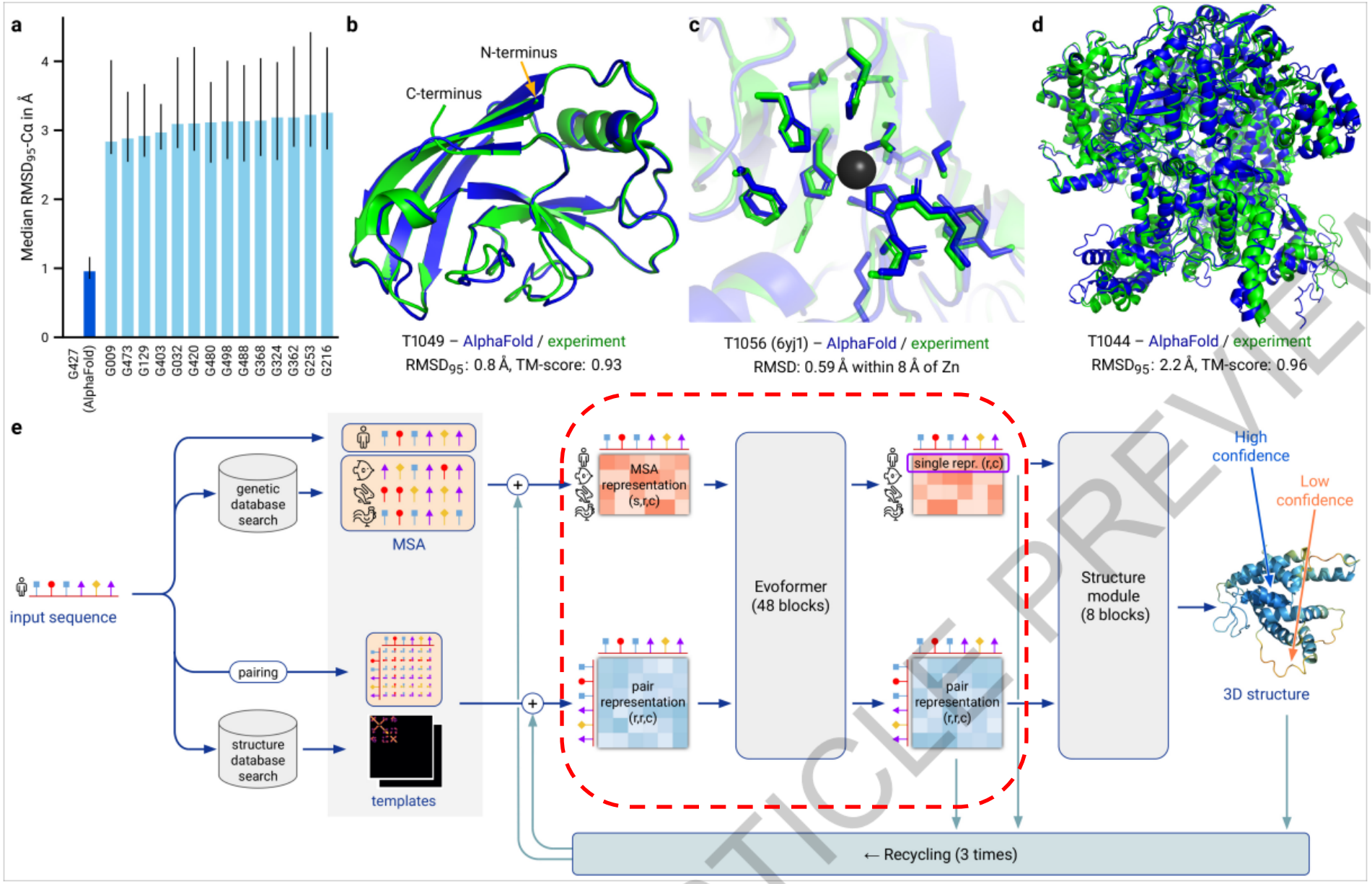
Message Passing Neural Network (MPNN)



Popular graph-level task: Molecule property prediction

Gilmer, Justin, et al. "Neural message passing for quantum chemistry." International conference on machine learning. PMLR, 2017.

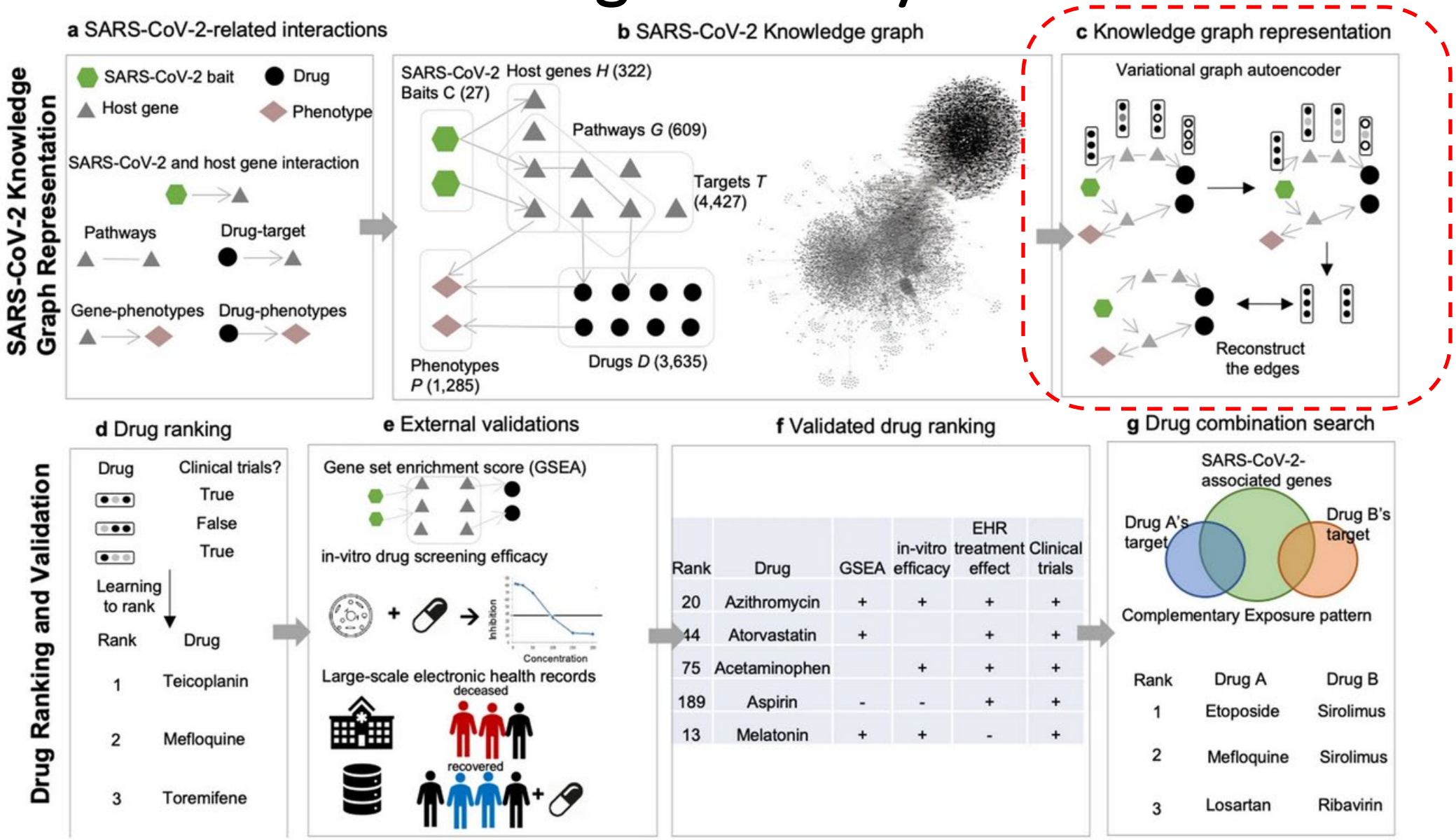
AlphaFold2 (Protein structure prediction)



Background on Graph Representation Learning

Jumper, John, et al. "Highly accurate protein structure prediction with AlphaFold." *Nature* 596.7873 (2021): 583-589.

Drug discovery



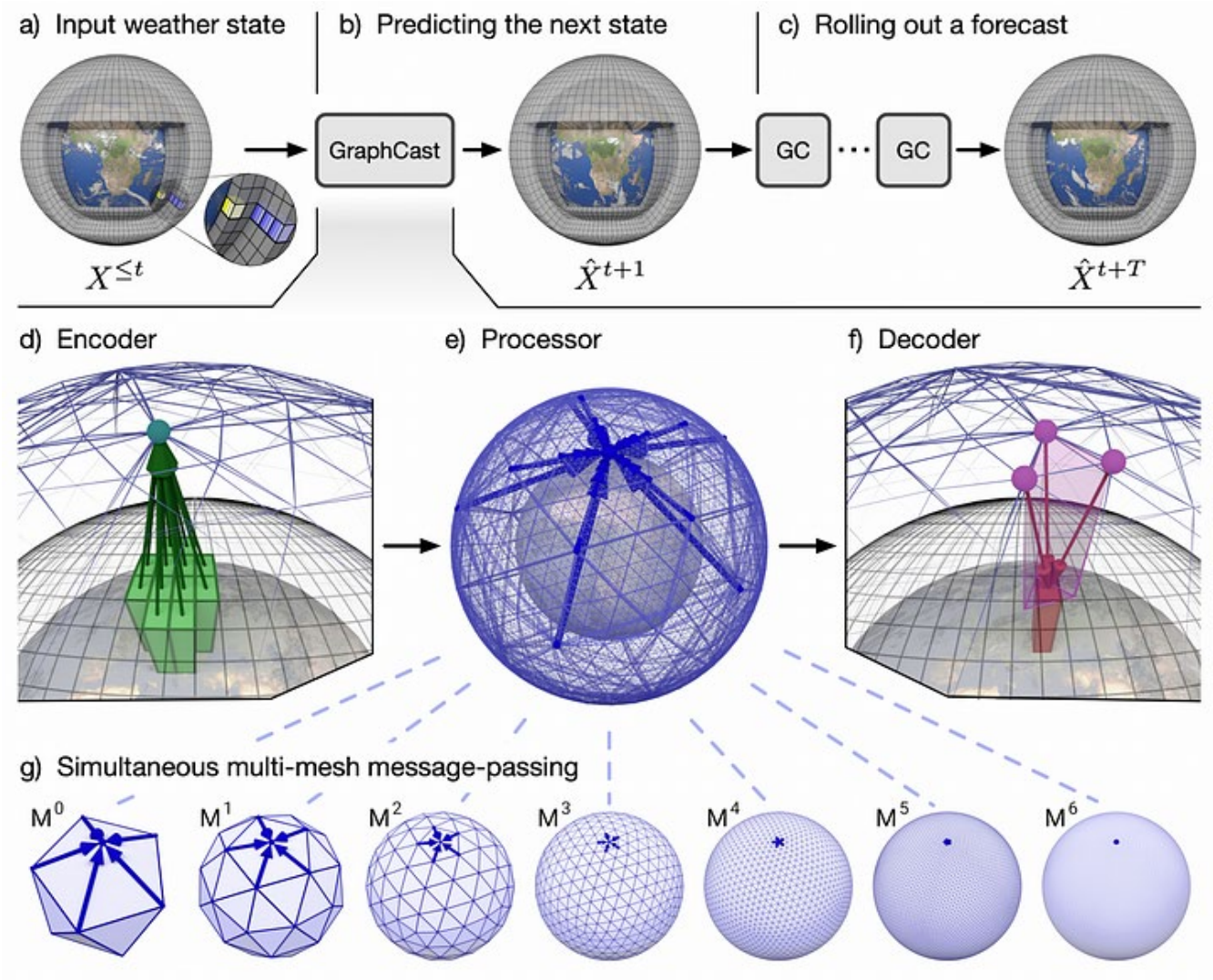
Rank	Drug	GSEA	in-vitro efficacy	EHR treatment effect	Clinical trials
20	Azithromycin	+	+	+	+
44	Atorvastatin	+		+	+
75	Acetaminophen		+	+	+
189	Aspirin	-	-	+	+
13	Melatonin	+	+	-	+

Rank	Drug A	Drug B
1	Etoposide	Sirolimus
2	Mefloquine	Sirolimus
3	Losartan	Ribavirin

Background on Graph Representation Learning

Hsieh, Kanglin, et al. "Drug repurposing for COVID-19 using graph neural network and harmonizing multiple evidence." Scientific reports 11.1 (2021): 23179.

GraphCast (Weather forecasting)



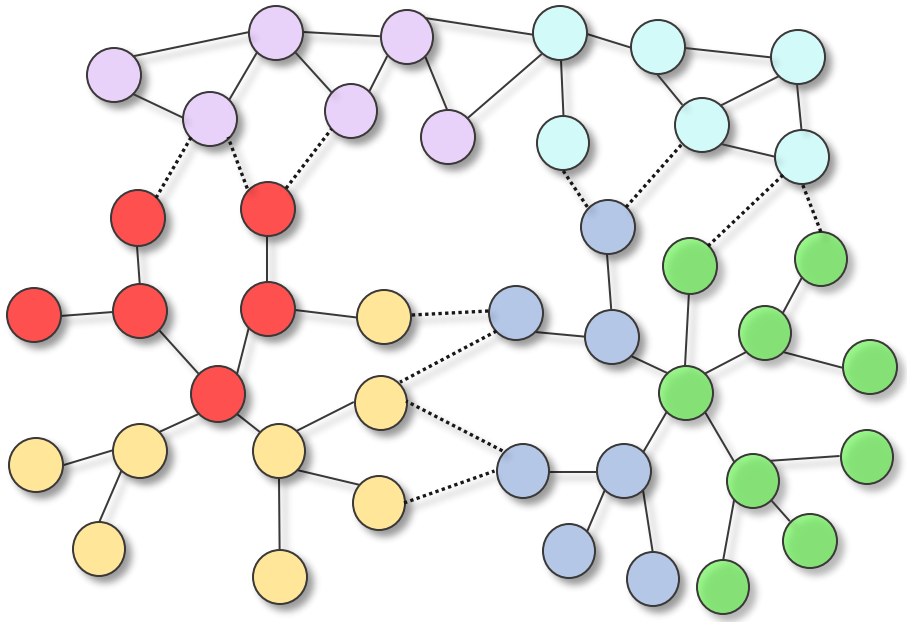
Lam, Remi, et al. "GraphCast: Learning skillful medium-range global weather forecasting." arXiv preprint arXiv:2212.12794 (2022).

Agenda

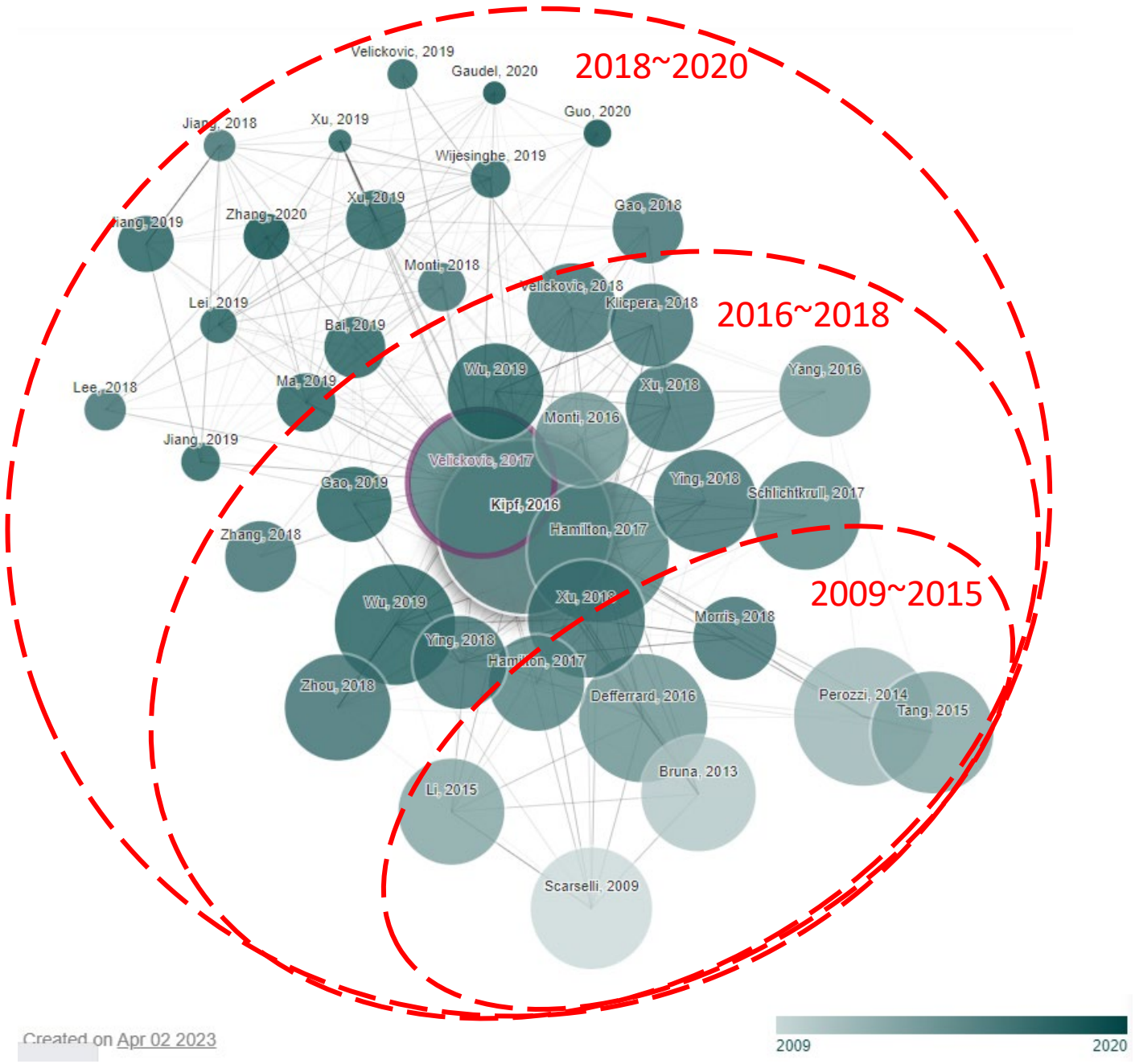
- Background on Graph Representation Learning
- Motivation of Continual Graph Learning
- Problem setup of CL and CGL
- CL techniques & CGL techniques
- Evaluation Metrics & CGL benchmarks
- Future directions

Learning on static graphs

- class 1
- class 2
- class 3
- class 4
- class 5
- class 6



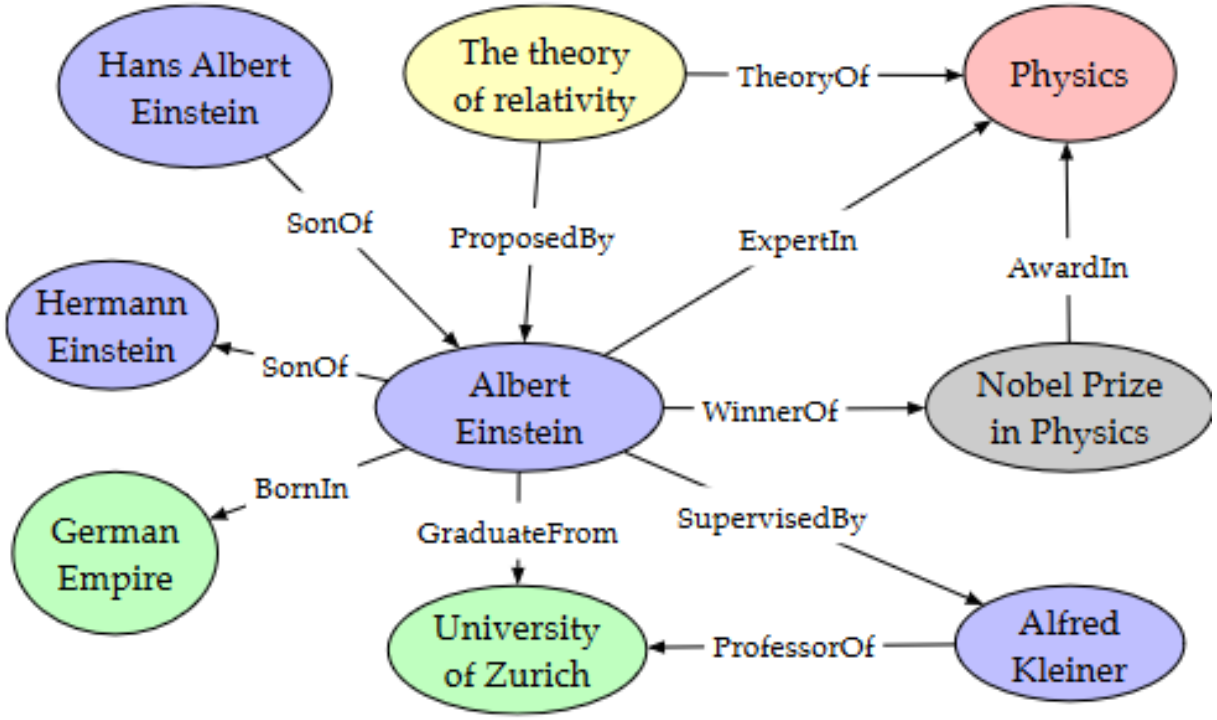
→ Model → Representation of each node



Citation network grows with time

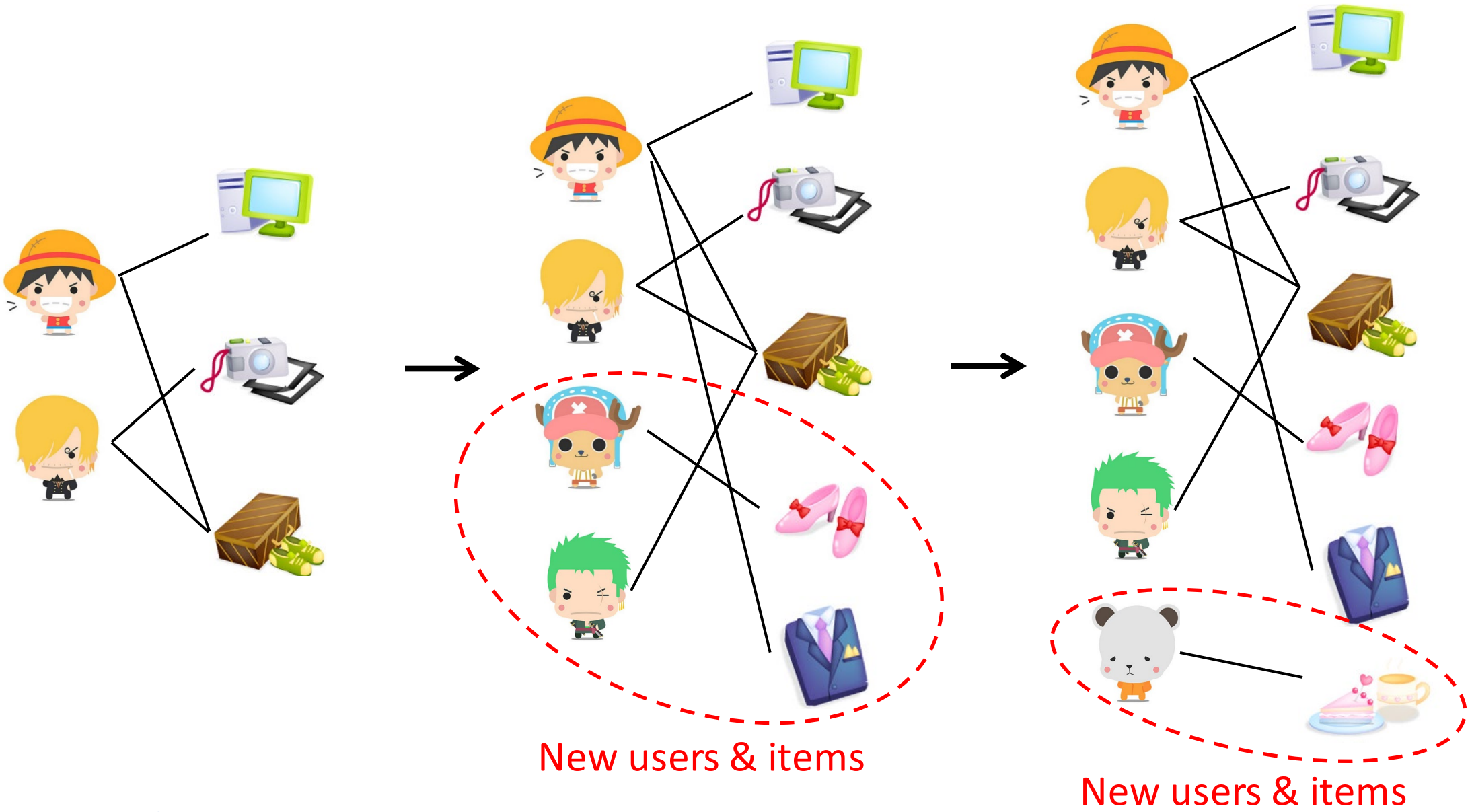
Knowledge graph grows with time

(Albert Einstein, **BornIn**, German Empire)
(Albert Einstein, **SonOf**, Hermann Einstein)
(Albert Einstein, **GraduateFrom**, University of Zurich)
(Albert Einstein, **WinnerOf**, Nobel Prize in Physics)
(Albert Einstein, **ExpertIn**, Physics)
(Nobel Prize in Physics, **AwardIn**, Physics)
(The theory of relativity, **TheoryOf**, Physics)
(Albert Einstein, **SupervisedBy**, Alfred Kleiner)
(Alfred Kleiner, **ProfessorOf**, University of Zurich)
(The theory of relativity, **ProposedBy**, Albert Einstein)
(Hans Albert Einstein, **SonOf**, Albert Einstein)

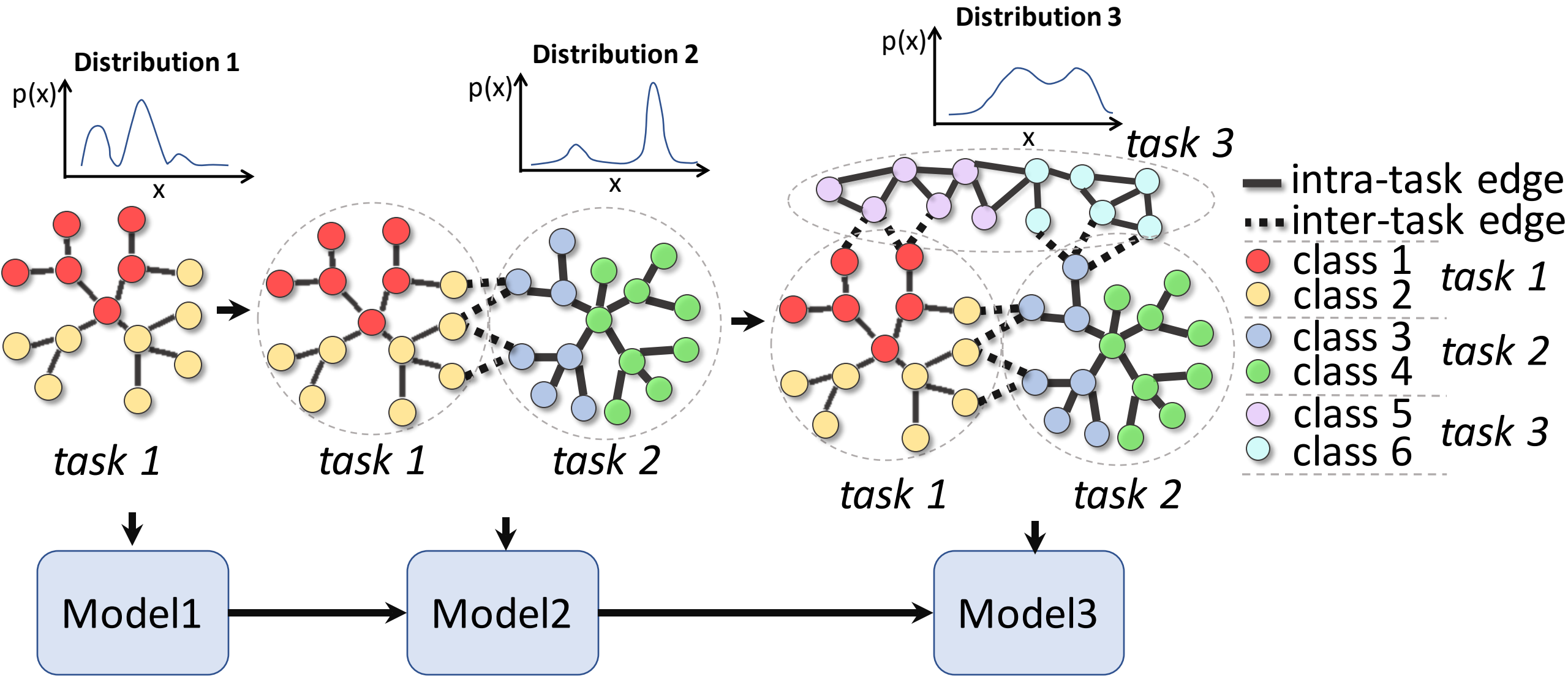


Ji, Shaoxiong, et al. "A survey on knowledge graphs: Representation, acquisition, and applications." IEEE transactions on neural networks and learning systems 33.2 (2021): 494-514

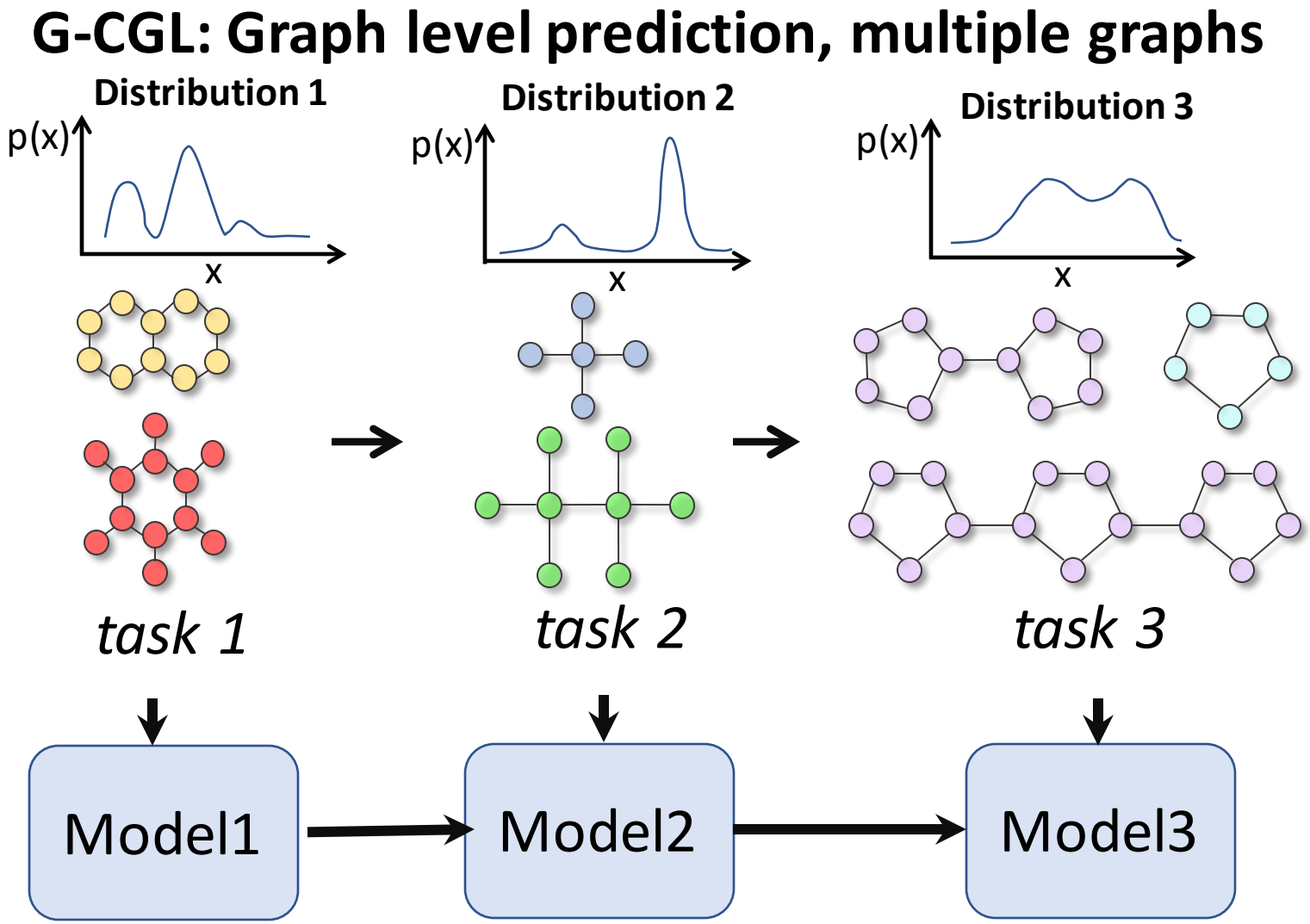
Recommender system graph grows with time



Model has to adapt

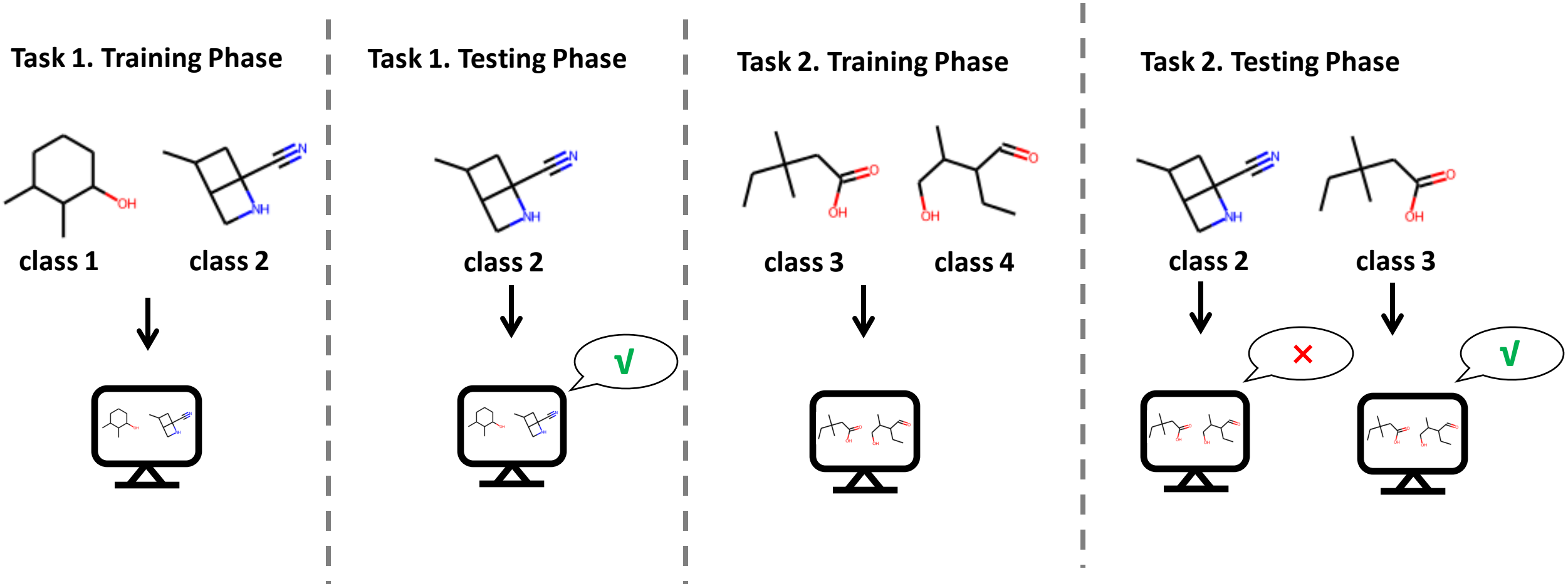


Realistic continual learning scenario

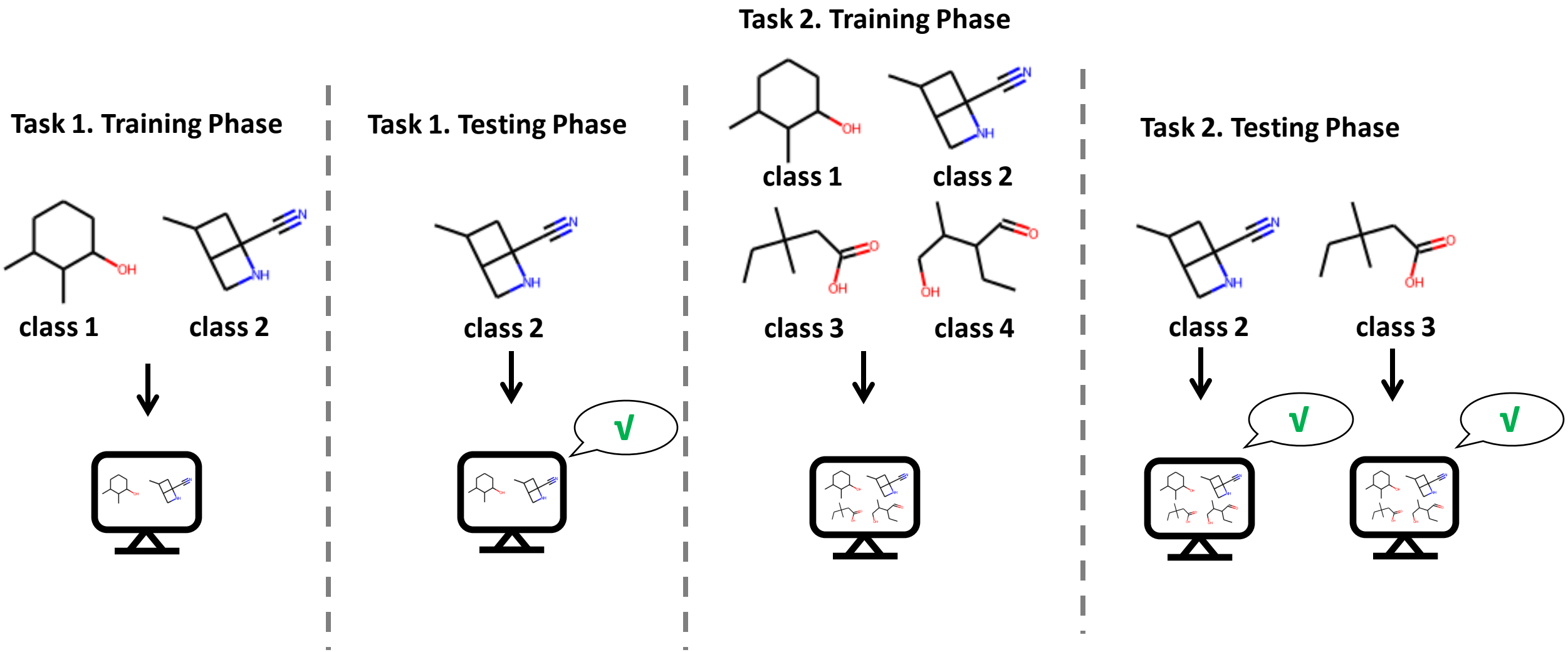


Problem setup of CGL

Catastrophic forgetting



Can we just retrain over all data?



Retraining is computationally impractical

Scale	Name	#Nodes	#Edges*
Medium	ogbn-products	2,449,029	61,859,140
Medium	ogbn-proteins	132,534	39,561,252
Small	ogbn-arxiv	169,343	1,166,243
Large	ogbn-papers100M	111,059,956	1,615,685,872
Medium	ogbn-mag	1,939,743	21,111,007

Scale	Name	#Nodes	#Edges*
Medium	ogbl-ppa	576,289	30,326,273
Small	ogbl-collab	235,868	1,285,465
Small	ogbl-ddi	4,267	1,334,889
Medium	ogbl-citation2	2,927,963	30,561,187
Medium	ogbl-wikikg2	2,500,604	17,137,181
Small	ogbl-biokg	93,773	5,088,434
Medium	ogbl-vessel*	3,538,495	5,345,897

Scale	Name	#Graphs	#Nodes per graph	#Edges per graph*
Small	ogbg-molhiv	41,127	25.5	27.5
Medium	ogbg-molpcba	437,929	26.0	28.1
Medium	ogbg-ppa	158,100	243.4	2,266.1
Medium	ogbg-code2	452,741	125.2	124.2

Task category	Name	#Graphs	#Total nodes	#Total edges
Node-level	MAG240M	1	244,160,499	1,728,364,232
Link-level	WikiKG90Mv2	1	91,230,610	601,062,811
Graph-level	PCQM4Mv2	3,746,619	52,970,652	54,546,813

Hu, Weihua, et al. "Open graph benchmark: Datasets for machine learning on graphs." *Advances in neural information processing systems* 33 (2020): 22118-22133.

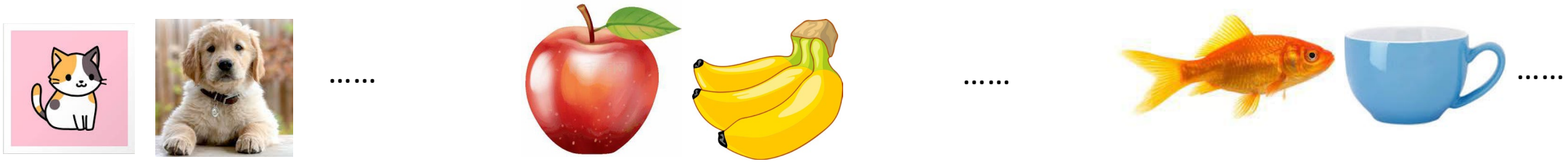
Objectives

- Enough plasticity to adapt to new tasks (including new topologies)
- Minimal forgetting on previous tasks (topologies)
- If possible, positive cross task transfer

Agenda

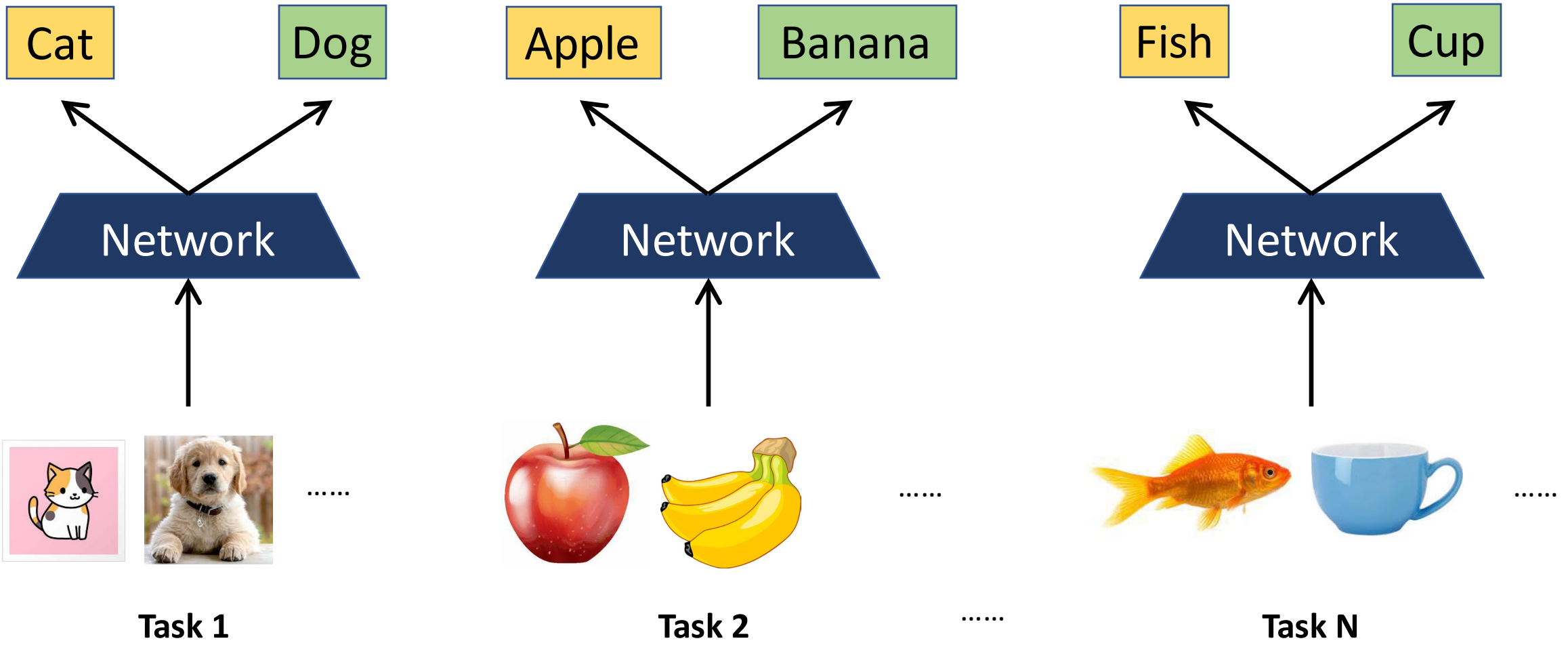
- Background on Graph Representation Learning
- Motivation of Continual Graph Learning
- Problem setup of CL and CGL
- CL techniques & CGL techniques
- Evaluation Metrics & CGL benchmarks
- Future directions

Basic concept: tasks

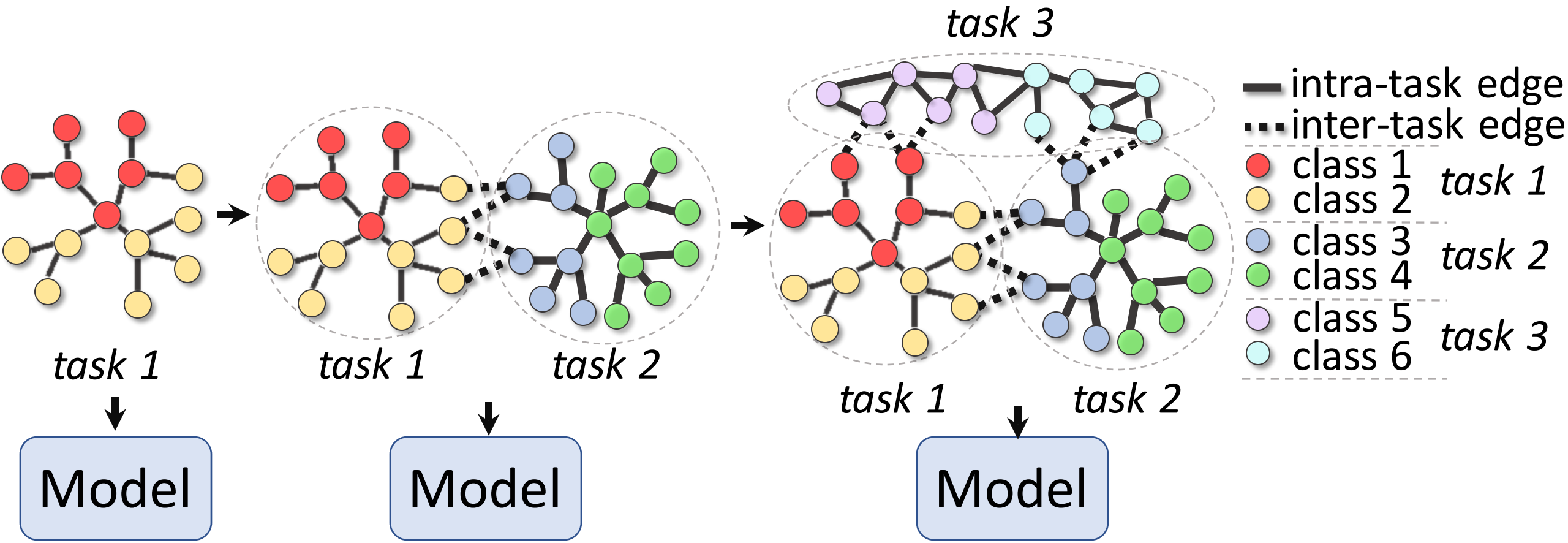


Task 1 **Task 2** **Task N**

Commonly adopted task constructions



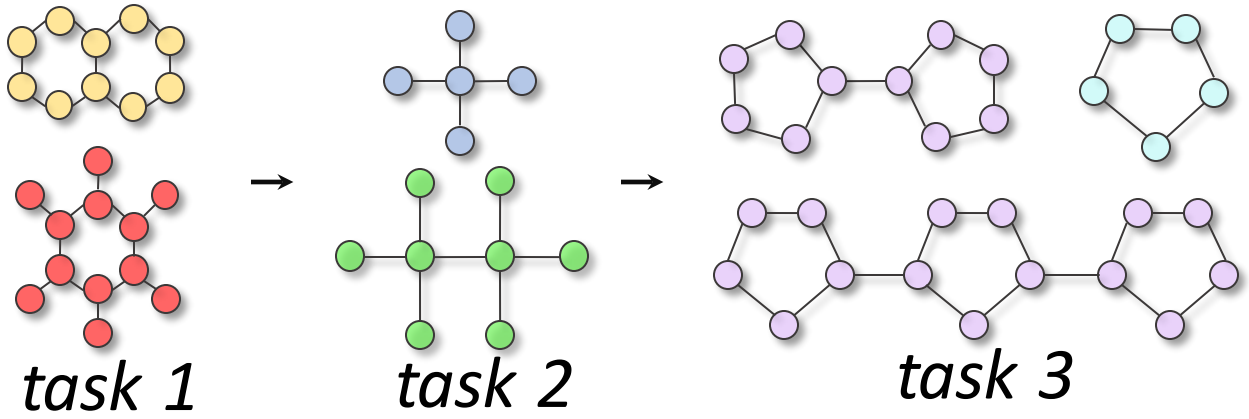
Realistic continual learning scenario



Problem setup of CGL

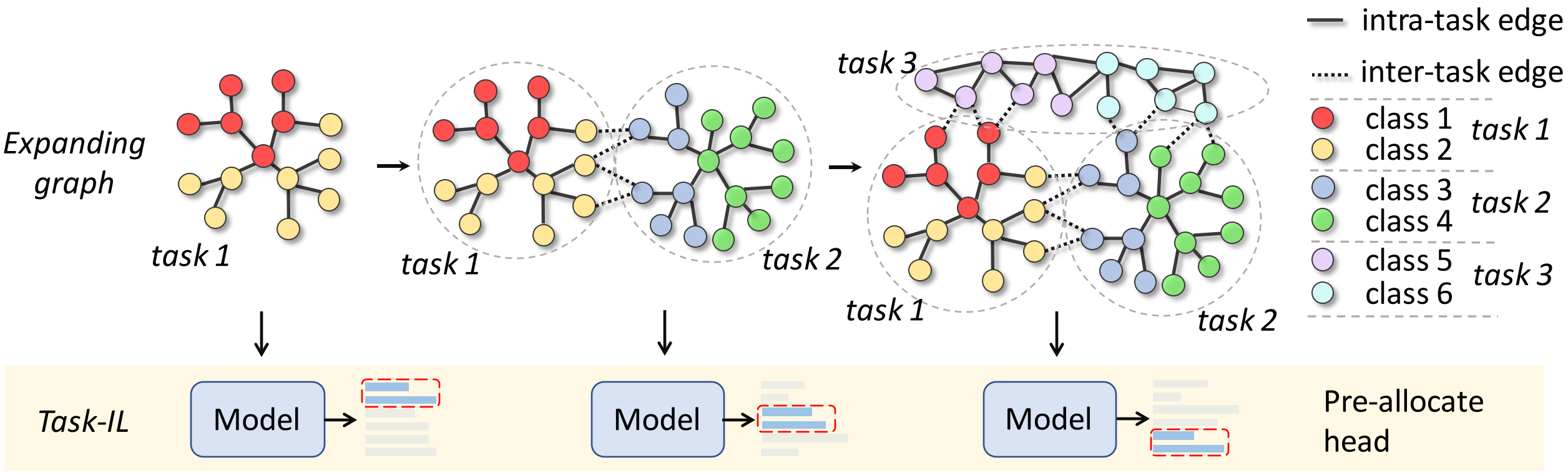
Realistic continual learning scenario

G-CGL: Graph level prediction, multiple graphs

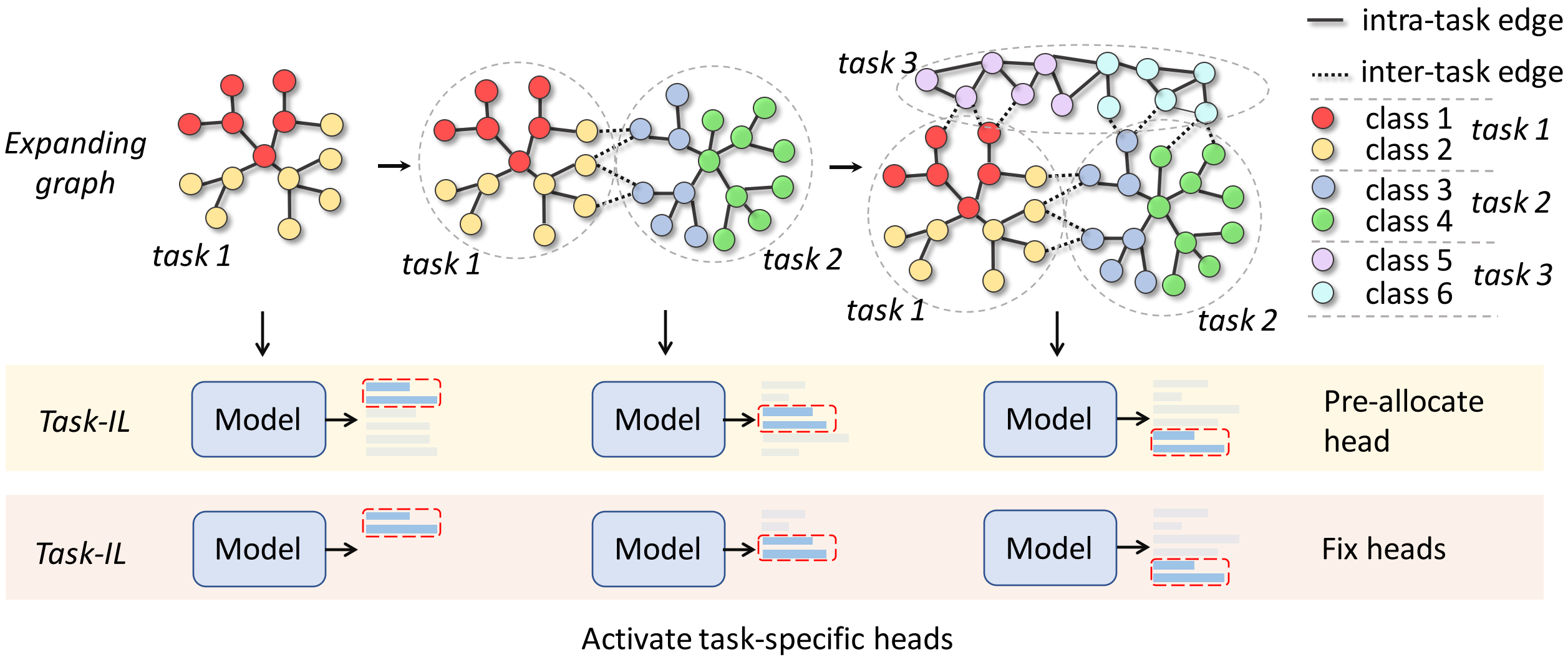


- Node- or graph-level tasks?

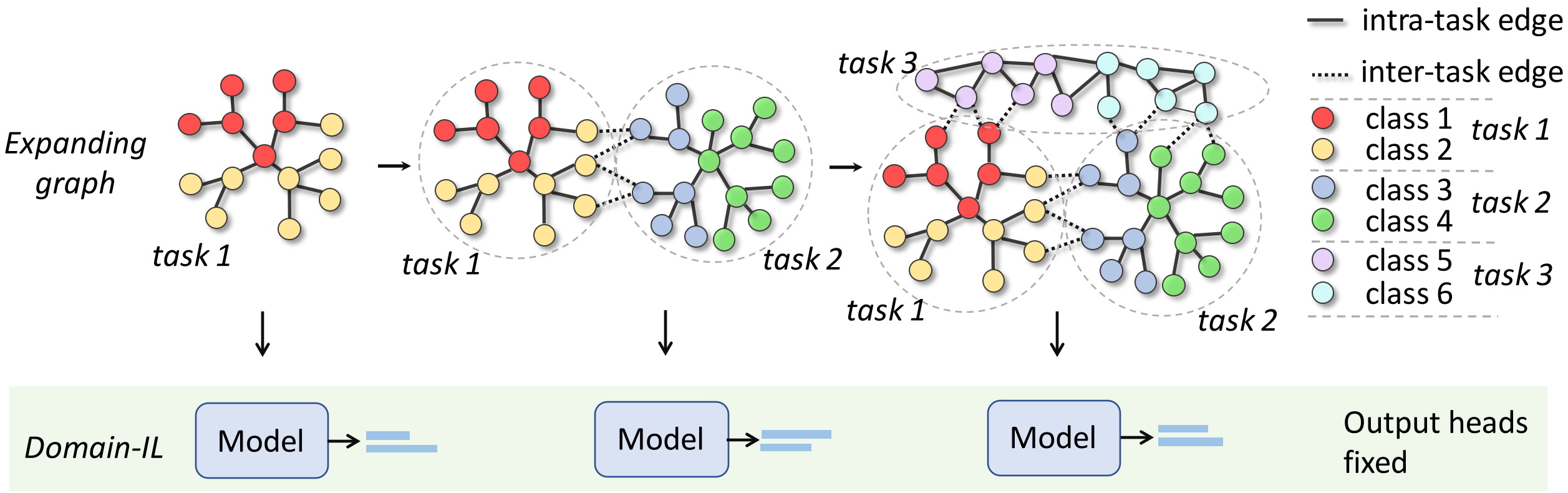
Task-IL & Node-level tasks



Task-IL & Node-level tasks

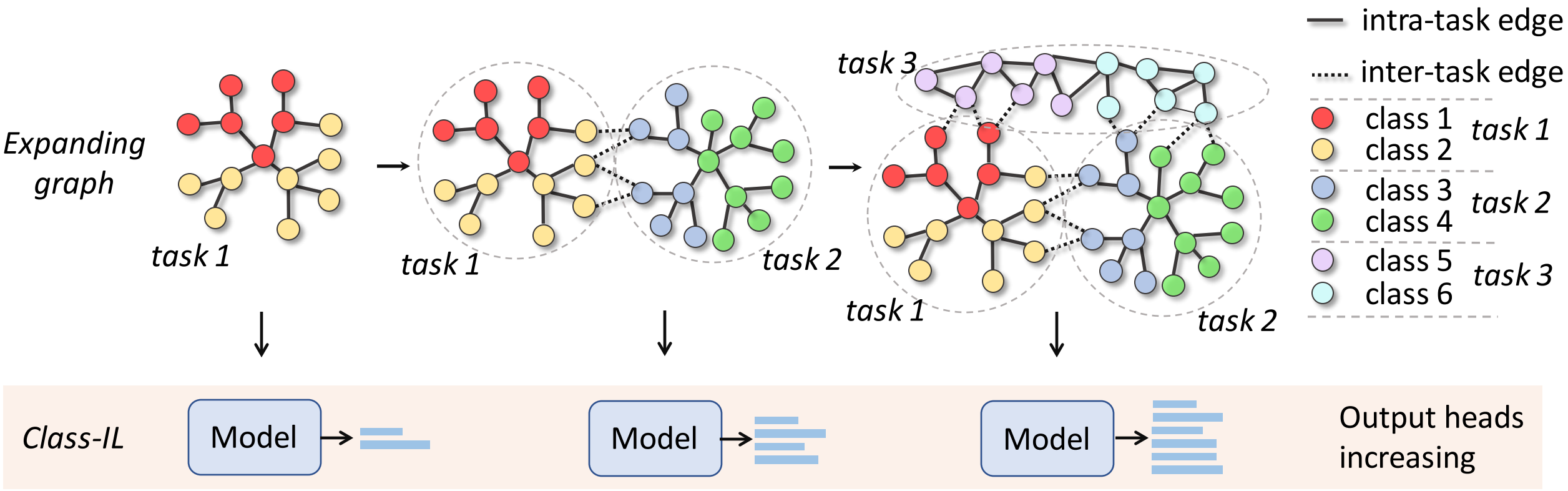


Domain-IL & Node-level tasks



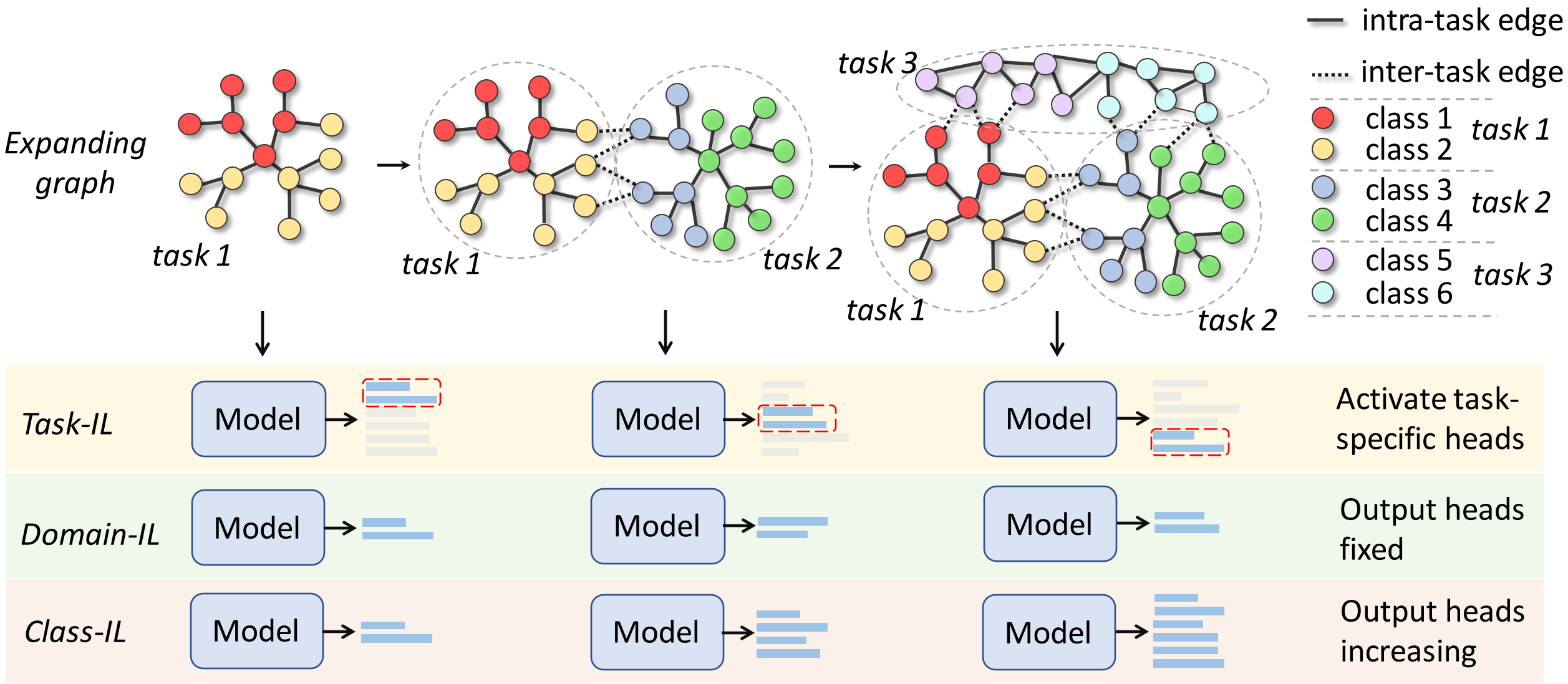
Problem setup of CGL

Class-IL & Node-level tasks



Problem setup of CGL

Summary of three incremental scenarios



Difficulty: Plasticity-stability dilemma

- Plasticity: adapting to new tasks
- Stability: maintaining performance on old tasks
- Catastrophic forgetting: models adapting well to new tasks may forget old tasks severely

Agenda

- Background on Graph Representation Learning
- Motivation of Continual Graph Learning
- Problem setup of CL and CGL
- CL techniques & CGL techniques
- Evaluation Metrics & CGL benchmarks
- Future directions

Rough Categories

- **Regularization:** Penalize changes to the model via regularizations
- **Parameter-isolation:** Separate parameters for new and old tasks (partially or entirely)
- **Memory-replay:** Replay old task data to the model when learning new ones

Rough Categories

- **Regularization:** Penalize changes to the model via regularizations
- **Parameter-isolation:** Separate parameters for new and old tasks (partially or entirely)
- **Memory-replay:** Replay old task data to the model when learning new ones

Elastic weight consolidation (EWC, regularization based)

Basic loss for learning
the currently given task

$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i(\theta_i - \theta^*_{A,i})^2$$

Regularization term for
minimizing the
forgetting problem

$\boldsymbol{\lambda}$:balances the contribution of the old tasks

$\boldsymbol{F_i}$:Parameter importance to task A
(measured by Fisher information)

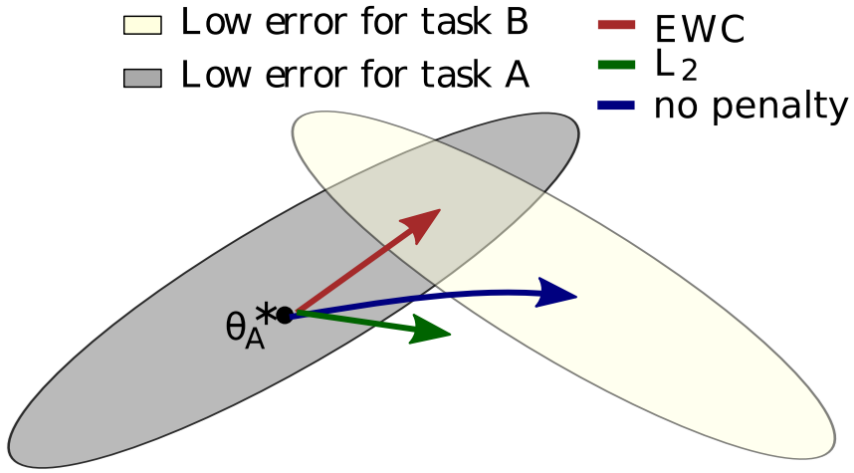
Kirkpatrick, James, et al. "Overcoming catastrophic forgetting in neural networks." Proceedings of the national academy of sciences 114.13 (2017): 3521-3526.

Elastic weight consolidation (EWC, regularization based)

Basic loss for learning
the currently given task

$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i(\theta_i - \theta^*_{A,i})^2$$

Regularization term for
minimizing the
forgetting problem



$\boldsymbol{\lambda}$:balances the contribution of the old tasks

$\boldsymbol{F_i}$:Parameter importance to task A
(measured by Fisher information)

Kirkpatrick, James, et al. "Overcoming catastrophic forgetting in neural networks." Proceedings of the national academy of sciences 114.13 (2017): 3521-3526.

Elastic weight consolidation (EWC, regularization based)

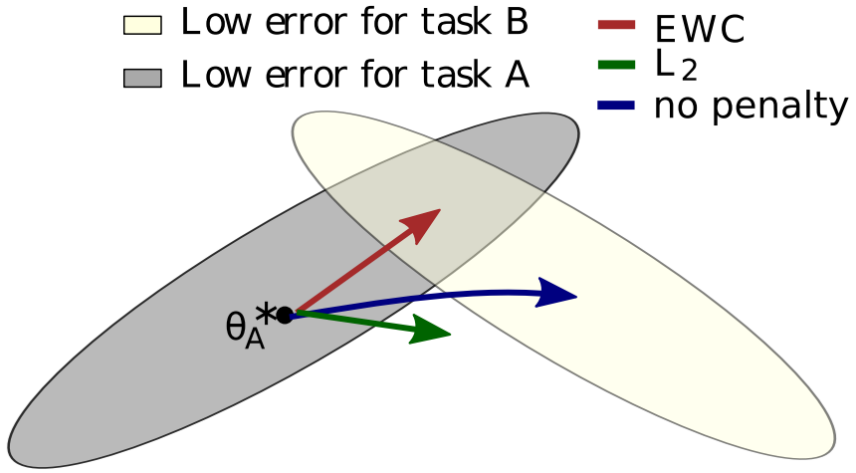
Basic loss for learning
the currently given task

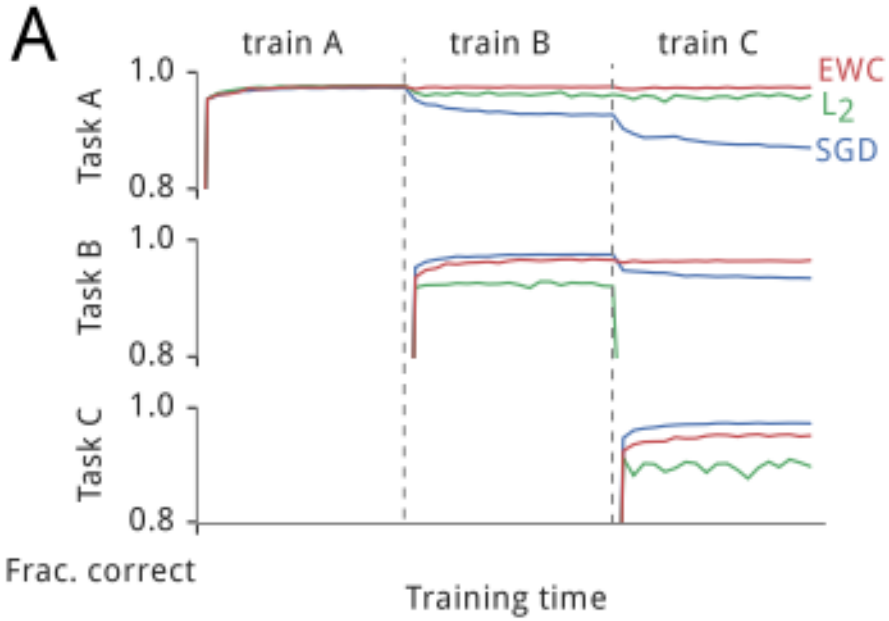
$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i(\theta_i - \theta^*_{A,i})^2$$

Regularization term for
minimizing the
forgetting problem

$\boldsymbol{\lambda}$:balances the contribution of the old tasks

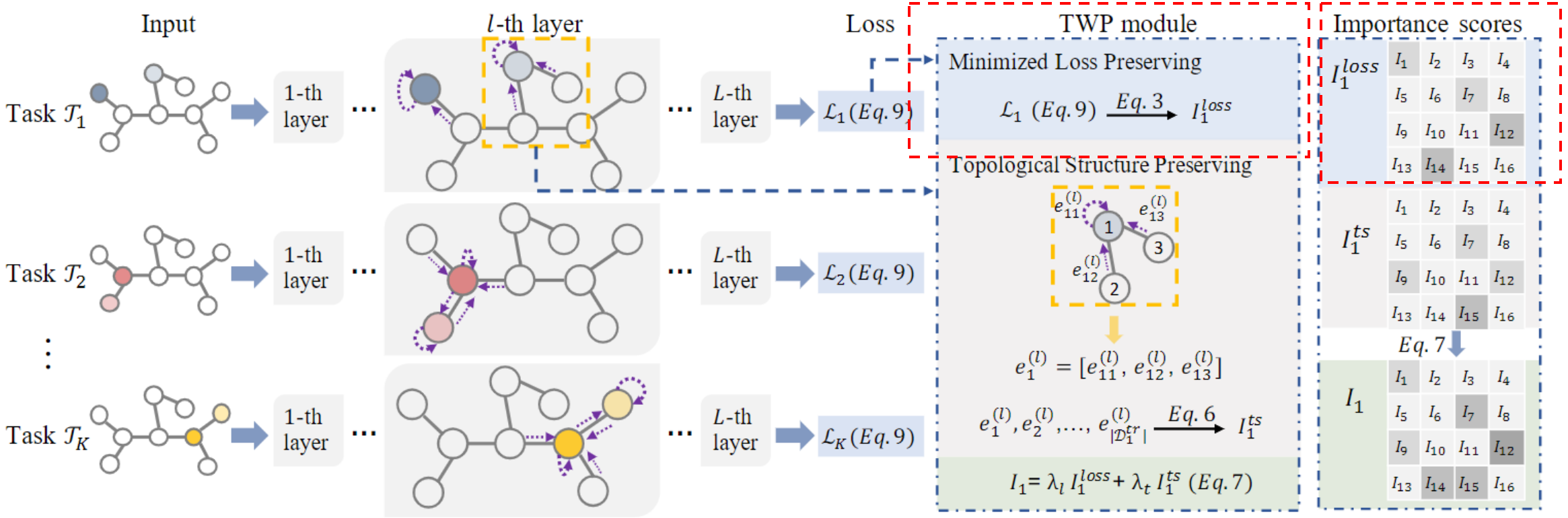
$\boldsymbol{F_i}$:Parameter importance to task A
(measured by Fisher information)





Kirkpatrick, James, et al. "Overcoming catastrophic forgetting in neural networks." Proceedings of the national academy of sciences 114.13 (2017): 3521-3526.

Topology-aware Weight Preserving (TWP, regularization based)



Minimized Loss Preserving

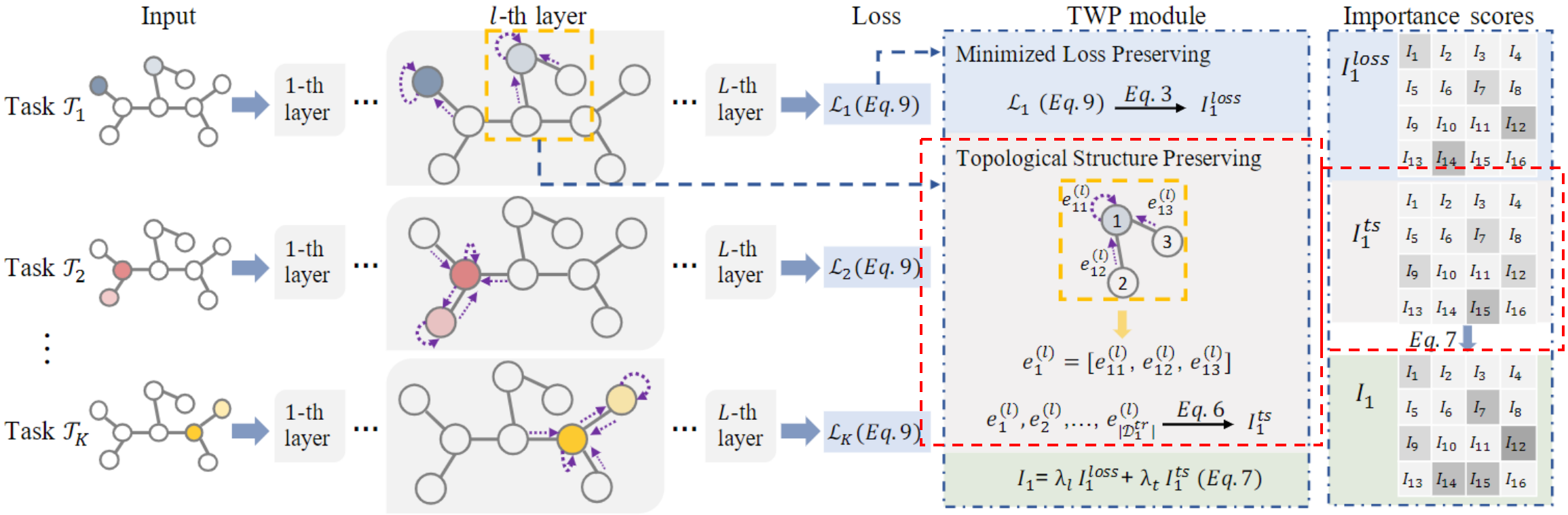
$$\mathcal{L}(X_k^{tr}; W + \Delta W) - \mathcal{L}(X_k^{tr}; W) \approx \sum_m f_m(X_k^{tr}) \Delta \boldsymbol{w}_m$$

$$f_m(X_k^{tr}) = \frac{\partial \mathcal{L}}{\partial \boldsymbol{w}_m} \qquad \boxed{I_k^{loss} = [\|f_m(X_k^{tr})\|]}$$

importance score (loss)

Liu, Huihui, Yiding Yang, and Xinchao Wang. "Overcoming catastrophic forgetting in graph neural networks." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35. No. 10. 2021.

Topology-aware Weight Preserving (TWP, regularization based)



Topological Structure Preserving

$$e_{ij}^{(l)} = a(\boldsymbol{H}_{i,j}^{(l-1)}; W^{(l)})$$

(for GAT)

Embedding of i and j

$$e_{ij}^{(l)} = (\boldsymbol{h}_i^{(l-1)} W^{(l)})^{\mathrm{T}} \tanh(\boldsymbol{h}_j^{(l-1)} W^{(l)})$$

(for any GNN)

$$g_m(\boldsymbol{H}^{(l-1)}) = \frac{\partial \left(\left| [e_1^{(l)}, \ldots, e_{|\mathcal{D}_k^{tr}|}^{(l)}] \right|_2^2 \right)}{\partial \boldsymbol{w}_m}$$

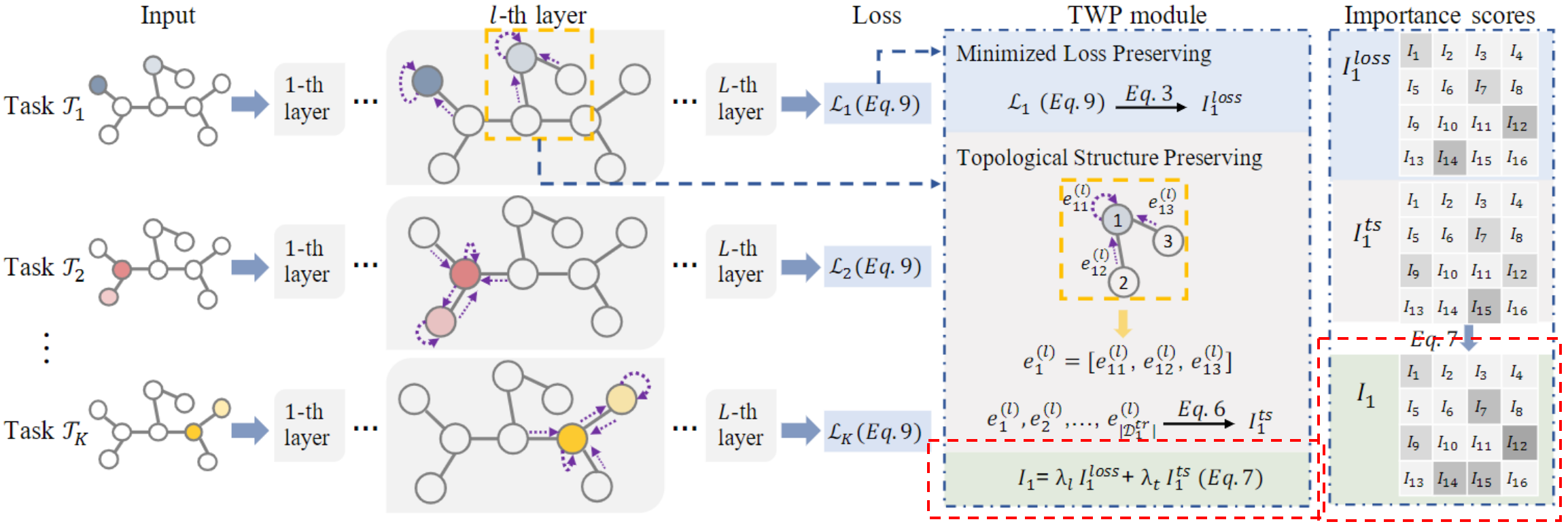
$$I_k^{ts} = [\|g_m(\boldsymbol{H}_k^{(l-1)})\|]$$

importance score (topology)

CGL techniques

Liu, Huihui, Yiding Yang, and Xinchao Wang. "Overcoming catastrophic forgetting in graph neural networks." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35. No. 10. 2021.

Topology-aware Weight Preserving (TWP, regularization based)



$$I_k = \lambda_l I_k^{loss} + \lambda_t I_k^{ts}$$

importance score (total)

$$\mathcal{L}'_{k+1}(W) = \mathcal{L}^{new}_{k+1}(W) + \sum_{n=1}^{k} I_n \otimes (W - W_n^*)^2$$

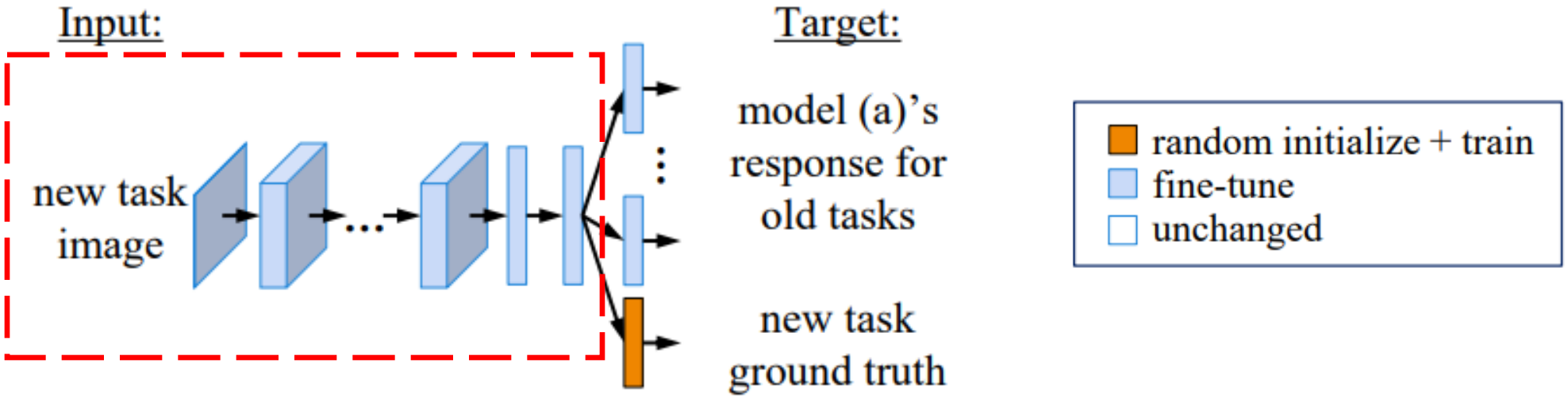
regularization

$$\mathcal{L}_{k+1}(W) = \mathcal{L}'_{k+1}(W) + \beta \left\| I_{k+1} \right\|_1$$

CGL techniques

Liu, Huihui, Yiding Yang, and Xinchao Wang. "Overcoming catastrophic forgetting in graph neural networks." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35. No. 10. 2021.

Learning without forgetting (LwF, regularization based)



Start with:

θ_s: shared parameters

θ_o: task specific parameters for each old task

X_n, Y_n: training data and ground truth on the new task

Initialize:

$Y_o \leftarrow \text{CNN}(X_n, \theta_s, \theta_o)$ // *compute output of old tasks for new data*

$\theta_n \leftarrow \text{RANDINIT}(|\theta_n|)$ // *randomly initialize new parameters*

Train:

Define $\hat{Y}_o \equiv \text{CNN}(X_n, \hat{\theta}_s, \hat{\theta}_o)$ // *old task output*

Define $\hat{Y}_n \equiv \text{CNN}(X_n, \hat{\theta}_s, \hat{\theta}_n)$ // *new task output*

$$\theta_s^*,\ \theta_o^*,\ \theta_n^* \leftarrow \underset{\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n}{\text{argmin}} \left(\lambda_o \mathcal{L}_{old}(Y_o, \hat{Y}_o) + \mathcal{L}_{new}(Y_n, \hat{Y}_n) + \mathcal{R}(\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n) \right)$$

Representative CL techniques

Li, Zhizhong, and Derek Hoiem. "Learning without forgetting." IEEE transactions on pattern analysis and machine intelligence 40.12 (2017): 2935-2947.

Self-Supervised Continual Graph Learning in Adaptive Riemannian Spaces (RieGrace, Regularization based)

RieGrace

Intra-distillation (self-supervised part): maximize the agreement between the final outputs and embeddings from shallow layer

$$\mathcal{J}(\mathbf{x}_i^{s,L}, \mathbf{x}_i^{s,H}) = -\log \frac{\exp Sim^{\mathcal{L}}(\mathbf{x}_i^{s,L}, \mathbf{x}_i^{s,H})}{\sum_{j=1}^{|\mathcal{V}|} \mathbb{I}\{i \neq j\} \exp Sim^{\mathcal{L}}(\mathbf{x}_i^{s,L}, \mathbf{x}_i^{s,H})}$$

Similarity between low- and high-level representations of node *i*

Inter-distillation: maximize the agreement between the outputs of old and new models

$$\mathcal{J}(\mathbf{x}_i^{t,H}, \mathbf{x}_i^{s,H}) = -\log \frac{\exp Sim^{\mathcal{L}}(\mathbf{x}_i^{t,H}, \mathbf{x}_i^{s,H})}{\sum_{j=1}^{|\mathcal{V}|} \mathbb{I}\{i \neq j\} \exp Sim^{\mathcal{L}}(\mathbf{x}_i^{t,H}, \mathbf{x}_i^{s,H})}$$

Similarity between high-level representations of node *i* from teacher (old) and student (new) model

Sun, Li, et al. "Self-Supervised Continual Graph Learning in Adaptive Riemannian Spaces." *arXiv preprint arXiv:2211.17068* (2022).

Graph Structure Aware Incremental Learning (GraphSAIL, Regularization based)

GraphSAIL

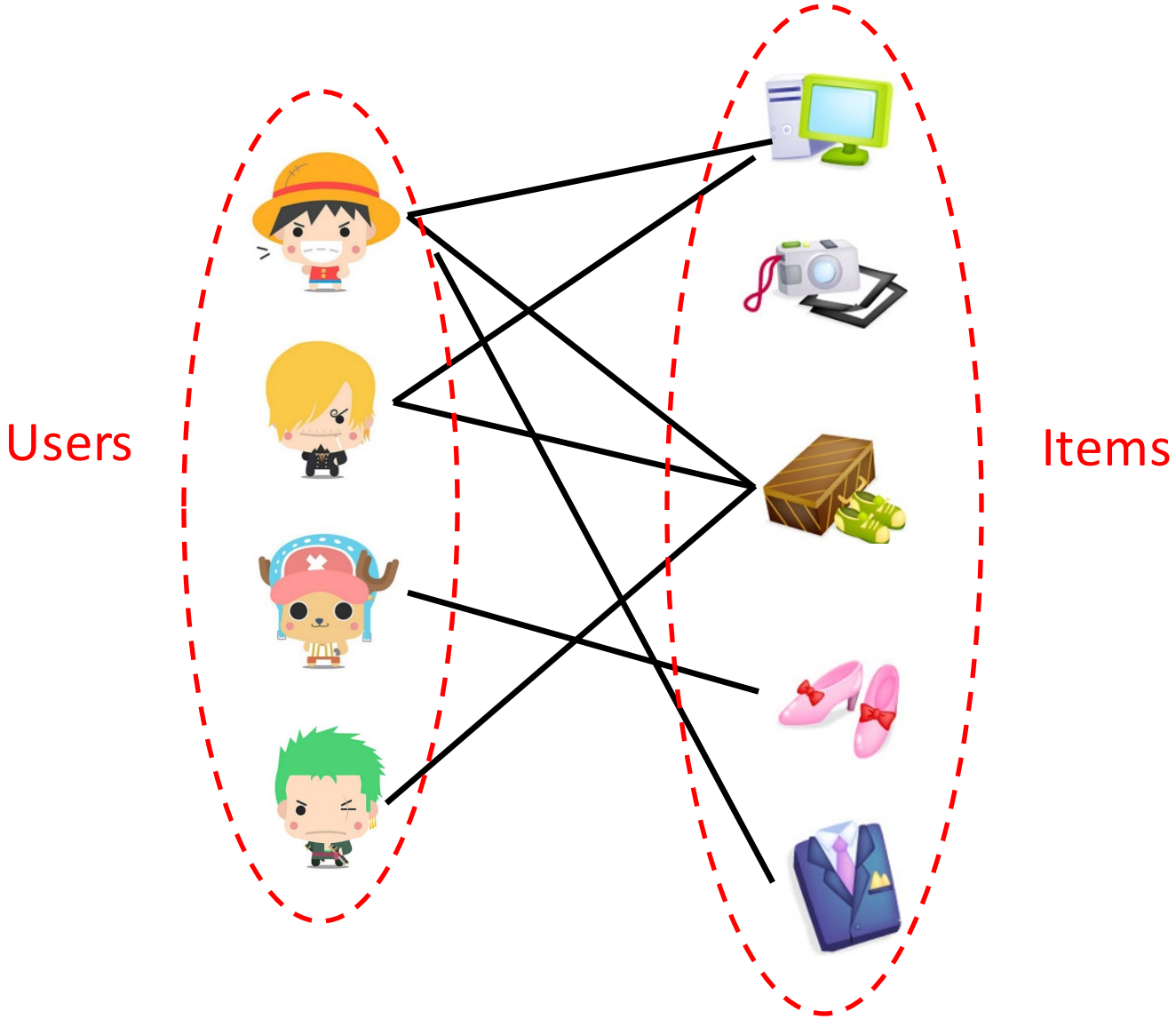
Local structure distillation:
Preserve the local neighbourhood information

Global structure distillation:
preserve the relative position of each node within the graph

Self-embedding distillation:
preserve the intra-information contained in each user/item node

Xu, Yishi, et al. "Graphsail: Graph structure aware incremental learning for recommender systems." *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020.

Recommender system graph



Graph Structure Aware Incremental Learning (GraphSAIL, Regularization based)

Local structure distillation

Users preference at t-1

Users preference at t

$$\mathcal{L}_{local} = \left(\frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} (emb_u^{t-1} \cdot c_{u,N_u^{t-1}}^{t-1} - emb_u^t \cdot c_{u,N_u^{t-1}}^t)^2 + \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} (emb_i^{t-1} \cdot c_{i,N_i^{t-1}}^{t-1} - emb_i^t \cdot c_{i,N_i^{t-1}}^t)^2 \right),$$

users

average local neighbourhood embeddings at t-1

items

$$c_{u,N_u^{t-1}}^t = \frac{1}{|\mathcal{N}_u^{t-1}|} \sum_{i' \in \mathcal{N}_u^{t-1}} emb_{i'}^t$$

$$c_{i,N_u^{t-1}}^t = \frac{1}{|\mathcal{N}_i^{t-1}|} \sum_{u' \in \mathcal{N}_i^{t-1}} emb_{u'}^t .$$

Neighbours of u in task t-1

Xu, Yishi, et al. "Graphsail: Graph structure aware incremental learning for recommender systems." *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020.

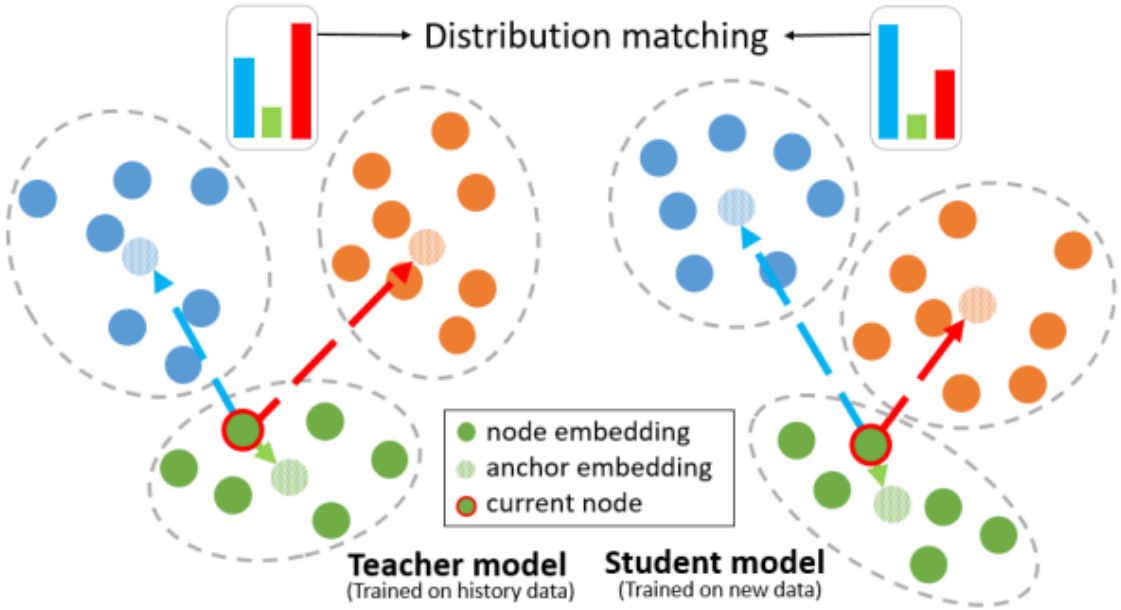
Graph Structure Aware Incremental Learning (GraphSAIL, Regularization based)

Global structure distillation

$\mathcal{A}_u^{t,k}$: Average embedding of cluster k

$$GS^t_{u,\mathcal{A}_u^{t,k}} = \frac{e^{SIM(emb_u^t, \mathcal{A}_u^{t,k})/\tau}}{\sum_{k'=1}^{K} e^{SIM(emb_u^t, \mathcal{A}_u^{t,k'})/\tau}}, \quad SIM(z_i, z_j) = z_i^T z_j$$

Similarity distribution over anchor embeddings (average of each cluster)



$$S_{u,\mathcal{A}_u} = D_{KL}(GS^t_{u,\mathcal{A}_u^t} || GS^{t-1}_{u,\mathcal{A}_u^{t-1}}) = \sum_{k=1}^{K} GS^t_{u,\mathcal{A}_u^{t,k}} \log\left(\frac{GS^t_{u,\mathcal{A}_u^{t,k}}}{GS^{t-1}_{u,\mathcal{A}_u^{t-1,k}}}\right)$$

Distance between two similarity distributions (at t and t-1)

Xu, Yishi, et al. "Graphsail: Graph structure aware incremental learning for recommender systems." *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020.

Graph Structure Aware Incremental Learning (GraphSAIL, Regularization based)

Self-embedding distillation

$$\mathcal{L}_{self} = \left(\frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{\eta_u}{||\eta_U||_2} ||emb_u^{t-1} - emb_u^t||_2 + \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \frac{\eta_i}{||\eta_I||_2} ||emb_i^{t-1} - emb_i^t||_2 \right)$$

Distance between the embeddings of u at t and t-1

$$\eta_u = \frac{|\mathcal{N}_u^{t-1}|}{|\mathcal{N}_u^t|}, \qquad \eta_i = \frac{|\mathcal{N}_i^{t-1}|}{|\mathcal{N}_i^t|}$$

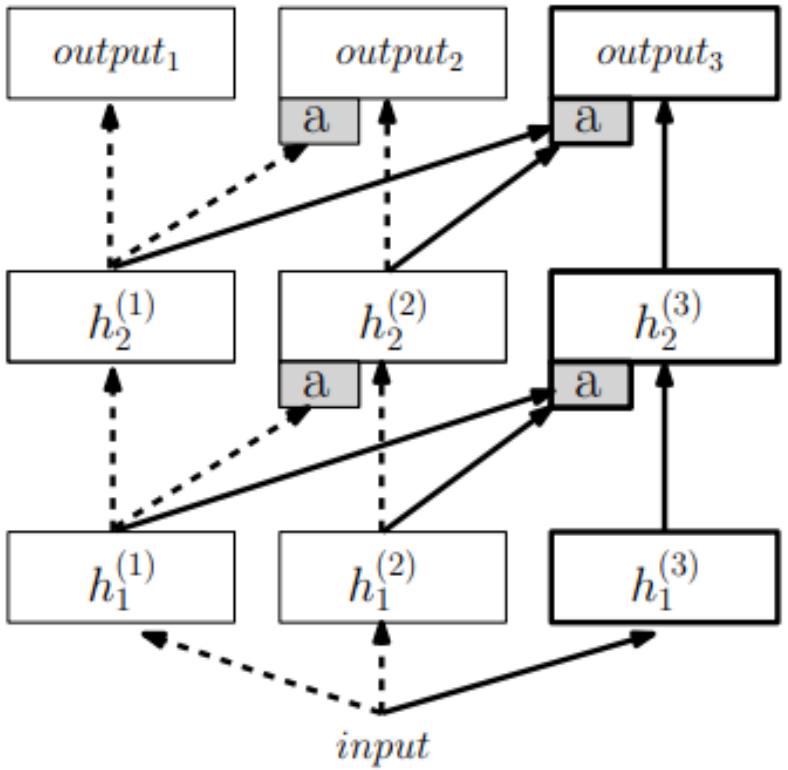
Coefficients controlling the distillation strength

Xu, Yishi, et al. "Graphsail: Graph structure aware incremental learning for recommender systems." *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020.

Rough Categories

- **Regularization:** Penalize changes to model parameters via regularizations
- **Parameter-isolation:** Separate parameters for new and old tasks (partially or entirely)
- **Memory-replay:** Replay old task data to the model when learning new ones

Progressive neural network (Parameter isolation based)



From current task From previous tasks

$$h_i^{(k)} = f\left(W_i^{(k)} h_{i-1}^{(k)} + \sum_{j<k} U_i^{(k:j)} h_{i-1}^{(j)}\right)$$

$W_i^{(k)}$ weight matrix of layer i of column k

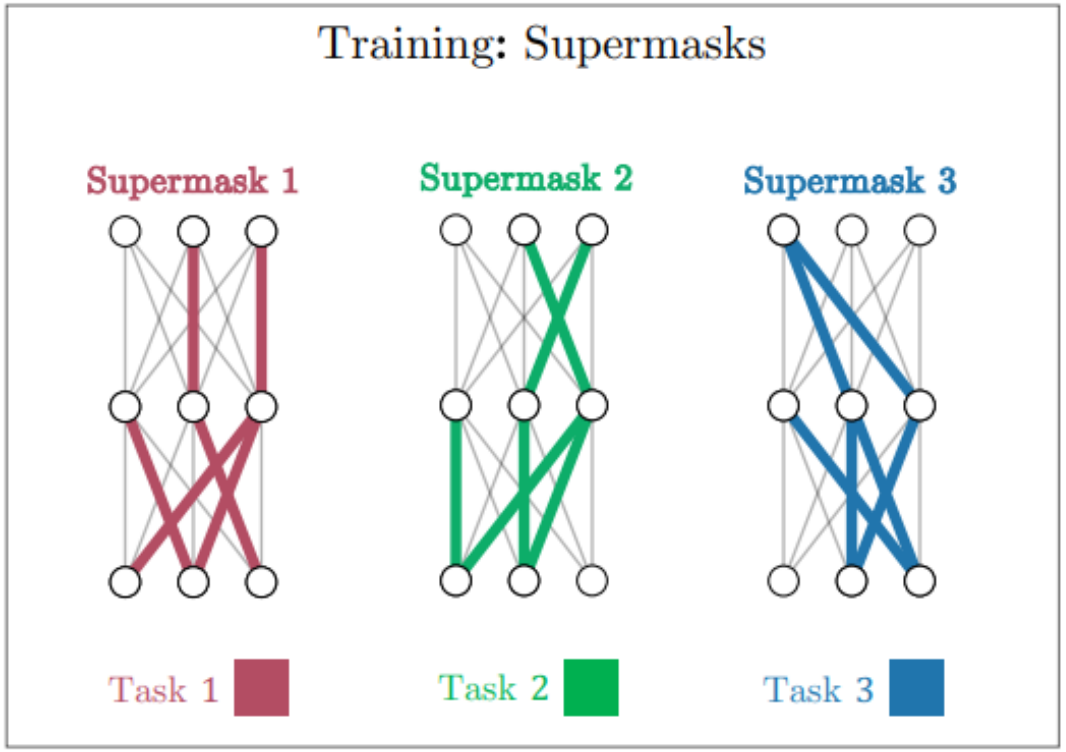
$U_i^{(k:j)}$ layer $i-1$ of column j, to layer i of column k

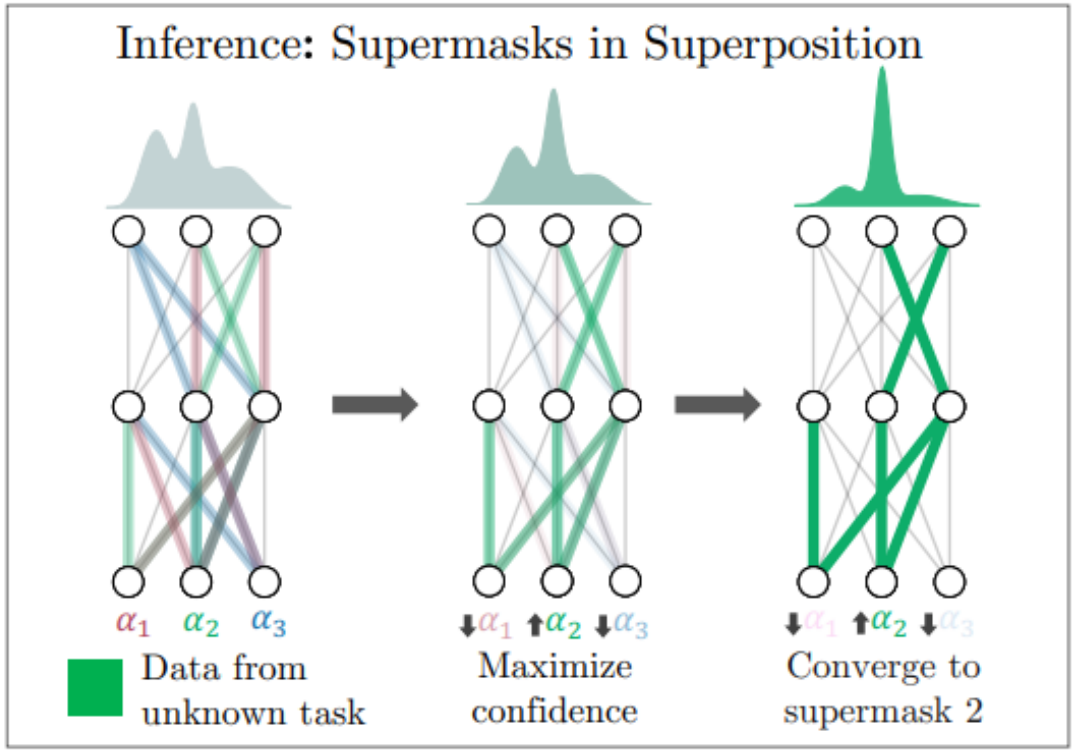
$$f(x) = \max(0, x)$$

Separate branch for each task

Rusu, Andrei A., et al. "Progressive neural networks." arXiv preprint arXiv:1606.04671 (2016).

Supermasks in Superposition (Parameter isolation based)





Learn a Supermask atop the backbone for each task: $\mathbf{p} = f(\mathbf{x}, W \odot M)$

Without task indicators: $\mathbf{p}(\alpha) = f\left(\mathbf{x}, W \odot \left(\sum_{i=1}^{k} \alpha_i M^i\right)\right)$ $\arg\max_i \left(-\frac{\partial \mathcal{H}(\mathbf{p}(\alpha))}{\partial \alpha_i}\right)$

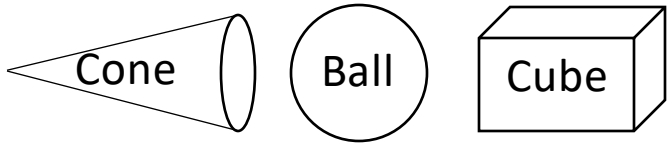
Wortsman, Mitchell, et al. "Supermasks in superposition." Advances in Neural Information Processing Systems 33 (2020): 15173-15184.

Hierarchical prototype networks (HPNs, parameter-isolation based)

Motivation: Inspired by human cognition process, Different representations of data in different classes may be better represented by different combinations of a shared pool of basic features

Basic feature types: **Color** **Shape**

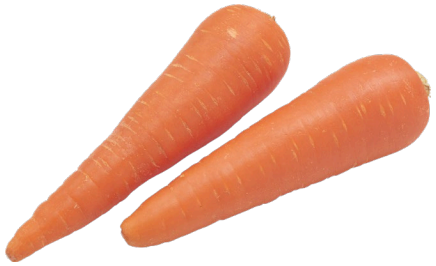
Basic feature instances: Orange Red Green



Objects (combinations of basic feature instances):

Green Ball

Red Ball



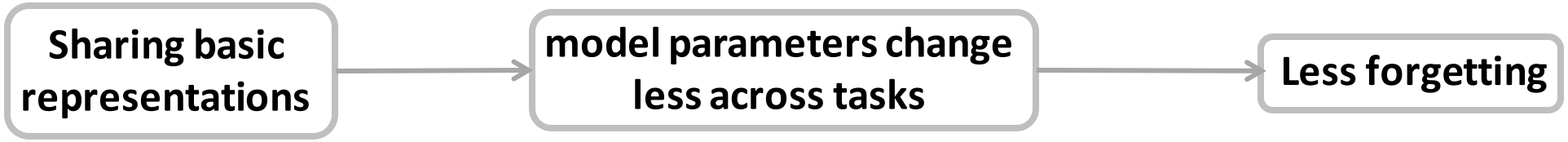
Orange Cone

Class (higher level hierarchy): **Fruits** **Vegetables**

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Hierarchical prototype networks for continual graph representation learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

Hierarchical prototype networks (HPNs, parameter-isolation based)

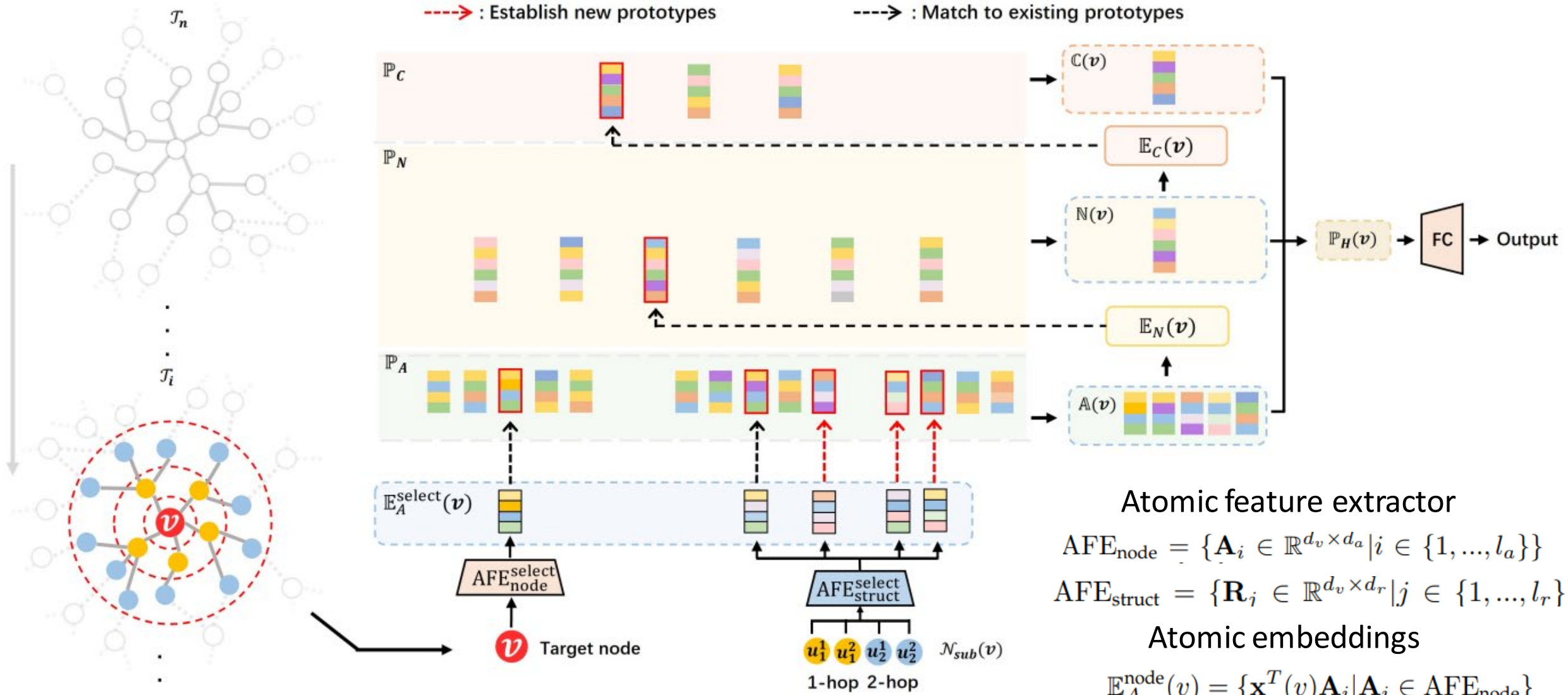
Graph nodes (e.g. people in social network, papers in citation networks) should also be decomposable.



Analogy	Our design
Basic feature types (color, shape)	**Atomic feature extractor (AFE)**
Basic feature instances (orange, red, green, etc.)	**Atom prototype (A-Prototype)**
Objects (watermelon, tomato, etc.)	**Node-level prototype (N-Prototype)**
Class (fruit, vegetable)	**Class-level prototype (C-Prototype)**

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Hierarchical prototype networks for continual graph representation learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

Hierarchical prototype networks (HPNs, parameter-isolation based)



Atomic feature extractor

$$\text{AFE}_{\text{node}} = \{\mathbf{A}_i \in \mathbb{R}^{d_v \times d_a} | i \in \{1, ..., l_a\}\}$$

$$\text{AFE}_{\text{struct}} = \{\mathbf{R}_i \in \mathbb{R}^{d_v \times d_r} | j \in \{1, ..., l_r\}\}$$

Atomic embeddings

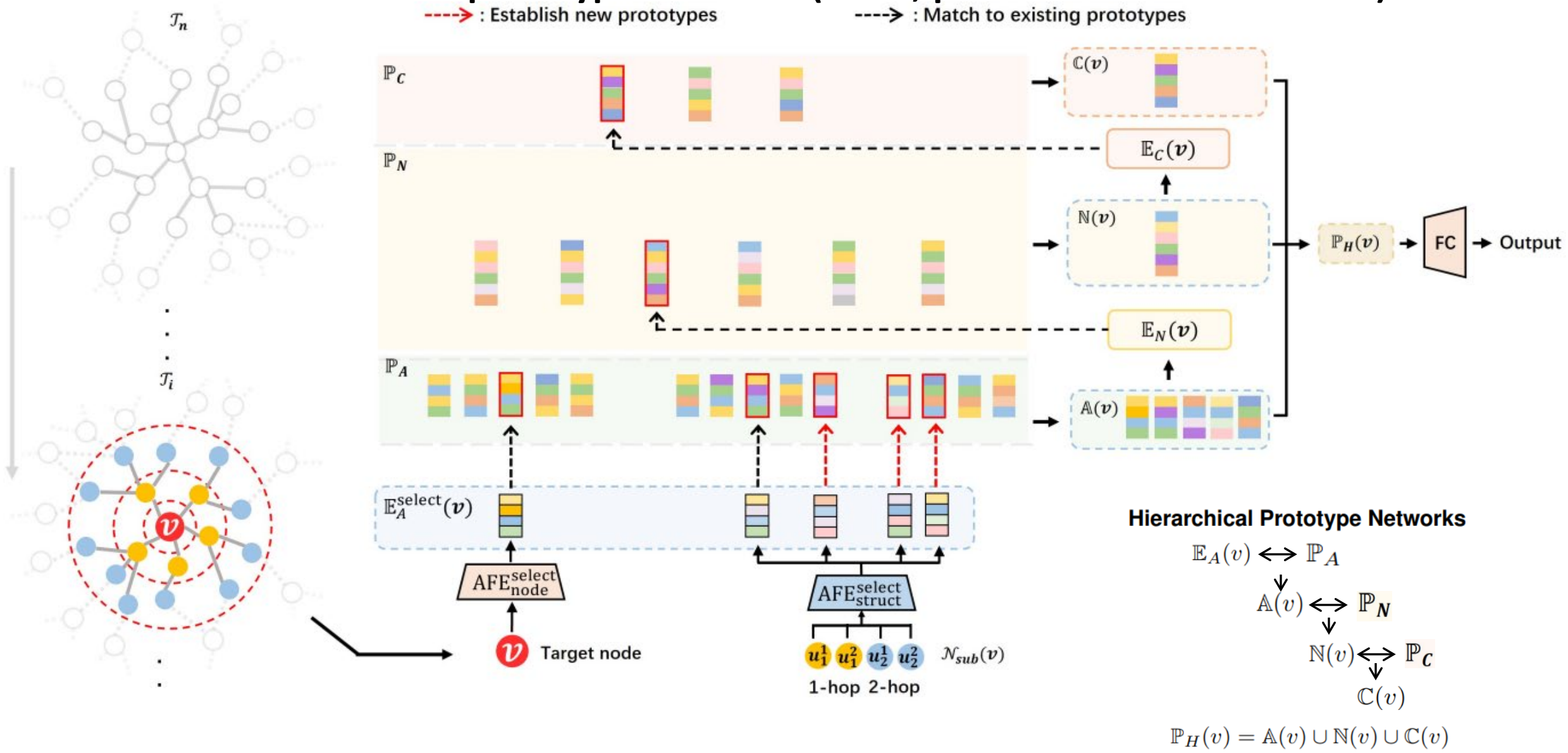
$$\mathbb{E}_A^{\text{node}}(v) = \{\mathbf{x}^T(v)\mathbf{A}_i | \mathbf{A}_i \in \text{AFE}_{\text{node}}\}$$

$$\mathbb{E}_A^{\text{struct}}(v) = \{\mathbf{x}^T(u)\mathbf{R}_i | \mathbf{R}_i \in \text{AFE}_{\text{struct}}, u \in \mathcal{N}_{sub}\}$$

$$\mathbb{E}_A(v) = \mathbb{E}_A^{\text{node}}(v) \cup \mathbb{E}_A^{\text{struct}}(v)$$

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Hierarchical prototype networks for continual graph representation learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

Hierarchical prototype networks (HPNs, parameter-isolation based)



Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Hierarchical prototype networks for continual graph representation learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

Hierarchical prototype networks (HPNs, parameter-isolation based)

- Theoretical upper bound on the number of prototypes

Theorem 1 (Upper bounds on numbers of prototypes). *Given the notations defined in HPNs, the upper bound on the number of A-prototypes n_a can be given by*

$$n_A \leqslant (l_a + l_r) \max_N S(d_a, N, 1 - t_A), \tag{14}$$

and the upper bounds on the number of N-prototypes and the C-prototypes are:

$$n_N \leqslant \max_N S(d_n, N, 1 - t_N) \tag{15}$$

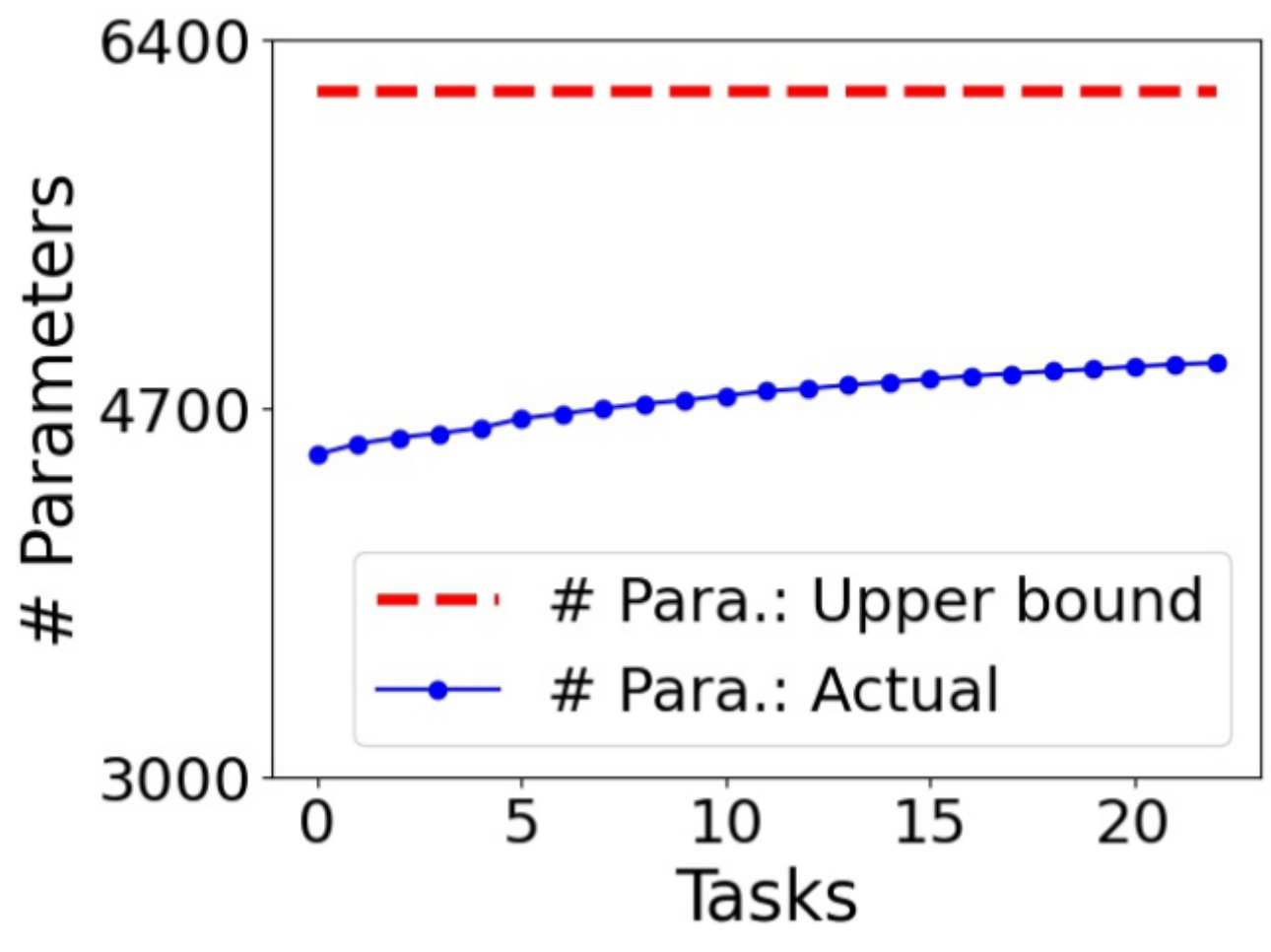
$$n_C \leqslant \max_N S(d_c, N, 1 - t_C) \tag{16}$$

where $S(n, N, t)$ is the spherical code defined on a n dimensional hypersphere .

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Hierarchical prototype networks for continual graph representation learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

Hierarchical prototype networks (HPNs, parameter-isolation based)

- Theoretical upper bound on the number of prototypes



Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Hierarchical prototype networks for continual graph representation learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

Hierarchical prototype networks (HPNs, parameter-isolation based)

- Theoretical justification of the capability to overcome forgetting

Theorem 2 (Task distance preserving). *For* HPNs *trained on consecutive tasks* $\mathcal{T}^p$ *and* $\mathcal{T}^{p+1}$. *If* $l_a d_a + l_r d_r \geqslant (l_r + 1)d_v$ *and* $\mathbf{W}$ *is column full rank, then as long as* $t_A < \lambda_{\min}(l_r + 1)\mathbf{dist}(\mathbb{V}_p, \mathbb{V}_{p+1})$, *learning on* $\mathcal{T}^{p+1}$ *will not modify representations* HPNs *generate for data from* $\mathcal{T}^p$, *i.e., catastrophic forgetting is avoided.*

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Hierarchical prototype networks for continual graph representation learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

Hierarchical prototype networks (HPNs, parameter-isolation based)

C.L.T.	Base	Cora		Citeseer		Actor		OGB-Arxiv		OGB-Products	
		AM/%	FM/%	AM/%	FM/%	AM/%	FM /%	AM/%	FM /%	AM/%	FM /%
None	GCN	63.5±1.9	-42.3±0.4	64.5±3.9	-7.7±1.6	43.6±3.6	-9.1±2.9	56.8±4.3	-19.8±3.2	45.2±5.6	-27.8±7.1
	GAT	71.9±3.8	-33.1±2.3	66.8±0.9	-19.6±0.3	53.1±2.7	-4.3±1.6	54.3±3.5	-21.7±4.6	44.9±6.9	-30.3±5.2
	GIN	68.3±2.3	-35.4±3.4	57.7±2.3	-36.4±0.3	45.5±2.3	-8.9±2.8	53.2±6.5	-23.5±8.1	43.1±7.4	-31.4±8.8
EWC [24]	GCN	63.1±1.2	-42.7±1.6	54.4±4.2	-30.3±0.9	44.3±1.1	-7.1±1.4	72.1±2.4	-9.1±1.9	66.7±0.5	-8.4±0.4
	GAT	72.2±1.5	-32.2±1.6	65.7±2.5	-19.7±2.3	54.2±2.5	-2.5±1.5	73.2±1.1	-10.8±2.1	67.9±1.0	-9.6±1.3
	GIN	69.6±2.6	-28.5±2.8	57.9±3.4	-36.3±2.4	47.6±2.1	-7.2±1.6	74.1±1.7	-8.3±2.0	67.3±2.3	-13.6±1.5
LwF [23]	GCN	76.1±1.4	-21.3±2.4	67.0±0.2	-8.3±2.7	49.7±2.5	-3.6±1.4	69.9±3.9	-12.1±2.8	66.3±2.5	-11.8±3.4
	GAT	70.8±2.8	-34.6±4.1	66.1±4.1	-18.9±1.5	52.8±2.7	-6.2±2.2	68.9±4.4	-13.6±3.3	65.1±4.1	-13.2±2.9
	GIN	74.1±2.7	-23.3±0.8	63.1±1.9	-16.5±2.2	49.7±2.6	-4.1±1.5	71.4±4.8	-15.9±5.6	65.9±4.0	-10.7±3.1
GEM [27]	GCN	75.7±3.0	-6.5±4.4	41.8±2.6	-31.9±1.4	52.7±3.1	+3.9±2.9	75.4±1.7	-13.6±0.5	71.3±1.7	-10.5±0.9
	GAT	69.8±3.0	-26.1±2.6	71.3±2.2	+9.0±1.5	54.3±3.6	-2.0±0.9	76.6±0.7	-11.3±0.4	70.4±0.8	-10.9±1.6
	GIN	80.2±3.3	-2.0±4.2	49.7±0.5	-24.5±0.9	45.2±2.8	-11.1±1.5	77.3 ±2.1	-11.2±1.6	76.5±3.3	-7.2±2.5
MAS [59]	GCN	65.5±1.9	-21.4±3.7	59.5±3.1	-0.1±2.4	50.7±2.4	-1.5±0.8	69.8±0.4	-18.8±0.9	62.0±1.1	-17.9±1.9
	GAT	84.7±0.7	-5.6±2.0	69.1±1.1	-4.8±3.3	53.7±2.6	-1.6±0.8	70.6±1.3	-16.7±1.6	64.4±2.3	-14.5±3.2
	GIN	76.7±2.6	-4.0±3.6	65.2±3.9	+0.0±2.0	51.6±2.6	-0.6±1.3	65.3 ±2.9	-17.0±2.3	61.4±3.8	-20.9±2.9
ERGN. [60]	GCN	63.5±2.4	-42.3±0.7	54.2±3.9	-30.3±1.9	52.4±3.3	+0.6±1.4	63.3±1.7	-18.1±0.9	60.7±2.8	-26.6±3.3
	GAT	71.1±2.5	-34.3±1.0	65.5±0.3	-20.4±3.9	51.4±2.2	-7.2±3.2	63.5±2.4	-19.5±1.9	61.3±1.7	-25.1±0.8
	GIN	68.3±0.4	-35.4±0.4	57.7±3.1	-36.4±1.3	42.7±3.9	-13.0±2.1	69.2±1.8	-11.8±1.4	61.8±4.7	-23.4±7.9
TWP [18]	GCN	68.9±0.9	-5.7±1.5	60.5±3.8	-0.3±4.4	50.6±2.0	-4.8±1.1	75.6±0.3	-10.4±0.5	69.9±0.4	-9.0±1.1
	GAT	81.3±3.2	-14.4±1.5	69.8±1.5	-8.9±2.6	54.0±1.8	-2.1±1.9	75.8±0.5	-5.9±0.3	69.3±2.3	-8.9±1.5
	GIN	73.7±3.2	-3.9 ±2.6	68.9±0.7	-2.4±1.9	49.9±1.9	-3.6±2.0	76.6±1.8	-11.3±1.1	69.9±1.4	-10.3±2.7
Join.	GCN	93.7± 0.5	-	78.9±0.4	-	57.0±0.9	-	77.2±0.8	-	72.9±1.2	-
	GAT	93.9± 0.9	-	79.3±0.8	-	57.1±0.9	-	81.8±0.3	-	73.7±2.4	-
	GIN	93.2± 1.2	-	78.7±0.9	-	56.9±0.6	-	82.3±1.9	-	77.9±2.1	-
HPNs		**93.7±1.5**	**+0.6±1.0**	**79.0±0.9**	**-0.6±0.7**	**56.8±1.4**	**-0.9±0.9**	**85.8± 0.7**	**+0.6±0.9**	**80.1±0.8**	**+2.9±1.0**

Comparison with State-of-the-arts

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Hierarchical prototype networks for continual graph representation learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

Rough Categories

- **Regularization:** Penalize changes to model parameters via regularizations
- **Parameter-isolation:** Separate parameters for new and old tasks (partially or entirely)
- **Memory-replay:** Replay old task data to the model when learning new ones

Tiny Episodic Memories (memory replay based)

Algorithm 1 Experience Replay for Continual Learning.

procedure ER($\mathcal{D}$, mem_sz, batch_sz, lr)
  $\mathcal{M} \leftarrow \{\} * \text{mem\_sz}$ ▷ Allocate memory buffer of size mem_sz
  $n \leftarrow 0$ ▷ Number of training examples seen in the continuum
  for $t \in \{1, \cdots, T\}$ **do**
    for $B_n \overset{K}{\sim} \mathcal{D}_t$ **do** ▷ Sample without replacement a mini-batch of size K from task t
      $B_{\mathcal{M}} \overset{K}{\sim} \mathcal{M}$ ▷ Sample a mini-batch from $\mathcal{M}$
      $\theta \leftarrow SGD(B_n \cup B_{\mathcal{M}}, \theta, \text{lr})$ ▷ Single gradient step to update the parameters by stacking current minibatch with minibatch from memory
      $\mathcal{M} \leftarrow \texttt{UpdateMemory}(\text{mem\_sz}, t, n, B_n)$ ▷ Memory update, see §4
      $n \leftarrow n + \text{batch\_sz}$ ▷ Counter update
  return $\theta, \mathcal{M}$

Memory replay

Memory update

Representative CL techniques

Chaudhry, Arslan, et al. "On tiny episodic memories in continual learning." *arXiv preprint arXiv:1902.10486* (2019).

Tiny Episodic Memories (memory replay based)

Algorithm 2 Reservoir sampling update. mem_sz is the number of examples the memory can store, t is the task id, n is the number of examples observed so far in the data stream, and B is the input mini-batch.

procedure UPDATEMEMORY(mem_sz, t, n, B)
  $j \leftarrow 0$
  for $(\mathbf{x}, y)$ in B **do**
    $M \leftarrow |\mathcal{M}|$ ▷ Number of samples currently stored in the memory
    if $M <$ mem_sz **then**
      $\mathcal{M}$.append$(\mathbf{x}, y, t)$
    else
      $i =$ randint$(0, n + j)$
      if $i <$ mem_sz **then**
        $\mathcal{M}[i] \leftarrow (\mathbf{x}, y, t)$ ▷ Overwrite memory slot.
    $j \leftarrow j + 1$
  return $\mathcal{M}$

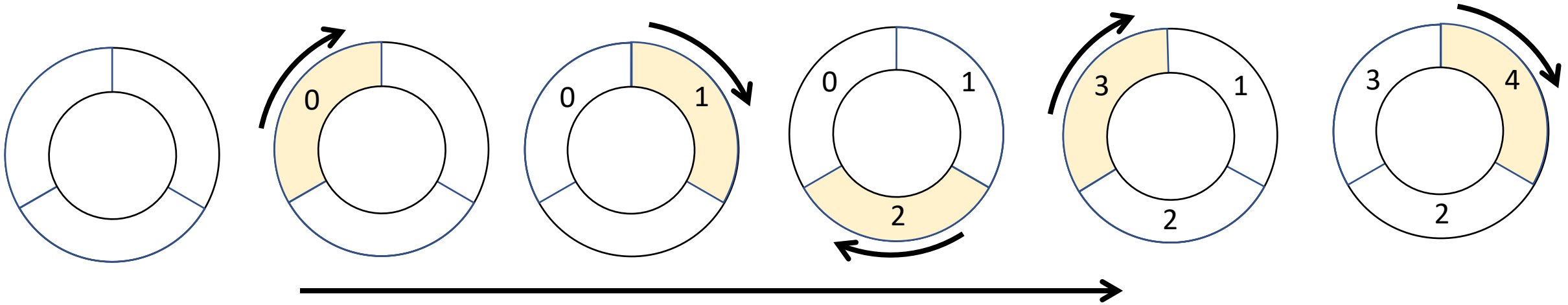
Representative CL techniques

Chaudhry, Arslan, et al. "On tiny episodic memories in continual learning." *arXiv preprint arXiv:1902.10486* (2019).

Tiny Episodic Memories (memory replay based)

Algorithm 3 Ring buffer.

```
procedure UPDATEMEMORY(mem_sz, t, n, B)
    for (x, y) in B do
        # Assume FIFO stacks M[t][y] of fixed size are already initialized
        M[t][y].append(x)
    return M
```



Representative CL techniques

Chaudhry, Arslan, et al. "On tiny episodic memories in continual learning." *arXiv preprint arXiv:1902.10486* (2019).

Tiny Episodic Memories (memory replay based)

Algorithm 4 K-Means. Memory is populated using samples closest (in feature space) to sequential K-Means centroids.

procedure UPDATEMEMORY(mem_sz, t, n, B)
    # Assume array $\mathcal{M}[t][y]$ of fixed size is already initialized
    # Assume K centroids c_j are already initialized
    # Assume cluster counters n_j are already initialized to 0
    for $(\mathbf{x}, y)$ in B **do**
        $j \leftarrow \text{argmin}_{j \in \{1,\cdots,K\}} ||\phi_\theta(\mathbf{x}) - c_j||$
        $n_j \leftarrow n_j + 1$
        $c_j \leftarrow c_j + \frac{1}{n_j} * (\phi_\theta(\mathbf{x}) - c_j)$
        $d = ||\phi_\theta(\mathbf{x}) - c_j||$
        if $d < \mathcal{M}[t][y][j]$.get_dst() **then**
            $\mathcal{M}[t][y][j] \leftarrow (\mathbf{x}, d)$
    return $\mathcal{M}$

Choose the closest cluster

Update the cluster center

▷ Store the current example if it is closer to the centroid

Chaudhry, Arslan, et al. "On tiny episodic memories in continual learning." *arXiv preprint arXiv:1902.10486* (2019).

Tiny Episodic Memories (memory replay based)

Algorithm 5 Mean of Features. Store examples that are closest to the running average feature vector.

procedure UPDATEMEMORY(mem_sz, t, n, B)
    # Assume heaps $\mathcal{M}[t][y]$ of fixed size are already initialized
    # Assume average features $f[t][y]$ are already initialized
    # Assume moving average decay hyper-parameter (α) is given
    for $(\mathbf{x}, y)$ in B **do**
        $f[t][y] \leftarrow \alpha * f[t][y] + (1 - \alpha) * \phi_\theta(\mathbf{x})$
        $d = ||\phi_\theta(\mathbf{x}) - f[t][y]||$
        if $\mathcal{M}[t][y]$.find_max() $> d$ **then**
            $\mathcal{M}[t][y]$.delete_max()
            $\mathcal{M}[t][y]$.insert($\mathbf{x}; d$)
    return $\mathcal{M}$

Update the running average

▷ Store the current example if it is closer to the center

Chaudhry, Arslan, et al. "On tiny episodic memories in continual learning." *arXiv preprint arXiv:1902.10486* (2019).

Gradient episodic memory (GEM, memory replay based)

Loss of the stored data:

$$\ell(f_\theta, \mathcal{M}_k) = \frac{1}{|\mathcal{M}_k|} \sum_{(x_i,k,y_i)\in\mathcal{M}_k} \ell(f_\theta(x_i,k), y_i)$$

Memory buffer storing representative data from an old task k

When observing a new triplet (x,y,t):

$$\begin{aligned} \text{minimize}_\theta \quad & \ell(f_\theta(x,t),y) \\ \text{subject to} \quad & \ell(f_\theta,\mathcal{M}_k) \le \ell(f_\theta^{t-1},\mathcal{M}_k) \text{ for all } k<t \end{aligned}$$

$$\langle g, g_k\rangle := \left\langle \frac{\partial \ell(f_\theta(x,t),y)}{\partial\theta}, \frac{\partial \ell(f_\theta,\mathcal{M}_k)}{\partial\theta} \right\rangle \ge 0, \text{ for all } k<t.$$

Lopez-Paz, David, and Marc'Aurelio Ranzato. "Gradient episodic memory for continual learning." Advances in neural information processing systems 30 (2017).

Gradient episodic memory (GEM, memory replay based)

Loss of the stored data:

$$\ell(f_\theta, \mathcal{M}_k) = \frac{1}{|\mathcal{M}_k|} \sum_{(x_i,k,y_i)\in\mathcal{M}_k} \ell(f_\theta(x_i,k), y_i)$$

Memory buffer storing representative data from an old task k

When observing a new triplet (x,y,t):

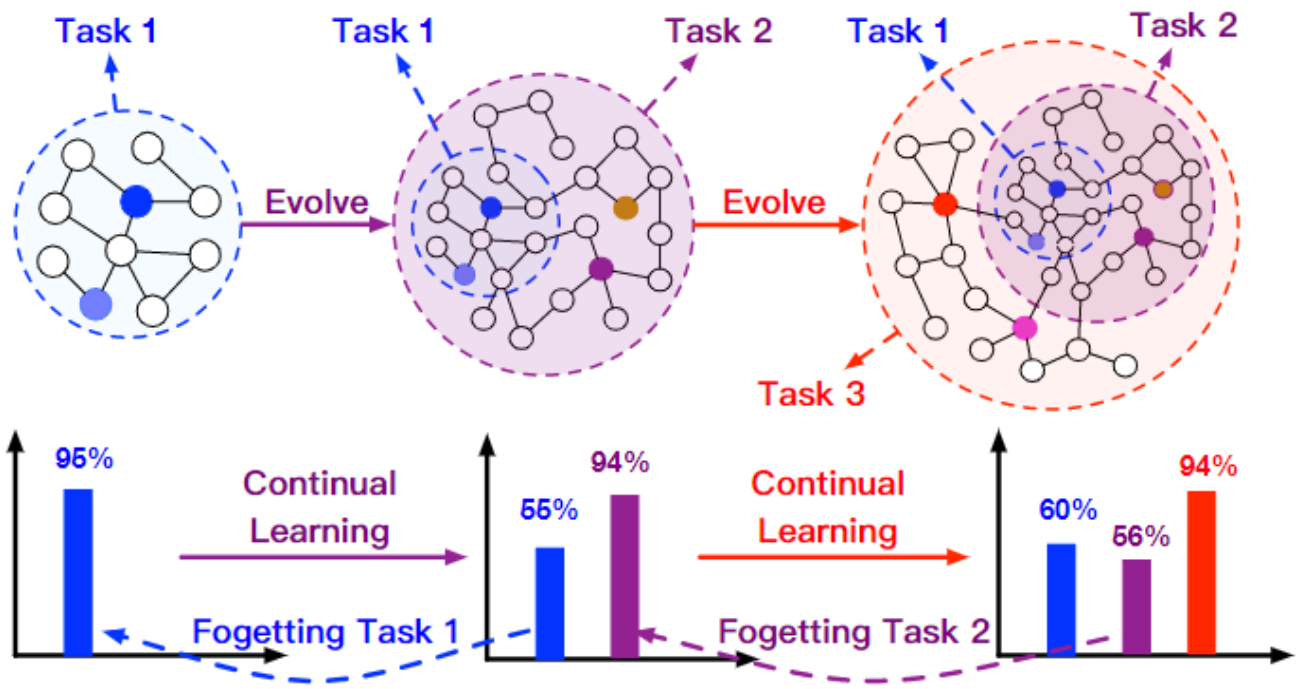
$$\begin{aligned} \text{minimize}_\theta \quad & \ell(f_\theta(x,t), y) \\ \text{subject to} \quad & \ell(f_\theta, \mathcal{M}_k) \le \ell(f_\theta^{t-1}, \mathcal{M}_k) \text{ for all } k < t \end{aligned}$$

$$\langle g, g_k \rangle := \left\langle \frac{\partial \ell(f_\theta(x,t), y)}{\partial \theta}, \frac{\partial \ell(f_\theta, \mathcal{M}_k)}{\partial \theta} \right\rangle \ge 0, \text{ for all } k < t.$$

Aim: Find directions in the parameter space that are favourable for all tasks

Lopez-Paz, David, and Marc'Aurelio Ranzato. "Gradient episodic memory for continual learning." Advances in neural information processing systems 30 (2017).

Experience Replay Graph Neural Networks (ER-GNN, memory replay based)



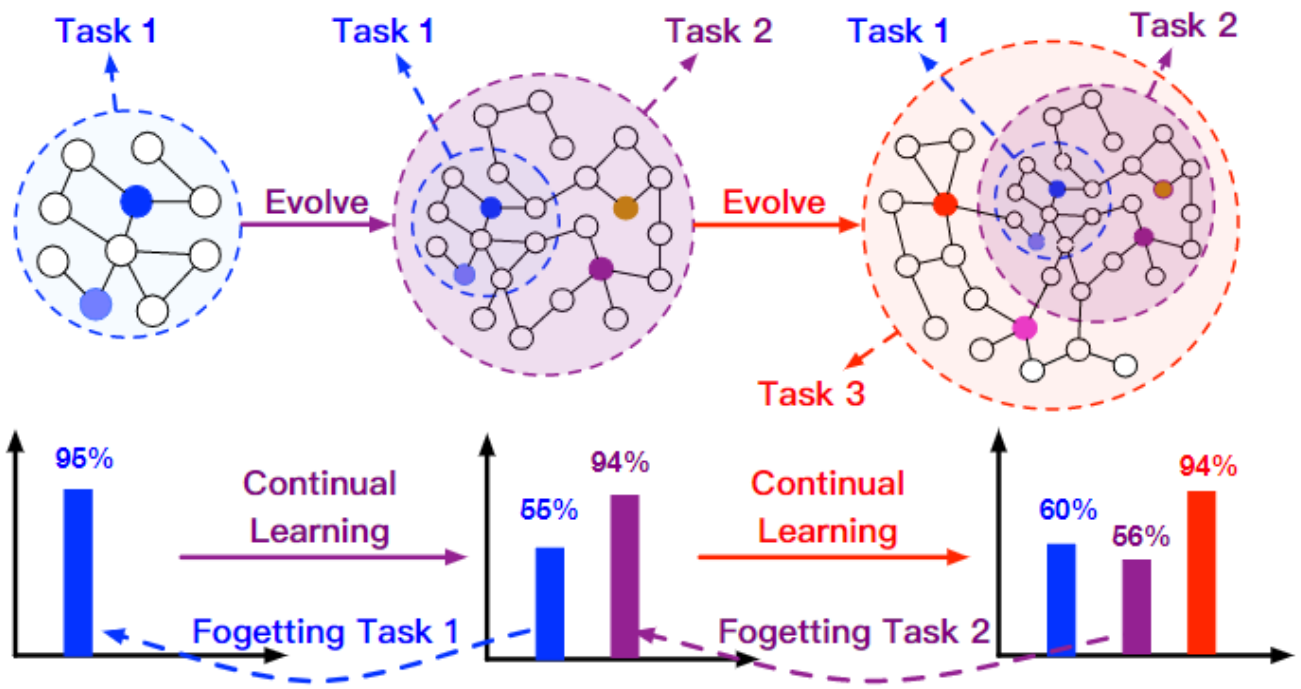
Store representative nodes
(without topological information)

Original loss:

$$\mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}) = -(\sum_{(\boldsymbol{x_i}, y_i) \in \mathcal{D}} (y_i \log f_{\boldsymbol{\theta}}(\boldsymbol{x_i}) + (1 - y_i) \log(1 - f_{\boldsymbol{\theta}}(\boldsymbol{x_i}))))$$

Zhou, Fan, and Chengtai Cao. "Overcoming catastrophic forgetting in graph neural networks with experience replay." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. No. 5. 2021.

Experience Replay Graph Neural Networks (ER-GNN, memory replay based)



Store representative nodes
(without topological information)

Original loss:

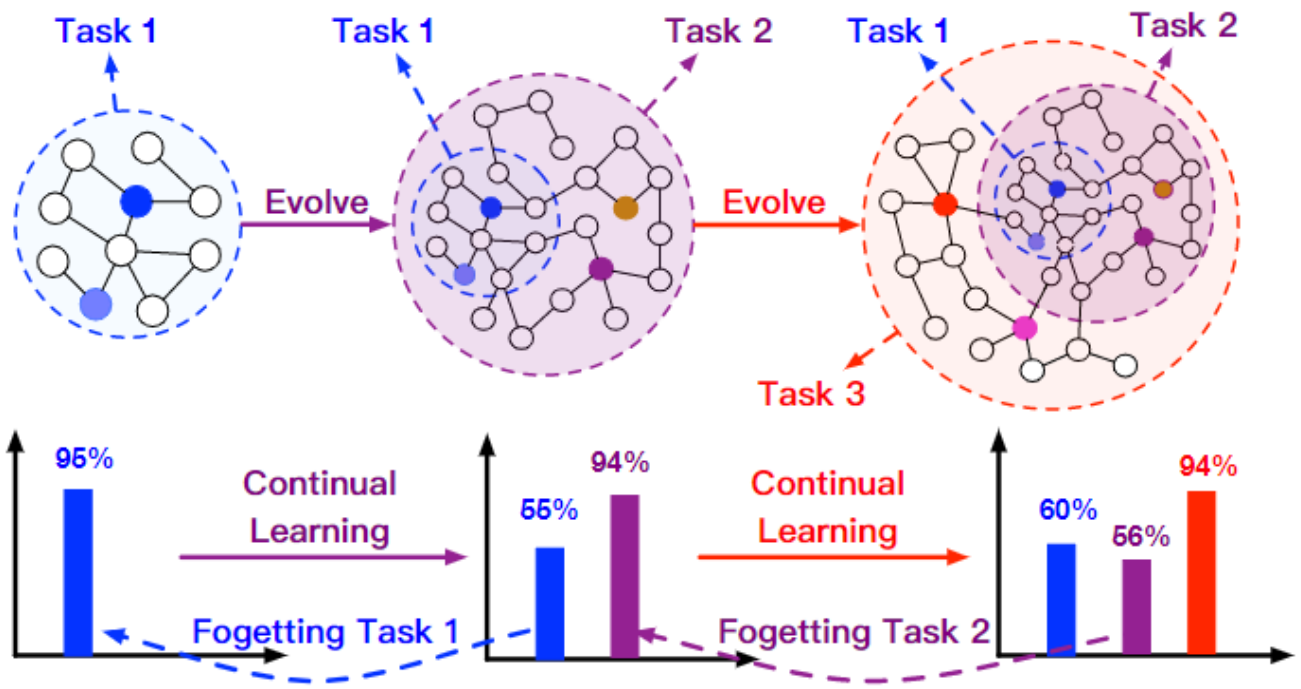
$$\mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}) = -(\sum_{(\boldsymbol{x_i}, y_i) \in \mathcal{D}} (y_i \log f_{\boldsymbol{\theta}}(\boldsymbol{x_i}) + (1 - y_i) \log(1 - f_{\boldsymbol{\theta}}(\boldsymbol{x_i}))))$$

Loss with memory buffer:

$$\mathcal{L}'_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}, B) = \beta \mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}) + (1 - \beta) \mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, B)$$

Zhou, Fan, and Chengtai Cao. "Overcoming catastrophic forgetting in graph neural networks with experience replay." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. No. 5. 2021.

Experience Replay Graph Neural Networks (ER-GNN, memory replay based)



Store representative nodes
(without topological information)

Original loss:

$$\mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}) = -(\sum_{(\boldsymbol{x_i}, y_i) \in \mathcal{D}} (y_i \log f_{\boldsymbol{\theta}}(\boldsymbol{x_i}) + (1 - y_i) \log(1 - f_{\boldsymbol{\theta}}(\boldsymbol{x_i}))))$$

Loss with memory buffer:

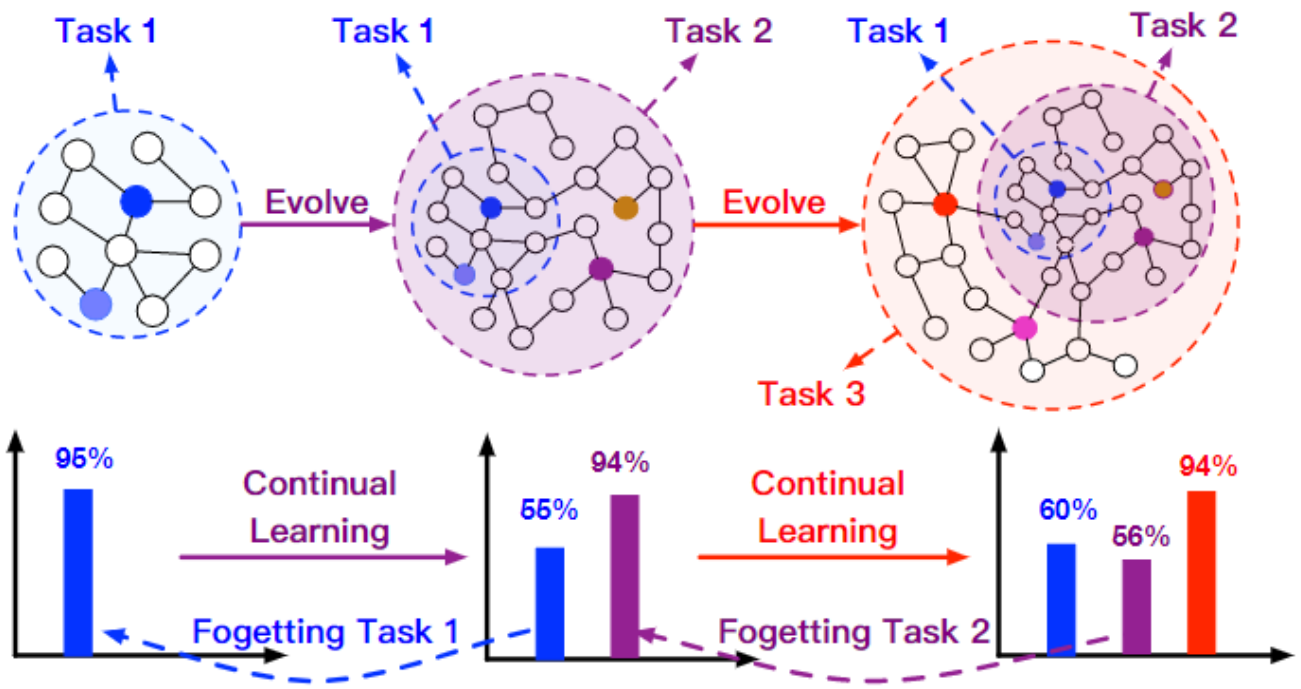
$$\mathcal{L}'_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}, B) = \beta \mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}) + (1 - \beta) \mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, B)$$

Objective:

$$\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta} \in \Theta} (\mathcal{L}'_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}, B))$$

Zhou, Fan, and Chengtai Cao. "Overcoming catastrophic forgetting in graph neural networks with experience replay." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. No. 5. 2021.

Experience Replay Graph Neural Networks (ER-GNN, memory replay based)



Store representative nodes
(without topological information)

Original loss:

$$\mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}) = -(\sum_{(\boldsymbol{x_i}, y_i)\in\mathcal{D}} (y_i \log f_{\boldsymbol{\theta}}(\boldsymbol{x_i}) + (1-y_i)\log(1-f_{\boldsymbol{\theta}}(\boldsymbol{x_i}))))$$

Loss with memory buffer:

$$\mathcal{L}'_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}, B) = \beta\mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}) + (1-\beta)\mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, B)$$

Objective:

$$\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}\in\Theta}(\mathcal{L}'_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}, B))$$

Balance the contribution:

$$\beta = |B|/(|\mathcal{D}_i^{\text{tr}}| + |B|)$$

$|\mathcal{D}_i^{\text{tr}}|$: Size of training set of the current task

$|B|$: Size of training set of the memory buffer

Zhou, Fan, and Chengtai Cao. "Overcoming catastrophic forgetting in graph neural networks with experience replay." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. No. 5. 2021.

Sparsified Subgraph Memory (SSM, memory replay based)

GNNs generate node representations based on multi-hop neighborhood

↓

Applying memory replay to graphs requires storing multi-hop neighbors

↓

L-hop neighbors grow in $O(d^L)$, d is average degree

↓

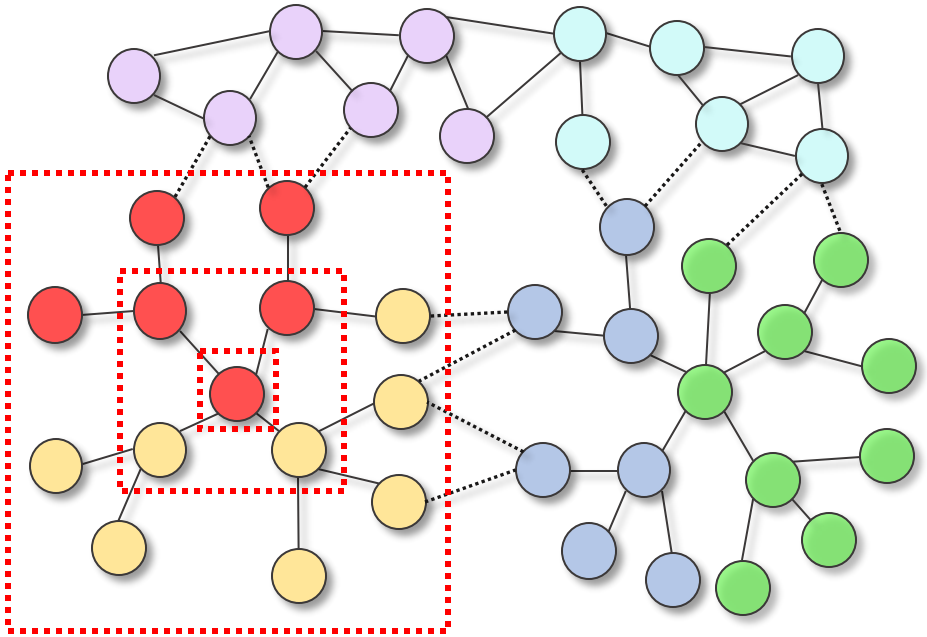
Memory explosion problem

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Sparsified Subgraph Memory for Continual Graph Representation Learning." *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2022.

Sparsified Subgraph Memory (SSM, memory replay based)

- GNNs typically require no less than 2-hop computation subgraphs (i.e. at least two layers)

Exponentially growing computation subgraph

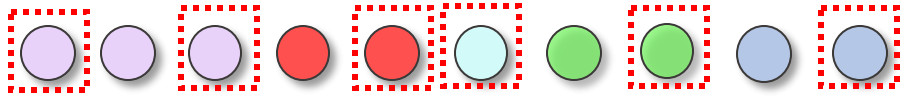


- Size of a computation subgraph grows in $O(d^L)$, d is average degree, L is number of hops
- Memory explosion (e.g. $d = 492$ in Reddit dataset)

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Sparsified Subgraph Memory for Continual Graph Representation Learning." *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2022.

Sparsified Subgraph Memory (SSM, memory replay based)

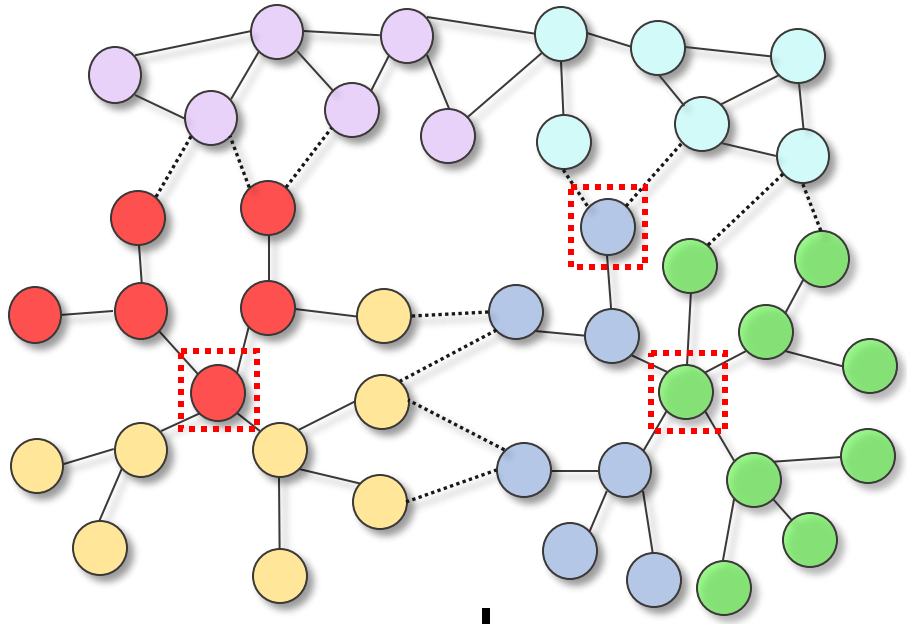
Storing independent data



Sampling (denoted by)

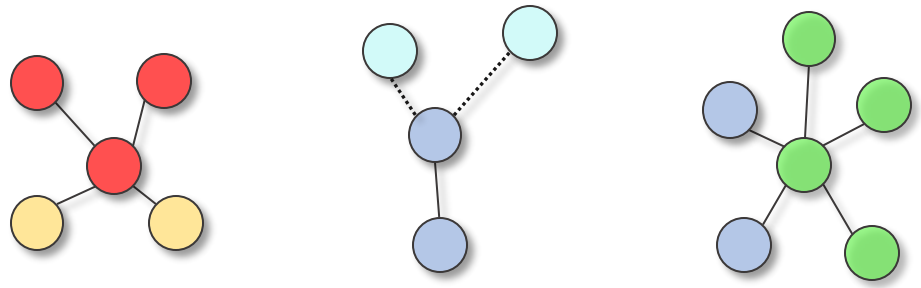
Memory buffer of independent data

Storing graph structured data



Sampling (e.g. keep 1-hop neighbors)

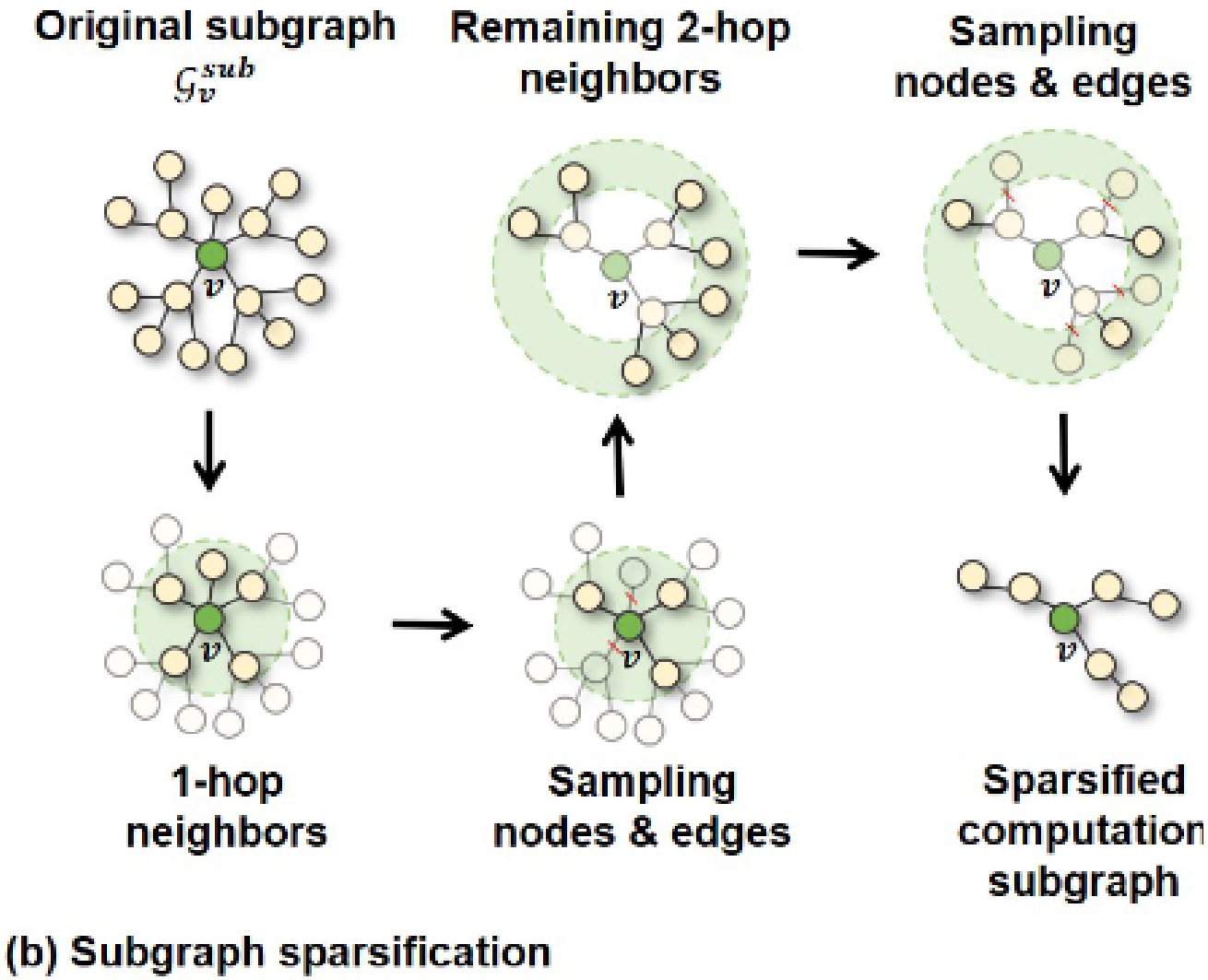
Buffer of computation subgraphs



CGL techniques

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Sparsified Subgraph Memory for Continual Graph Representation Learning." *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2022.

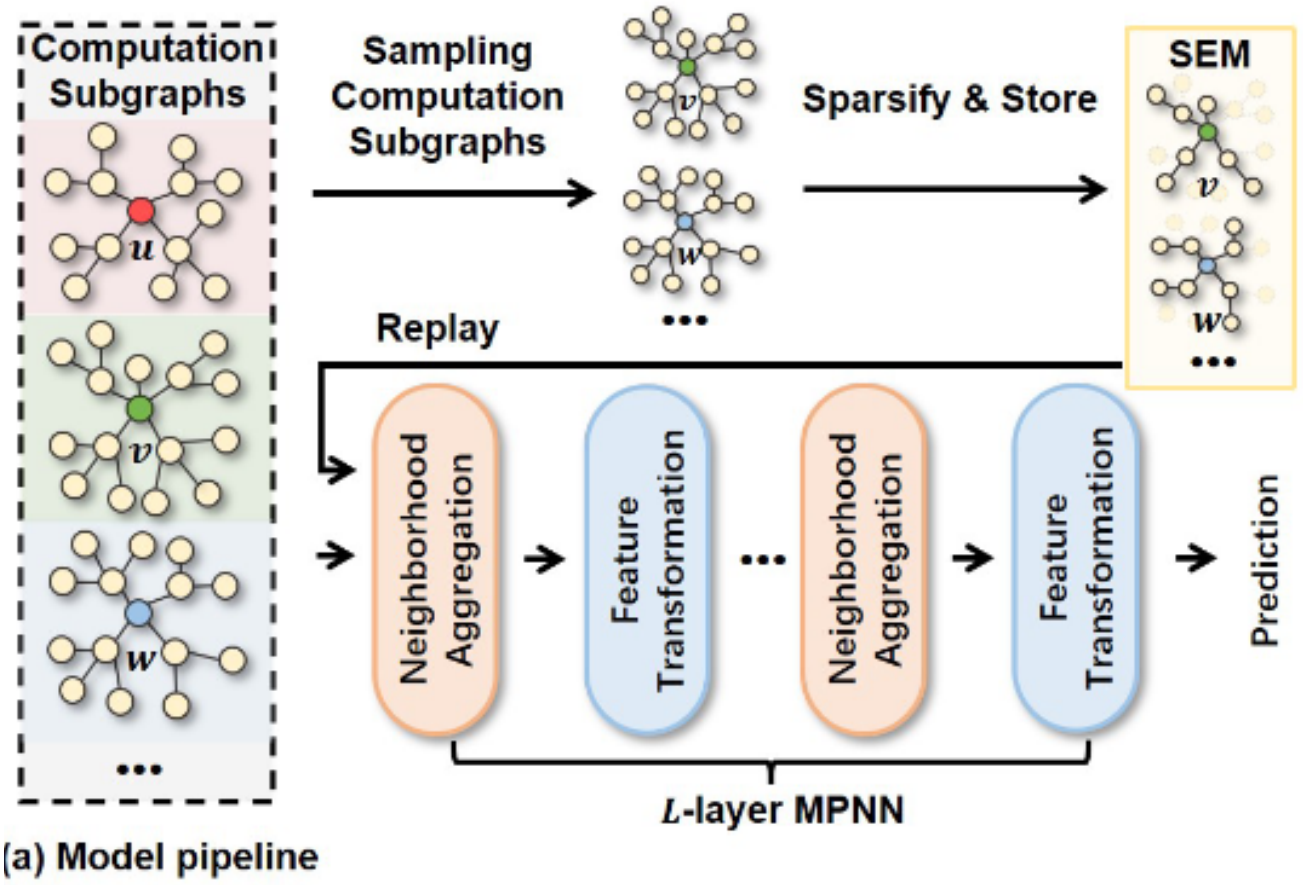
Sparsified Subgraph Memory (SSM, memory replay based)



- Sparsify hop by hop
- First selected a fixed number of 1-hop neighbors
- Then selected a fixed number of 2-hop neighbors (must be connected to selected 1-hop neighbors)
- Any sampling strategy can be used

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Sparsified Subgraph Memory for Continual Graph Representation Learning." *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2022.

Sparsified Subgraph Memory (SSM, memory replay based)



(a) Model pipeline

- Each node has a computation subgraph, size is $O(d^L)$
- Selected a fixed number of computations to store
- Sparsify each selected computation subgraph into a fixed number of nodes, size reduces to $O(1)$
- Plug and play, easy to use in any GNNs

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Sparsified Subgraph Memory for Continual Graph Representation Learning." *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2022.

Sparsified Subgraph Memory (SSM, memory replay based)

Continual learning techniques	CoraFull		OGB-Arxiv		Reddit		OGB-Products	
	AA/% ↑	AF/% ↑	AA/% ↑	AF /% ↑	AA/% ↑	AF /% ↑	AA/% ↑	AF /% ↑
Fine-tune	3.5±0.5	-95.2±0.5	4.9±0.0	-89.7±0.4	5.9±1.2	-97.9±3.3	3.4±0.8	-82.5±0.8
EWC [19]	52.6±8.2	-38.5±12.1	8.5±1.0	-69.5±8.0	10.3±11.6	-33.2±26.1	23.8±3.8	-21.7±7.5
MAS [2]	12.3±3.8	-83.7±4.1	4.9±0.0	-86.8±0.6	13.1±2.6	-35.2±3.5	16.7±4.8	-57.0±31.9
GEM [7]	8.4±1.1	-88.4±1.4	4.9±0.0	-89.8±0.3	28.4±3.5	-71.9±4.2	5.5±0.7	-84.3±0.9
TWP [27]	62.6±2.2	-30.6±4.3	6.7±1.5	-50.6±13.2	13.5±2.6	-89.7±2.7	14.1±4.0	-11.4±2.0
LwF [20]	33.4±1.6	-59.6±2.2	9.9±12.1	-43.6±11.9	86.6±1.1	-9.2±1.1	48.2±1.6	-18.6±1.6
ER-GNN [9]	34.5±4.4	-61.6±4.3	30.3±1.5	-54.0±1.3	88.5±2.3	-10.8±2.4	56.7±0.3	-33.3±0.5
Joint (Not under continual setting)	81.2±0.4	-3.3±0.8	51.3±0.5	-6.7±0.5	97.1±0.1	-0.7±0.1	71.5±0.1	-5.8±0.3
SEM-uniform (Ours)	73.0±0.3	-14.8±0.5	47.1±0.5	-11.7±1.5	94.3±0.1	-1.4±0.1	62.0±1.6	-9.9±1.3
SEM-degree (Ours)	**75.4±0.1**	-9.7±0.0	**48.3±0.5**	-10.7±0.3	**94.4±0.0**	-1.3±0.0	**63.3±0.1**	-9.6±0.3

Comparison under the challenging class-IL

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Sparsified Subgraph Memory for Continual Graph Representation Learning." *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2022.

Rough Categories

	Pros:	Cons:
• **Regularization:** Penalize changes to model parameters via regularizations	• Good stability on old tasks • Suitable for different architectures	• Poor plasticity for new tasks
• **Parameter-isolation:** Separate parameters for new and old tasks (partially or entirely)	• Good stability and plasticity	• Memory usage • May not fit different architectures • Computation burden
• **Memory-replay:** Replay old task data to the model when learning new ones	• Good stability and plasticity • Suitable for different architectures	• Memory usage • Computation burden

Representative CL techniques

Agenda

- Background on Graph Representation Learning
- Motivation of Continual Graph Learning
- Problem setup of CL and CGL
- CL techniques & CGL techniques
- Evaluation Metrics & CGL benchmarks
- Future directions

Evaluation metrics

Performance Matrix: model's performance (accuracy, f1 score, etc.) after learning each task, the most thorough metric

$$M^p \in R^{T \times \mathrm{T}}$$

T: number of tasks in the sequence

$M^p_{i,j}$: Performance on task j after learning from task 1 to i

Evaluation metrics

Performance Matrix: model's performance (accuracy, f1 score, etc.) after learning each task, the most thorough metric

$$M^p \in R^{T \times \mathrm{T}}$$

T: number of tasks in the sequence

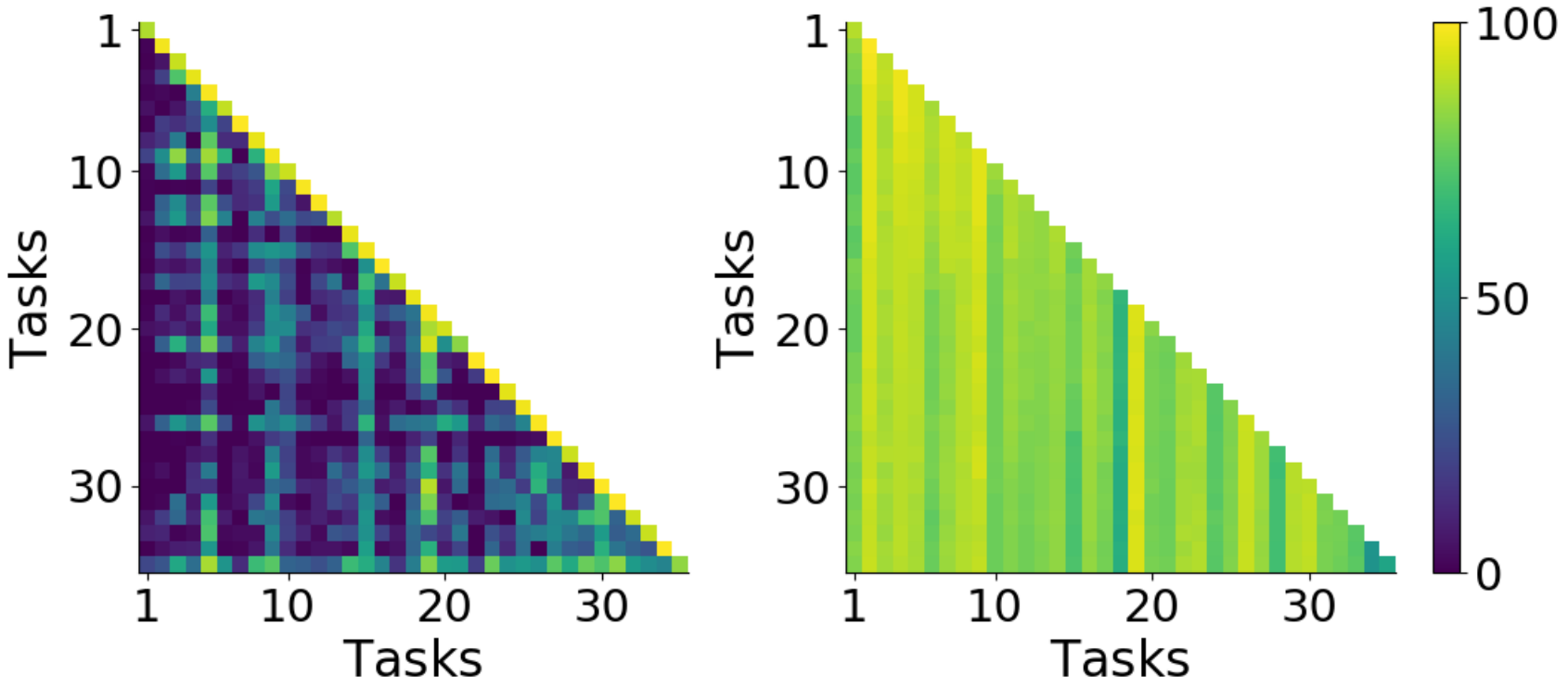
$M^p_{i,j}$: Performance on task j after learning from task 1 to i

An example

	T1	T2	T3
T1	100%		
T2	90%	98%	
T3	60%	70%	99%

Evaluation metrics

Can be visualized



Evaluation metrics

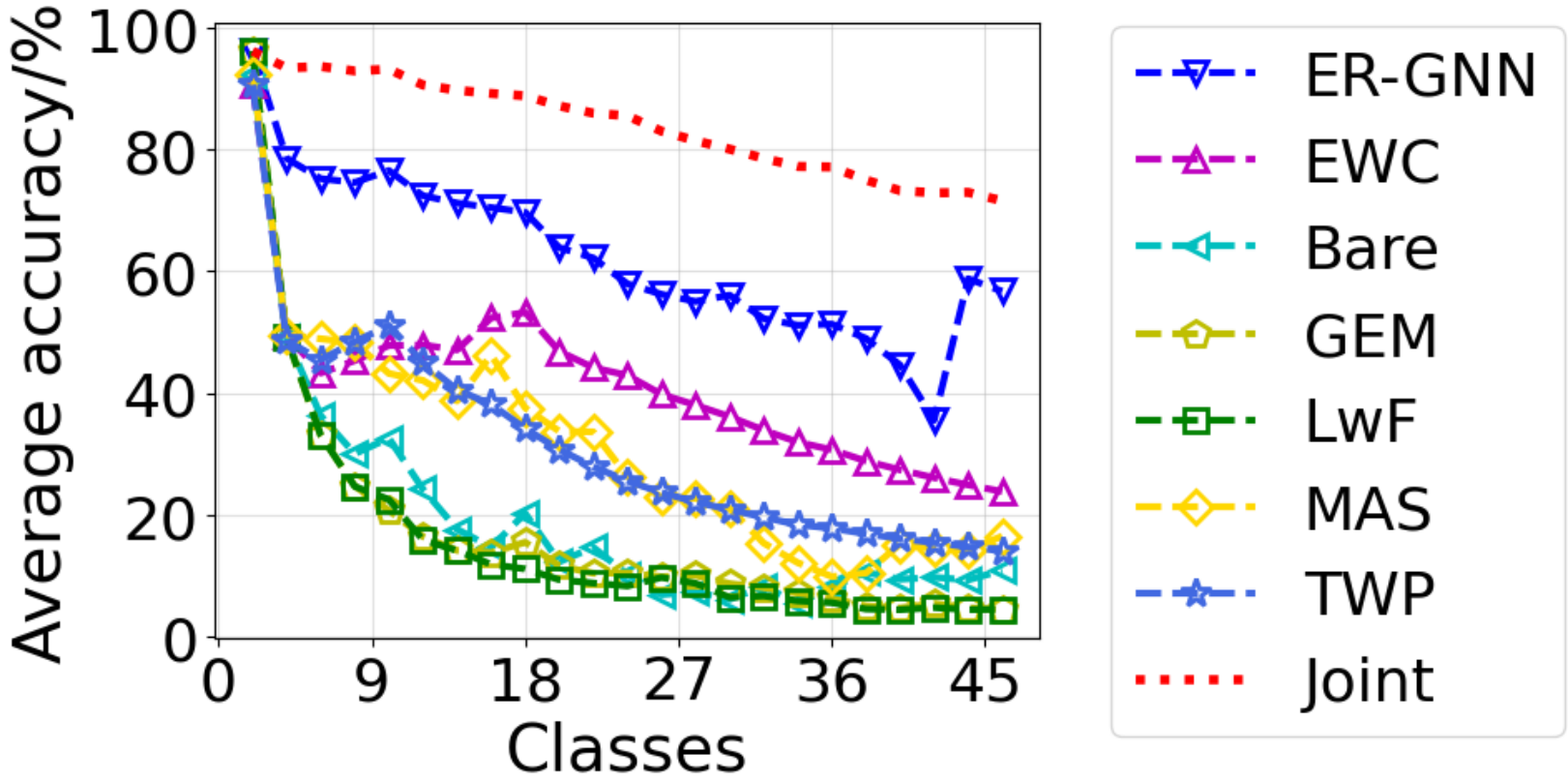
Average Performance (AP): Performance (e.g. accuracy, f1 score) averaged over currently learnt tasks, applicable after learning each task:

$$\left\{ \frac{\sum_{j=1}^{i} \mathrm{M}_{i,j}^{p}}{i} \middle| i = 1, \ldots, T \right\}$$

	T1	T2	T3
T1	100%		
T2	90%	98%	
T3	60%	70%	99%

$$\frac{(60\%+70\%+99\%)}{3}=76.3\%$$

Learning dynamics (curve of AP)



Evaluation metrics

Average Forgetting (AF): Performance decrease averaged over all learnt tasks, applicable from the second task

$$\left\{ \frac{\sum_{j=1}^{i-1} \mathrm{M}_{i,j}^{p} - \mathrm{M}_{j,j}^{p}}{i-1} \middle| i = 2, \ldots, T \right\}$$

	T1	T2	T3
T1	100%		
T2	90%	98%	
T3	60%	70%	99%

$$\frac{(60\%-100\%)+(70\%-98\%)}{3-1}=-29\%$$

Average Forgetting (AF) another definition: Performance decrease averaged over all learnt tasks, applicable from the second task

$$f_j = \max_{i=1,\ldots,T-1} \mathrm{M}^p_{i,j} - \mathrm{M}^p_{T,j}$$

Final AF $\frac{\sum_{j=1}^{T-1} f_j}{T-1}$ **AF after each task** $\left\{ \frac{\sum_{j=1}^{i-1} f_j}{i-1} \middle| i = 2, \ldots, T \right\}$

	T1	T2	T3
T1	75%		
T2	90%	98%	
T3	60%	70%	99%

$$\frac{(60\%-90\%)+(70\%-98\%)}{3-1}=-24\%$$

Forward transfer

Benefit from learning on all the (j-1) tasks before j

performance on task j with random model initialization

$$\frac{\sum_{j=2}^{T} \mathrm{M}_{j-1,j}^{p} - \bar{b}_j}{T-1}$$

CGLB: Benchmark Tasks for Continual Graph Learning

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Cglb: Benchmark tasks for continual graph learning." *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022.

Benchmark tasks

CGLB
Continual Graph Learning Benchmark

N-CGL

G-CGL

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Cglb: Benchmark tasks for continual graph learning." *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022.

Benchmark tasks

CGLB
Continual Graph Learning Benchmark

N-CGL

Datasets for Task-IL & Class-IL

CoraFull-CL: Classification on new classes of articles

Arxiv-CL: Classification on new classes of articles

Reddit-CL: Classification on new classes of communities

Products-CL: Classification on new classes of products

G-CGL

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Cglb: Benchmark tasks for continual graph learning." *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022.

Benchmark tasks

CGLB
Continual Graph Learning Benchmark

N-CGL

Datasets for Task-IL & Class-IL

CoraFull-CL: Classification on new classes of articles

Arxiv-CL: Classification on new classes of articles

Reddit-CL: Classification on new classes of communities

Products-CL: Classification on new classes of products

G-CGL

Datasets for Task-IL

Datasets for Class-IL

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Cglb: Benchmark tasks for continual graph learning." *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022.

Benchmark tasks

CGLB
Continual Graph Learning Benchmark

N-CGL

Datasets for Task-IL & Class-IL

CoraFull-CL: Classification on new classes of articles

Arxiv-CL: Classification on new classes of articles

Reddit-CL: Classification on new classes of communities

Products-CL: Classification on new classes of products

G-CGL

Datasets for Task-IL

SIDER-tIL: Prediction on new molecule properties

Tox21-tIL: Prediction on new molecule properties

Aromaticity-CL: Classification on new classes of molecules

Datasets for Class-IL

Aromaticity-CL: Classification on new classes of molecules

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Cglb: Benchmark tasks for continual graph learning." *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022.

Benchmark tasks

Continual Graph Learning Benchmark

Table 1: The detailed statistics of the constructed benchmark datasets for N-CGL.

Benchmark datasets	CoraFull-CL	Arxiv-CL	Reddit-CL	Products-CL
Data source	CoraFull [34]	OGB-Arxiv[1]	Reddit [17]	OGB-Products[2]
Learning scenario	task-IL & class-IL	task-IL & class-IL	task-IL & class-IL	task-IL & class-IL
# nodes	19,793	169,343	227,853	2,449,028
# edges	130,622	1,166,243	114,615,892	61,859,036
# classes	70	40	40	46
# tasks	35	20	20	23
average # nodes per task	660	8,467	11,393	122,451
average # edges per task	4,354	58,312	5,730,794	2,689,523

CGL Benchmarks

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Cglb: Benchmark tasks for continual graph learning." *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022.

Benchmark tasks

CGLB
Continual Graph Learning Benchmark

Table 2: The detailed statistics of the constructed benchmark datasets for G-CGL.

Benchmark datasets	SIDER-tIL	Aromaticity-CL	Tox21-tIL
Data source	SIDER [53]	PubChemBioAssayAromaticity [54]	Tox21 [3]
Learning scenario	task-IL	task-IL & class-IL	task-IL
# graphs	1,427	3,868	7,831
# nodes	48,006	115,061	145,459
# edges	100,912	253,018	302,190
# classes	27	30	12
# tasks	27	15	12
average # graphs per task	53	155	653
average # nodes per task	1,778	7,671	12,122
average # edges per task	3,737	16,868	25,183

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Cglb: Benchmark tasks for continual graph learning." *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022.

CGLB
Continual Graph Learning Benchmark

Benchmark results

1. Task-IL & Class-IL
2. Node-level & Graph-level
3. W/WO Inter-task edges
4. Comparisons among different baselines
5. Visualization results

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Cglb: Benchmark tasks for continual graph learning." *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022.

Benchmark tasks

CGLB
Continual Graph Learning Benchmark

Toolkit

1. Continual learning pipelines
2. Implementations of different baselines
3. Visualization tools

https://github.com/QueuQ/CGLB

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "Cglb: Benchmark tasks for continual graph learning." *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022.

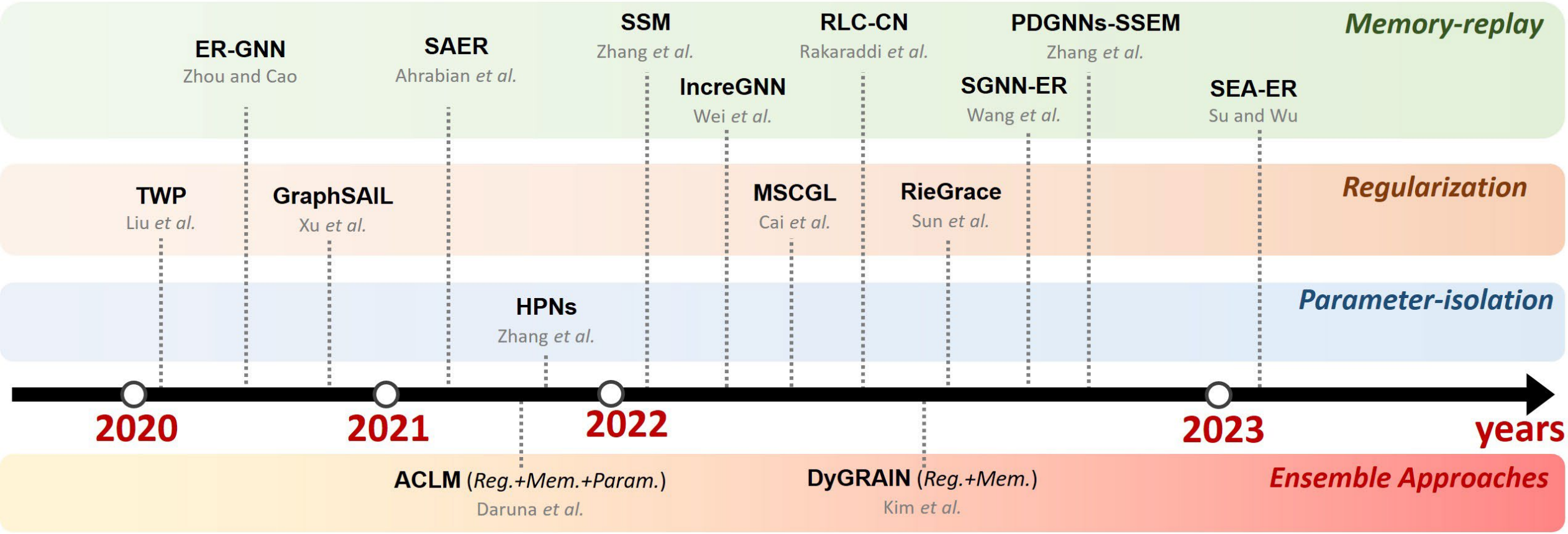
Benchmark tasks

Continual Learning on Graphs: Challenges, Solutions, and Opportunities

Method	Applications	Task Granularity	Technique	Characteristics
TWP [1]	General	Node/Graph	Reg.	Preserve the topology learnt from previous tasks
RieGrace [30]	General	Node	Reg.	Maintain previous knowledge via knowledge distillation
GraphSAIL [31]	Recommender Systems	Node	Reg.	Local and global structure preservation, node information preservation
MSCGL [32]	General	Node	Reg.	Parameter changes orthogonal to previous parameters
ER-GNN [33]	General	Node	Mem.	Replay representative nodes
SSM [34]	General	Node	Mem.	Replay representative sparsified computation subgraphs
PDGNNs-SSEM [35]	General	Node	Mem.	Replay representative sufficient subgraph embeddings
IncreGNN [36]	General	Node	Mem.	Replay nodes according to their influence
RLC-CN [37]	General	Node	Mem.	Model structure adaption and dark experience replay
SGNN-ER [38]	General	Node	Mem.	Model retraining with generated fake historical data
SAER [39]	Recommender System	Node	Mem.	Buffer the representative user-item pairs based on reservoir sampling
SEA-ER [40]	General	Node	Mem.	Minimize the structural difference between the memory buffer and the original graph
HPNs [2]	General	Node	Para.	Extracting and storing basic features to encourage knowledge sharing across tasks, model expanding to accommodate new patterns
DyGRAIN [41]	General	Node	Mem.+Reg.	Alleviate catastrophic forgetting and concept shift of previous task nodes via memory replay and knowledge distillation
ACLM [42]	Knowledge Graph	Node	Mem.+Reg.+Para.	Adapting general CL techniques to CGL tasks

CGLB
Continual Graph Learning Benchmark

Continual Learning on Graphs: Challenges, Solutions, and Opportunities



Catastrophic Forgetting in Deep Graph Networks: an Introductory Benchmark for Graph Classification

	MNIST	CIFAR10	OGBG-PPA
Size	70000	60000	158100
Node Attrs.	3	5	0
Edge Attrs.	0	0	7
Classes	10	10	37
Avg $\lvert\mathcal{V}_g\rvert$	70,57	117,63	243,4
Avg $\lvert\mathcal{E}_g\rvert$	564,63	941,07	2266,1
Data Split	55K/5K/15K	45K/5K/15K	49%/29%/22%
Class Split	2+2+2+2+2	2+2+2+2+2	17+5+5+5+5

Carta, Antonio, et al. "Catastrophic forgetting in deep graph networks: an introductory benchmark for graph classification." *arXiv preprint arXiv:2103.11750* (2021).

Agenda

- Background on Graph Representation Learning
- Motivation of Continual Graph Learning
- Problem setup of CL and CGL
- CL techniques & CGL techniques
- Evaluation Metrics & CGL benchmarks
- Future directions

Remaining challenges & Future directions

- **Trade-off between Effectiveness and Space Complexity**
- **Dependency on the Task Boundaries**
- **Extension to other Modalities (Heterogeneous graphs)**
- **Extension to more graph related tasks**
- **Into large models, on large graphs**

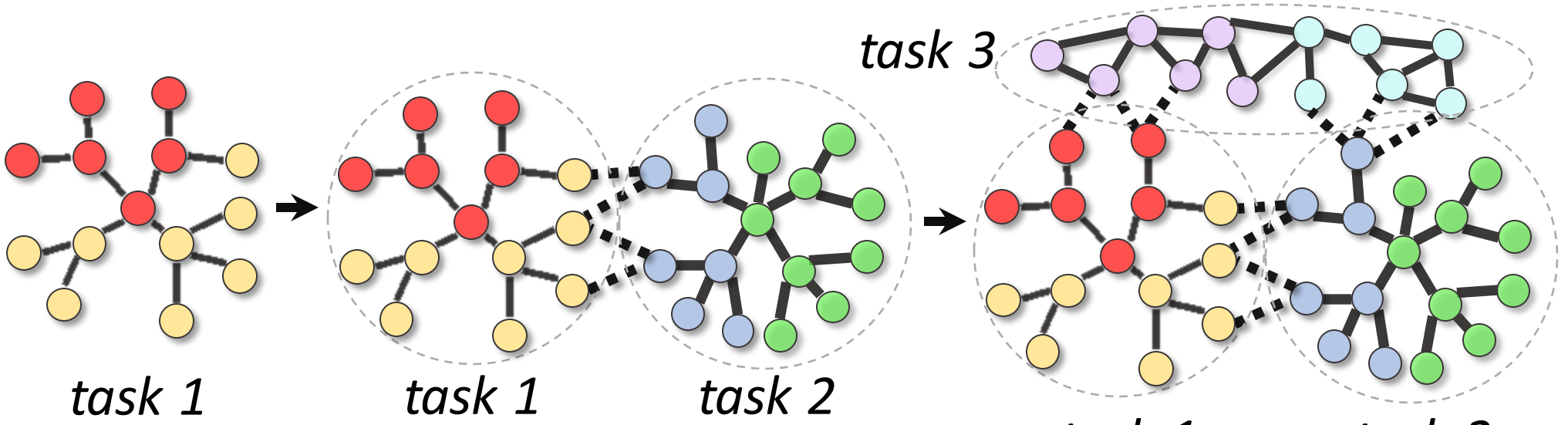
Remaining challenges & Future directions

Trade-off between Effectiveness and Space/Computation Complexity

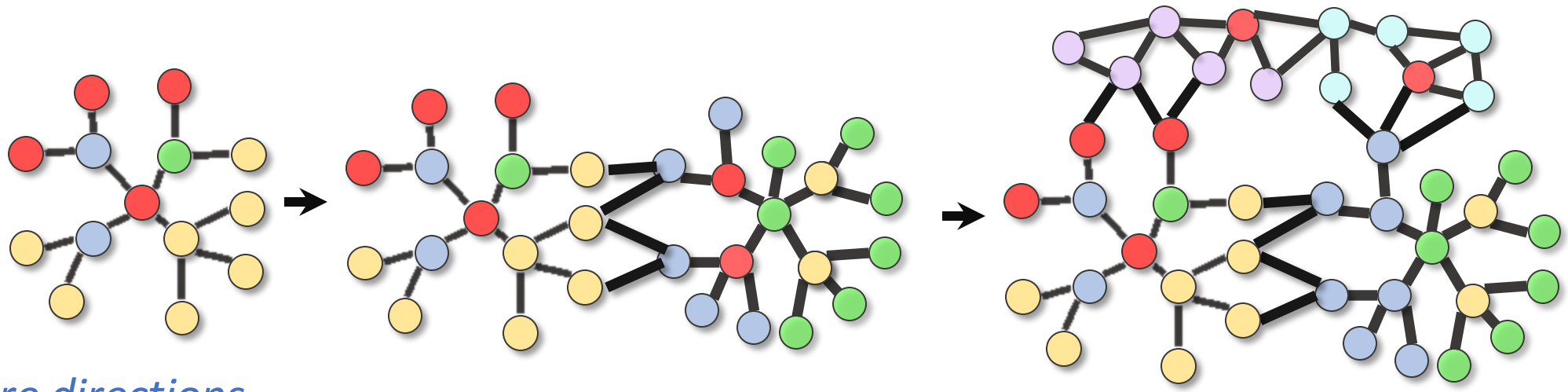
- **Regularization:** No extra memory buffer. Poor plasticity for new tasks.
- **Parameter-isolation:** Extra memory consumption for network expansion. Good performance.
- **Memory-replay:** Extra memory consumption for representative data. Good performance.

Remaining challenges & Future directions

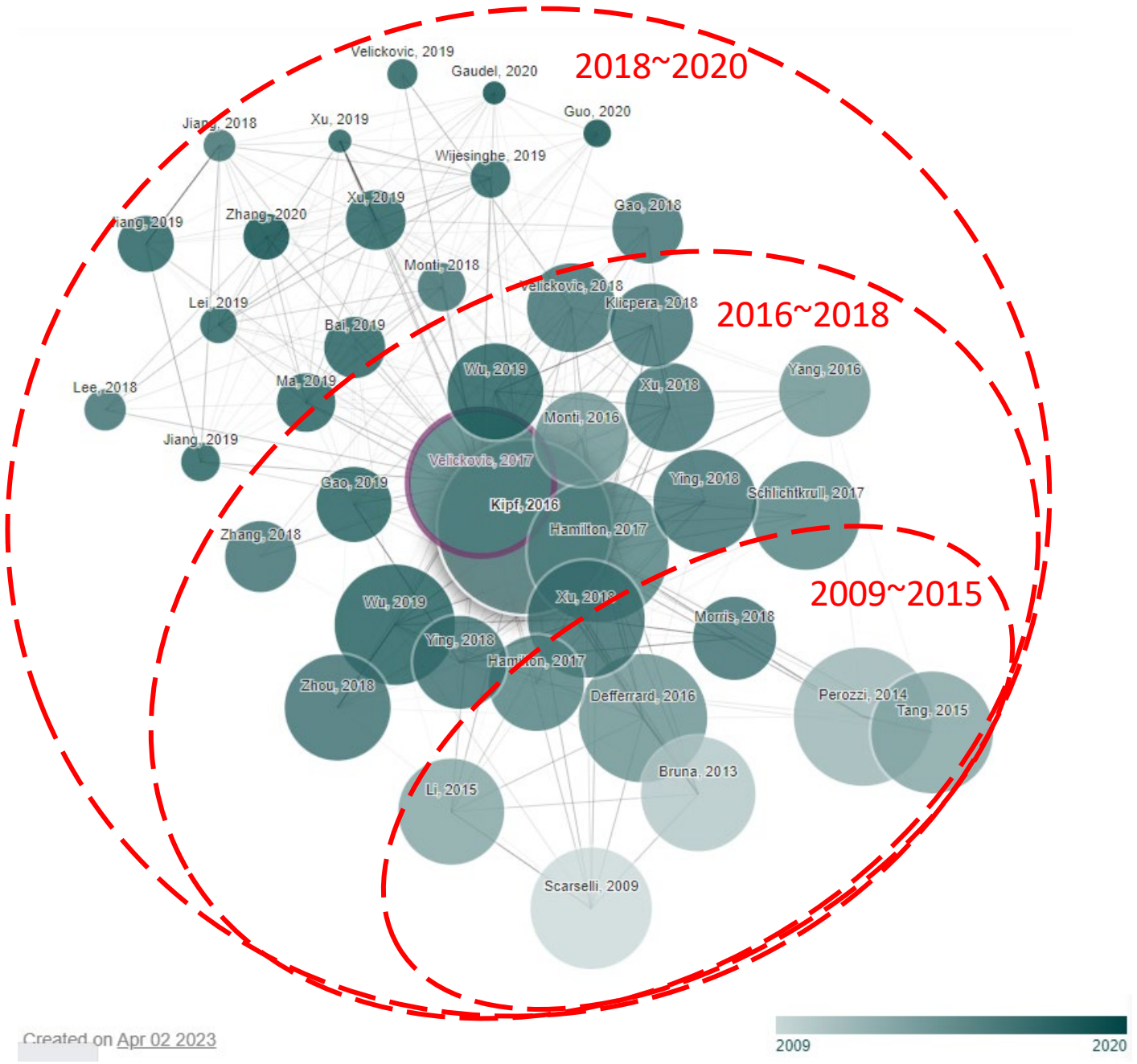
Dependency on the Task Boundaries



Explicit Task Boundaries



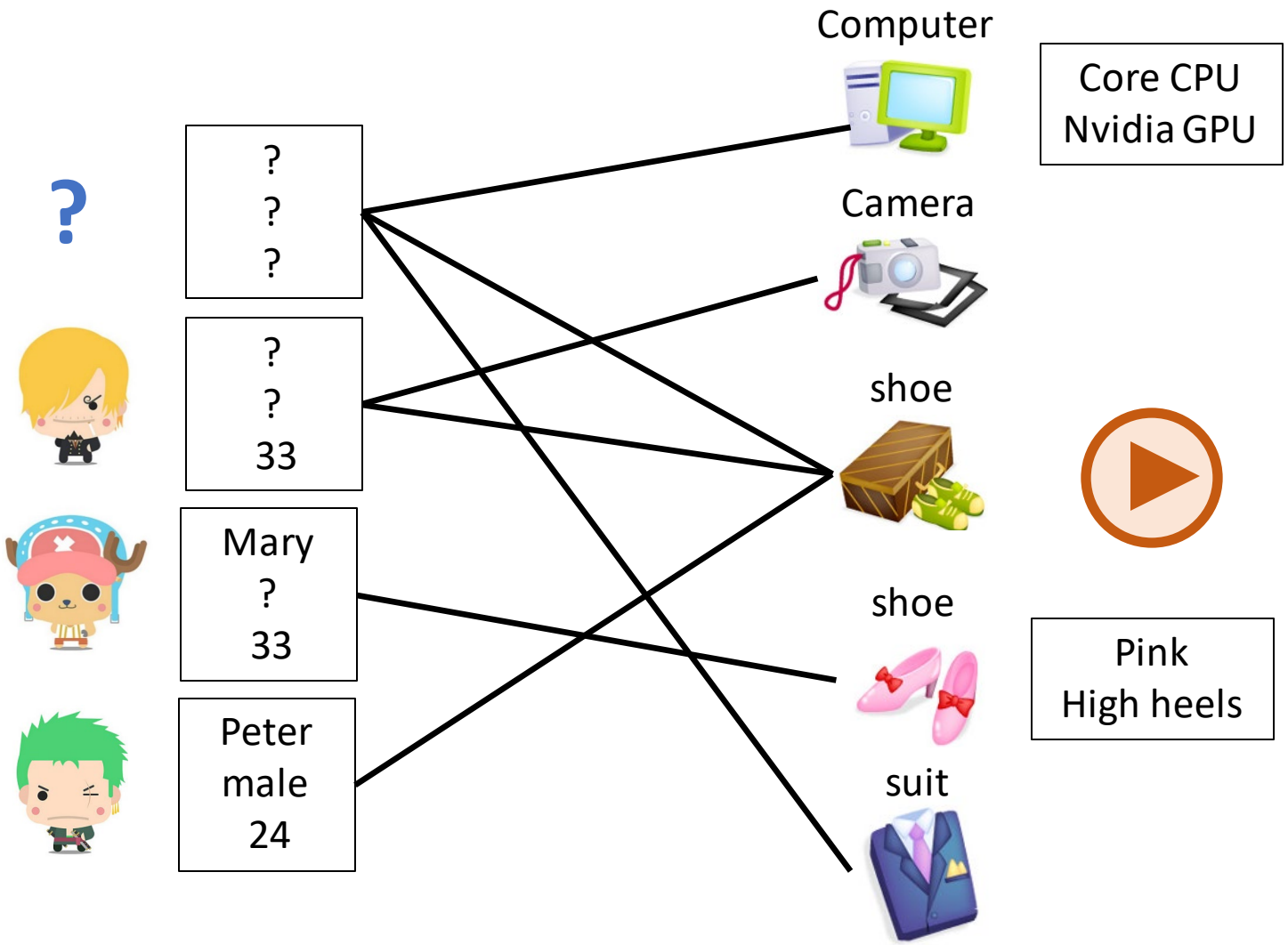
Implicit Task Boundaries



- Research may focus on different topics in different years.
- Papers on new and old topics are blended without clear boundaries.
- When to treat the new data as new tasks?
- Techniques from other areas may help!

Remaining challenges & Future directions

Extension to other Modalities (Heterogeneous Graphs)

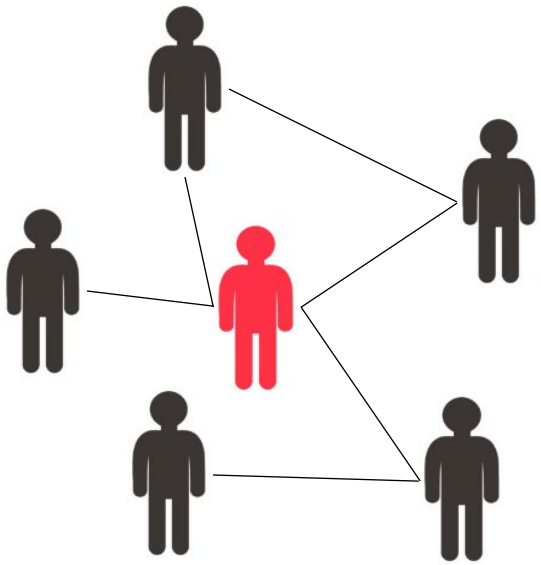


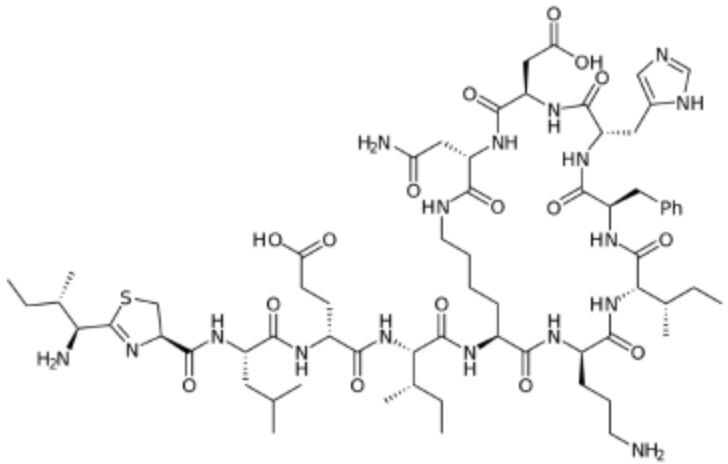
- **Different Modalities**
- **Different number of modalities**
- **Can learning on one modality benefit the others?**

Remaining challenges & Future directions

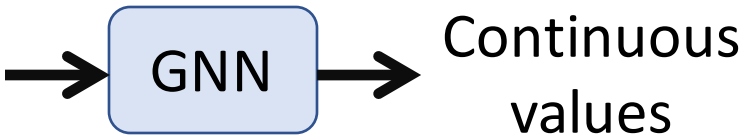
Extension to more graph related tasks

- **Anomaly detection**
- **Graph/Node regression**
- **Etc.**



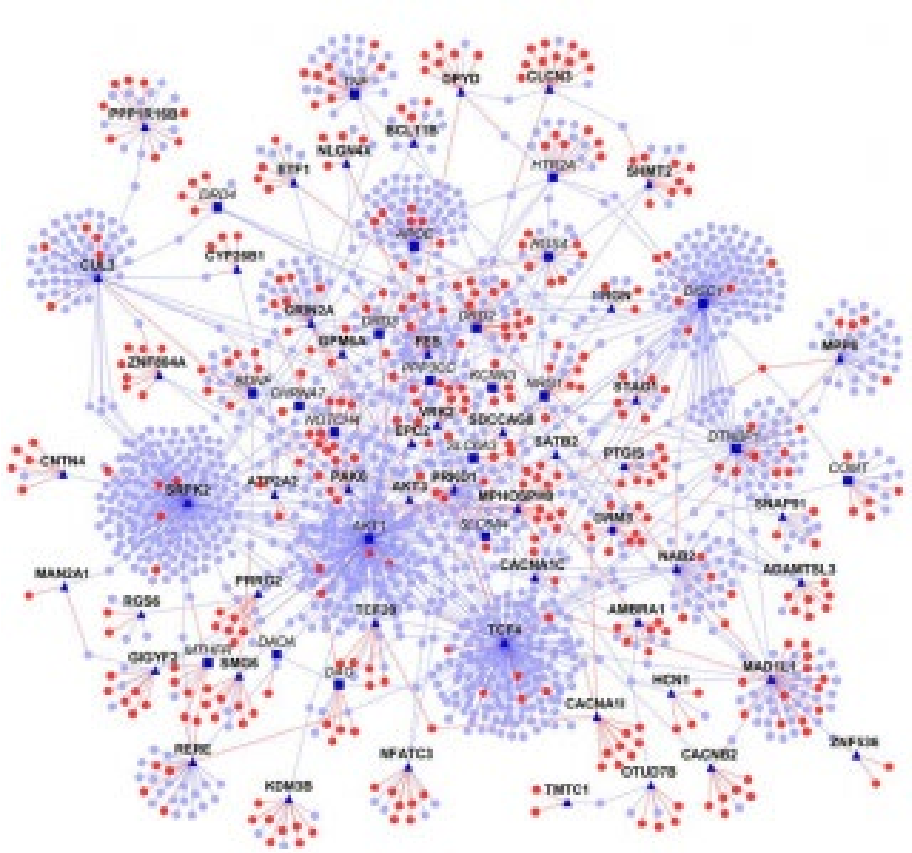


Molecules



Graph data are pervasive in our world

Into large models, on large graphs



- **Graphs could grow to be very large gradually**
- **Models can also grow accordingly**

https://www.genengnews.com/insights/protein-protein-interactions-get-a-new-groove-on/

Further readings

Gilmer, Justin, et al. "Neural message passing for quantum chemistry." International conference on machine learning. PMLR, 2017.
https://proceedings.mlr.press/v70/gilmer17a

For understanding the basic logic of GNNs

Parisi, German I., et al. "Continual lifelong learning with neural networks: A review." Neural networks 113 (2019): 54-71.
https://www.sciencedirect.com/science/article/pii/S0893608019300231

For an overview of continual learning

Van de Ven, Gido M., and Andreas S. Tolias. "Three scenarios for continual learning." arXiv preprint arXiv:1904.07734 (2019).
https://arxiv.org/abs/1904.07734

For an understanding of different incremental scenarios

Zhang, Xikun, Dongjin Song, and Dacheng Tao. "CGLB: Benchmark tasks for continual graph learning." Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track. 2022.
https://openreview.net/forum?id=5wNiiIDynDF

For an overview of CGL settings and challenges