In the name of the absolute power and the absolute knowledge

Social and Cognitive Robotics

Chapter 3+: **Continual Learning**

Alireza Taheri, *PhD, Assistant Professor*
artaheri@sharif.edu

Head of the Social and Cognitive Robotics Lab.,
Center of Excellence in Design, Robotics, and Automation
School of Mechanical Engineering,
Sharif University of Technology, Tehran, Iran

2024

Outlines

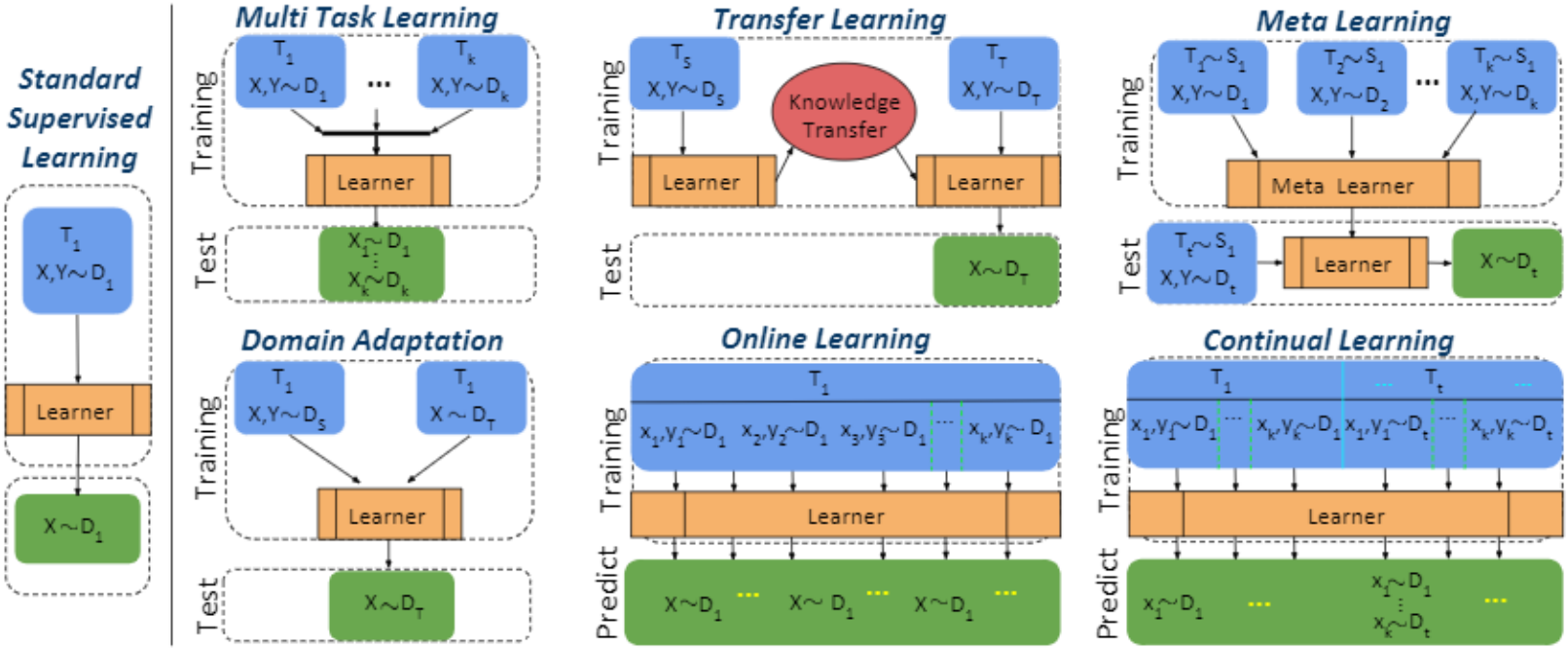
- *Chapter 1:* **Continual Learning**
  - Problem Statement
  - Problem Variations
    - Task Incremental
    - Class Incremental
    - Domain Incremental
    - Task Agnostic
  - Metrics
  - Methods
    - Replay methods
    - Regularization-based methods
    - Parameter Isolation methods
  - Edge of Knowledge
  - Conclusion

Problem Statements

- What is Continual Learning?

In contrast, many real world settings look like:

Problem Statements *(cont.)*



Problem Variations

- Task ID
- Task order
- Discrete/Continuous

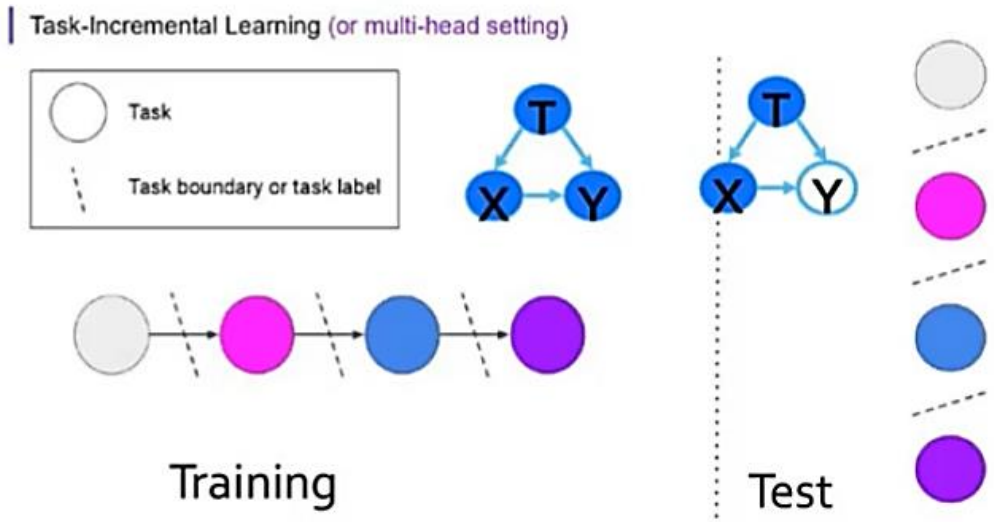
Task Incremental

Class Incremental

Domain Incremental

Problem Variations *(cont.)*

Task Incremental



Task Incremental

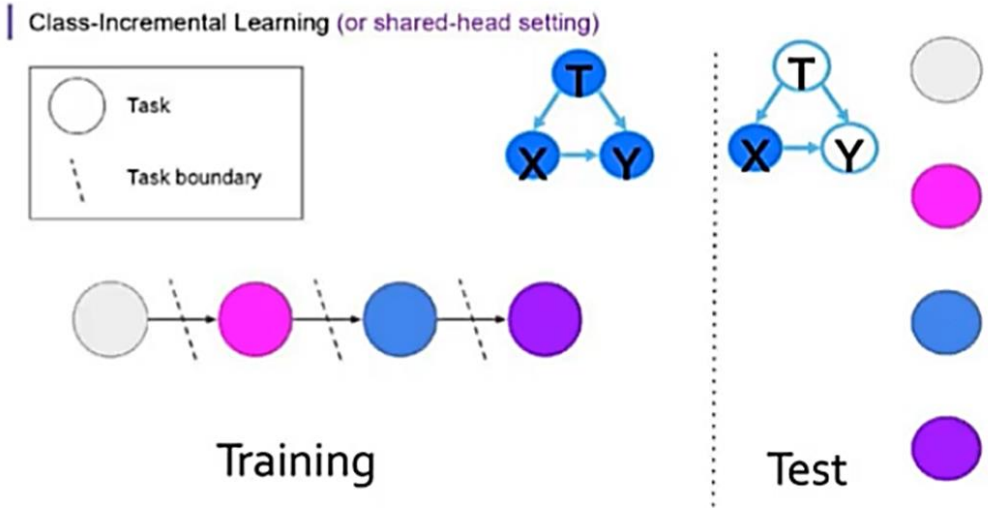
Class Incremental

Domain Incremental

$$\{\mathcal{Y}^{(t)}\} \neq \{\mathcal{Y}^{(t+1)}\}$$

Problem Variations *(cont.)*

Class Incremental



Task Incremental

Class Incremental

Domain Incremental

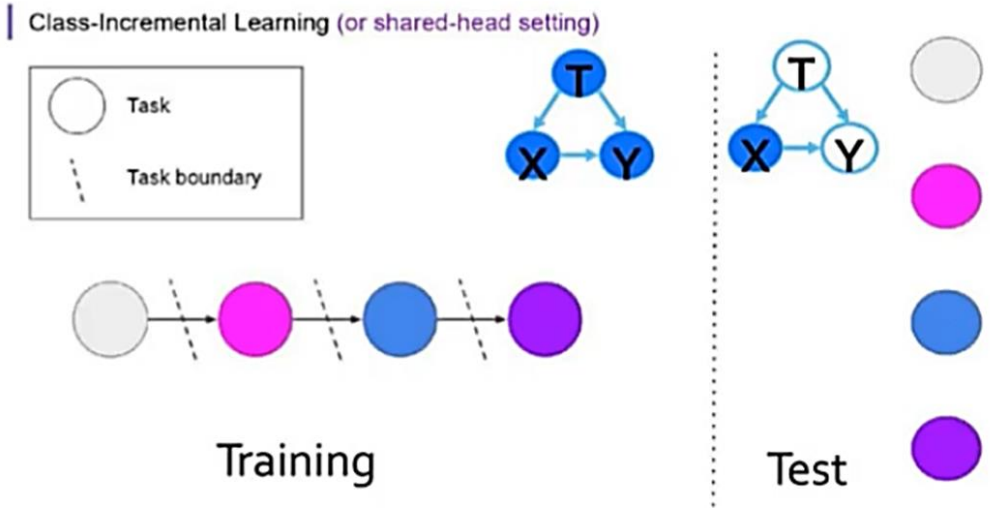
$$P(\mathcal{X}^{(t)}) \neq P(\mathcal{X}^{(t+1)})$$

$$\{\mathcal{Y}^{(t)}\} = \{\mathcal{Y}^{(t+1)}\}$$

$$P(\mathcal{Y}^{(t)}) \neq P(\mathcal{Y}^{(t+1)})$$

Problem Variations *(cont.)*

Domain Incremental



Task Incremental

Class Incremental

Domain Incremental

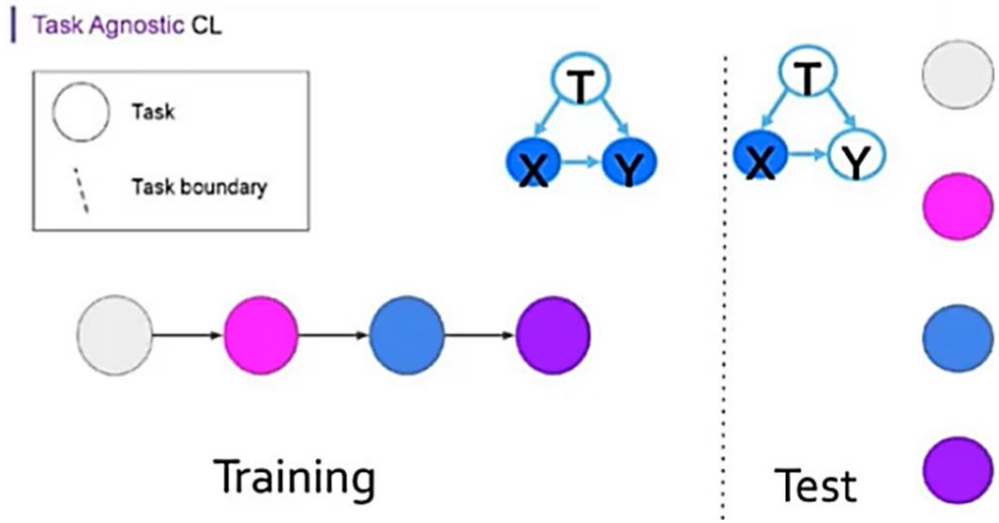
$$P(\mathcal{X}^{(t)}) \neq P(\mathcal{X}^{(t+1)})$$

$$\{\mathcal{Y}^{(t)}\} = \{\mathcal{Y}^{(t+1)}\}$$

$$P(\mathcal{Y}^{(t)}) = P(\mathcal{Y}^{(t+1)})$$

Problem Variations *(cont.)*

Task Agnostic



Task Incremental

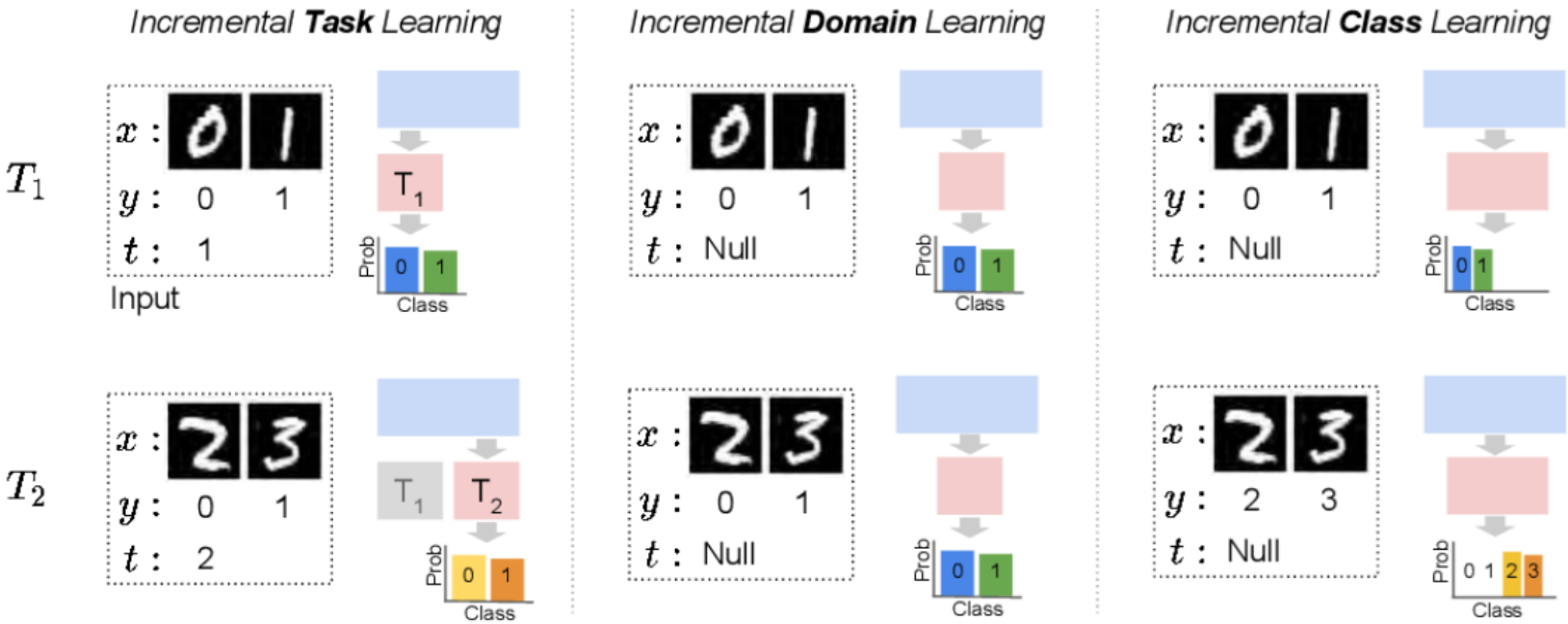
Class Incremental

Domain Incremental

Task Agnostic

Problem Variations *(cont.)*

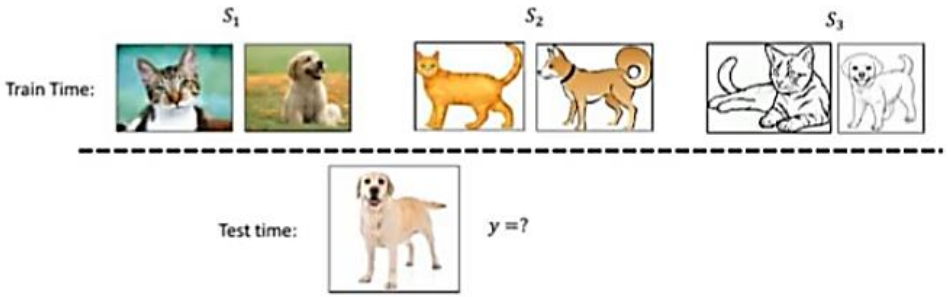
Comparison



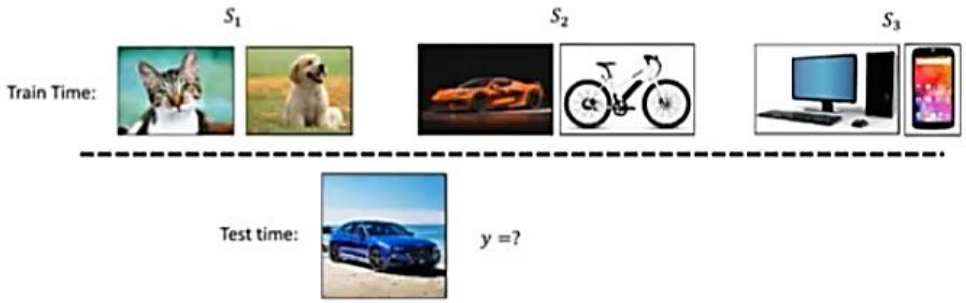
Problem Variations *(cont.)*

Comparison

Domain Incremental



Class Incremental



Metrics

Stability / Plasticity

Accuracy

Forward transfer

Previous tasks cause better performance on future tasks

Backward transfer

Future tasks cause better performance on Previous tasks

Metrics *(cont.)*

- How to calculate metrics?

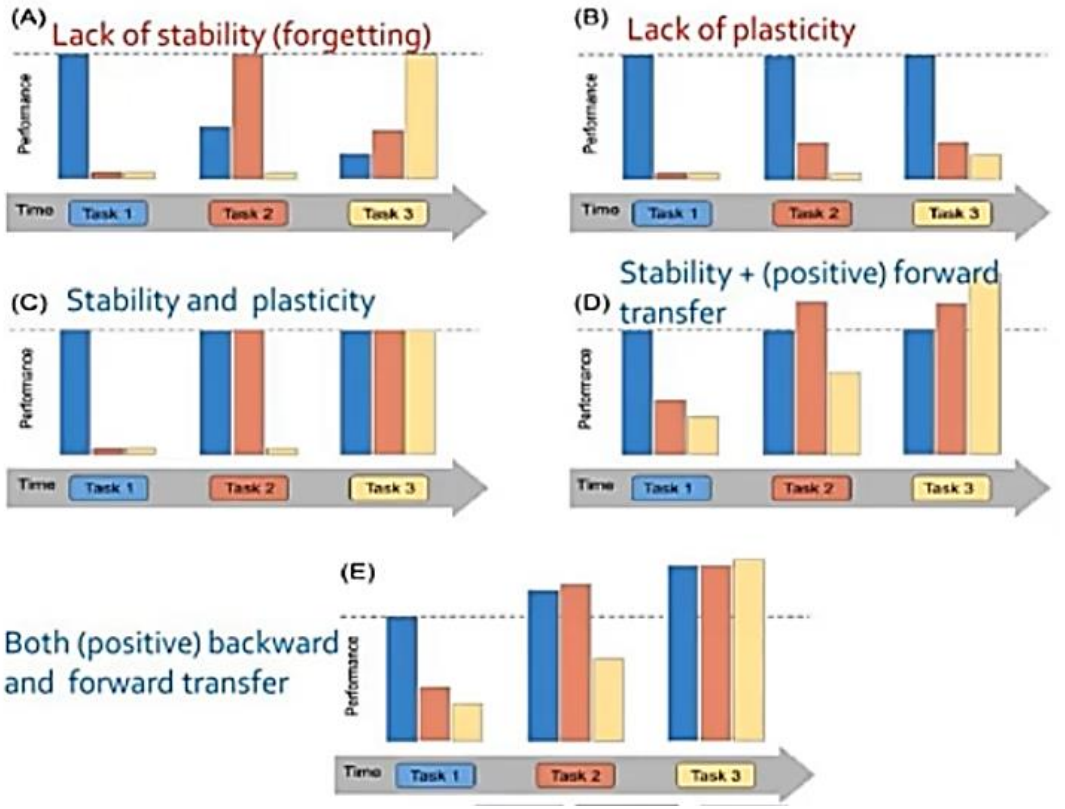
$$\textbf{Average Accuracy: ACC} = \frac{1}{T}\sum_{i=1}^{T} R_{T,i}$$

$$\textbf{Backward Transfer: BWT} = \frac{1}{T-1}\sum_{i=1}^{T-1} R_{T,i} - R_{i,i}$$

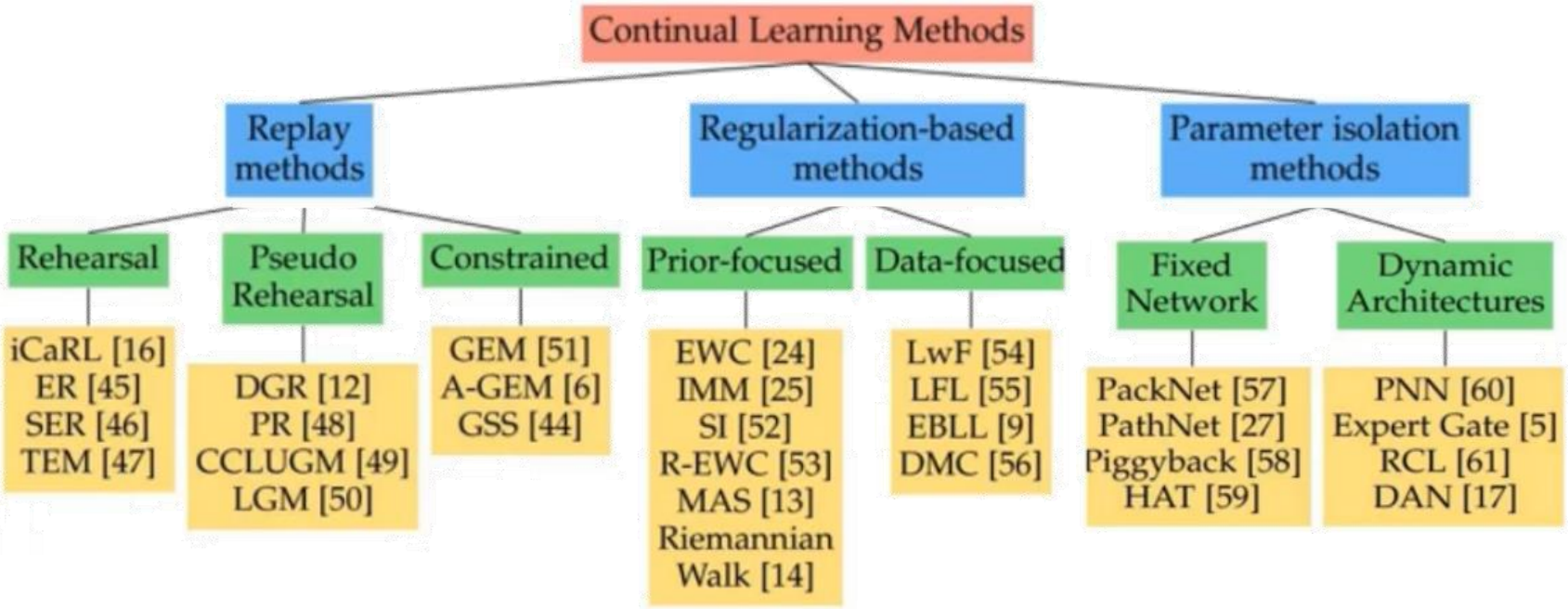
$$\textbf{Forward Transfer: FWT} = \frac{1}{T-1}\sum_{i=2}^{T} R_{i-1,i} - \bar{b}_i.$$

Let's see some example

Metrics *(cont.)*



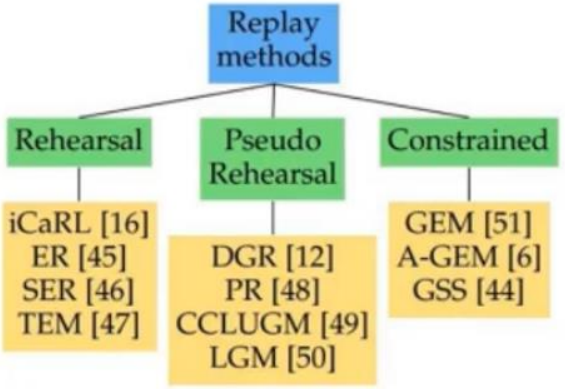
Methods


Continual Learning Methods
Replay methods
Regularization-based methods
Parameter isolation methods
Rehearsal
Pseudo Rehearsal
Constrained
Prior-focused
Data-focused
Fixed Network
Dynamic Architectures
iCaRL [16]
ER [45]
SER [46]
TEM [47]
DGR [12]
PR [48]
CCLUGM [49]
LGM [50]
GEM [51]
A-GEM [6]
GSS [44]
EWC [24]
IMM [25]
SI [52]
R-EWC [53]
MAS [13]
Riemannian Walk [14]
LwF [54]
LFL [55]
EBLL [9]
DMC [56]
PackNet [57]
PathNet [27]
Piggyback [58]
HAT [59]
PNN [60]
Expert Gate [5]
RCL [61]
DAN [17]

Methods *(cont.)*

Replay methods: Rehearsal

Rehearsal

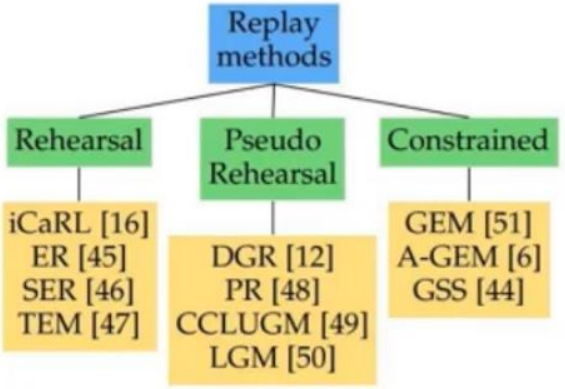


Algorithm 1 Continual learning with Rehearsal.

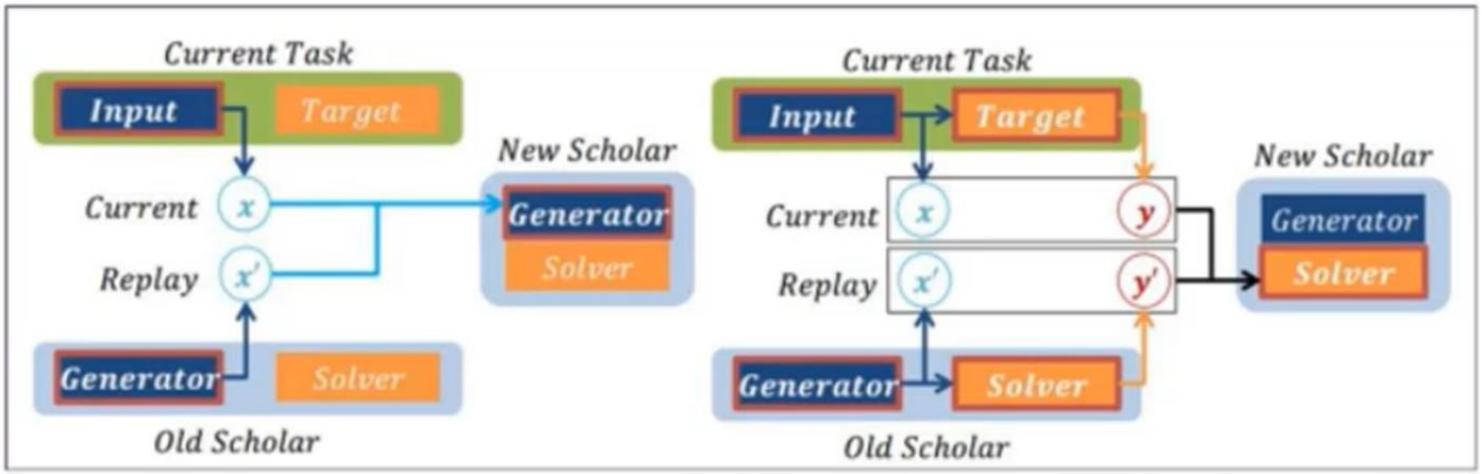
1: **function** REHEARSALBATCH($B, \mathcal{M}$)
2: $\tilde{B} \leftarrow$ RETRIEVALPOLICY($\mathcal{M}$) ▷ Retrieve exemplars
3: $w \leftarrow SGD\left(B \cup \tilde{B}, w\right)$ ▷ Optimize objective for union
4: STORAGEPOLICY($\mathcal{M}, B$) ▷ Update rehearsal memory

Methods *(cont.)*

Replay methods: Pseudo Rehearsal



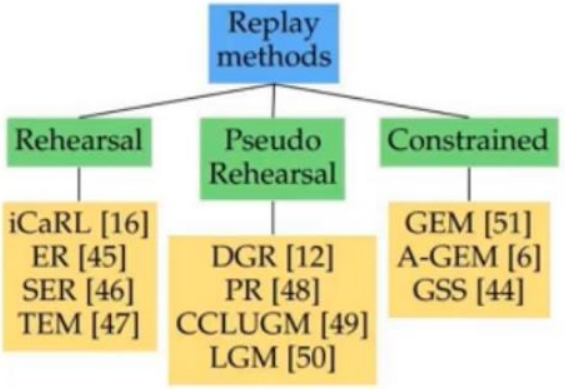
Pseudo-Rehearsal



Methods *(cont.)*

Replay methods: Constrained

Constrained



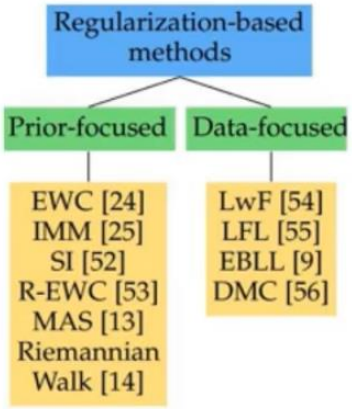
For $t = 0, \ldots, T$

minimize $\mathcal{L}(f_\theta(\cdot, z_t), (x_t, y_t))$

subject to $\mathcal{L}(f_\theta, \mathcal{M}_k) \leq \mathcal{L}(f_\theta^{t-1}, \mathcal{M}_k)$ for all $z_k < z_t$

Methods *(cont.)*

Regularization-based methods: Data Focused



Data focused

LEARNINGWITHOUTFORGETTING:

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:

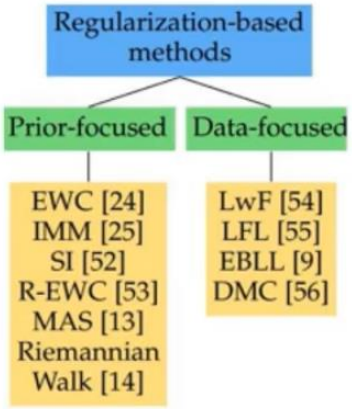
Define $\hat{Y}_o \equiv \text{CNN}(X_n, \hat{\theta}_s, \hat{\theta}_o)$ // *old task output*

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

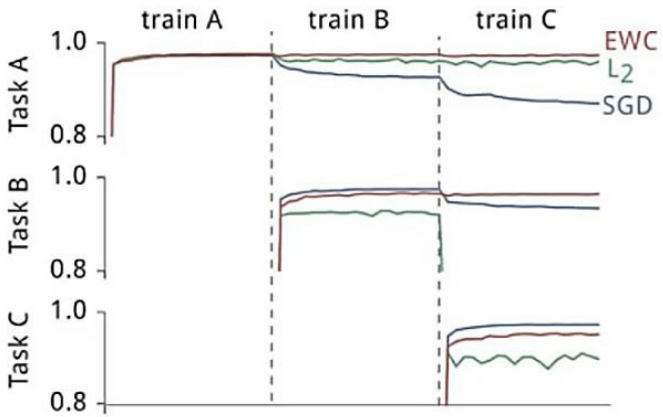
$$\theta_s^*, \theta_o^*, \theta_n^* \leftarrow \underset{\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n}{\text{argmin}} \left(\lambda_o \mathcal{L}_{old}(Y_o, \hat{Y}_o) + \mathcal{L}_{new}(Y_n, \hat{Y}_n) + \mathcal{R}(\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n) \right)$$

Methods *(cont.)*

Regularization-based methods: Prior Focused



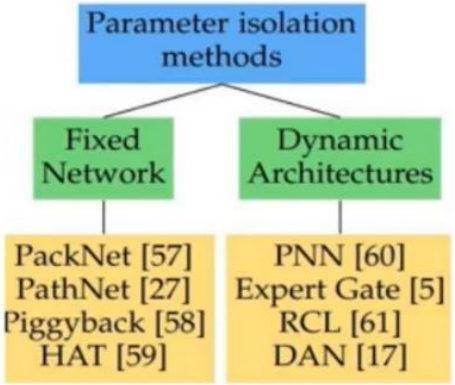
Prior focused



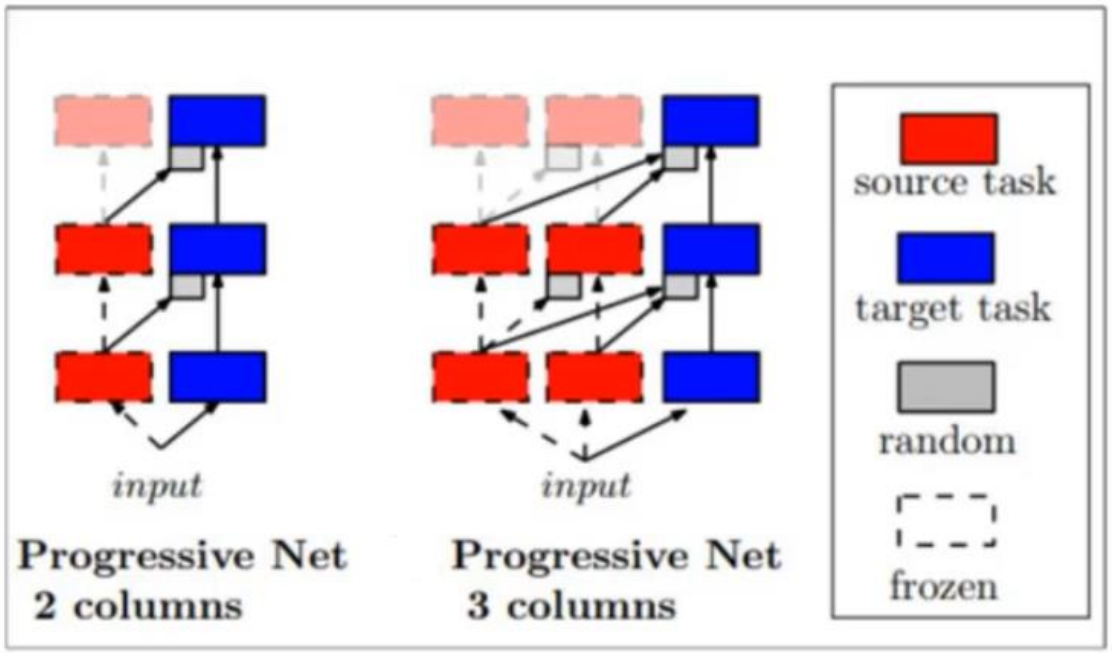
$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum \frac{\lambda}{2} F_i(\theta_i - \theta^*_{A,i})^2$$

Methods *(cont.)*

Parameter Isolation methods: Dynamic Architecture



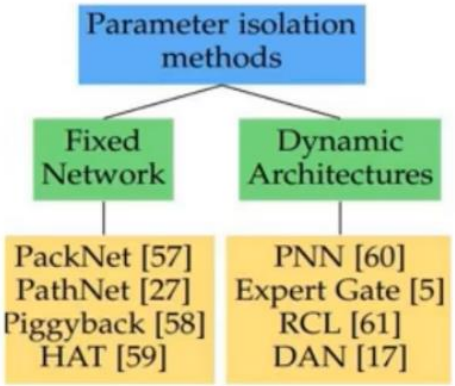
Dynamic Architecture

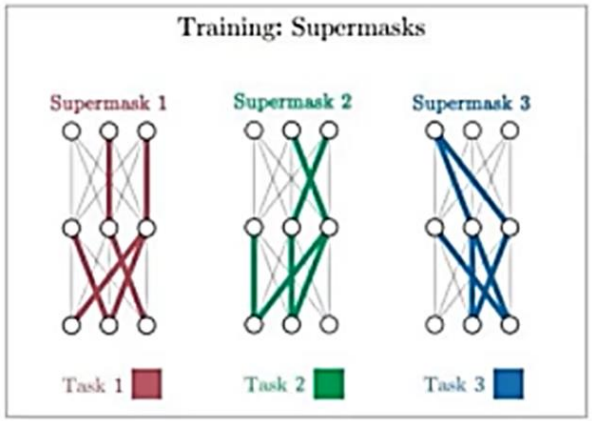


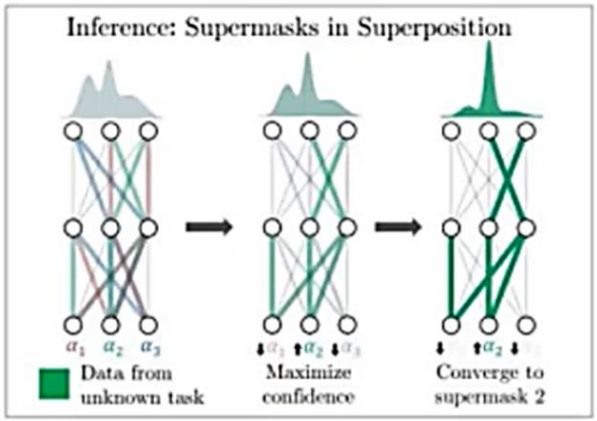
Methods *(cont.)*

Parameter Isolation methods: Fixed Network

Fixed Network







$$\mathbf{p} = f\left(\mathbf{x}, W \odot M^i\right)$$

$$\mathbf{p}(\alpha) = f\left(\mathbf{x}, W \odot \left(\sum_{i=1}^{k} \alpha_i M^i\right)\right)$$

$$\alpha \leftarrow \alpha - \eta \nabla_\alpha \mathcal{H}\left(\mathbf{p}\left(\alpha\right)\right)$$

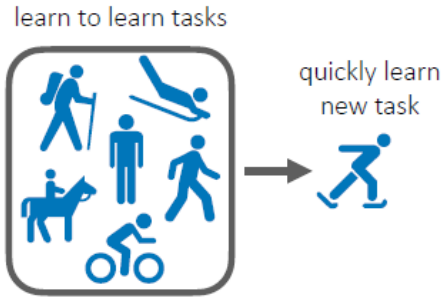
Edge of Knowledge

Remember these

Continual Learning

Meta Learning

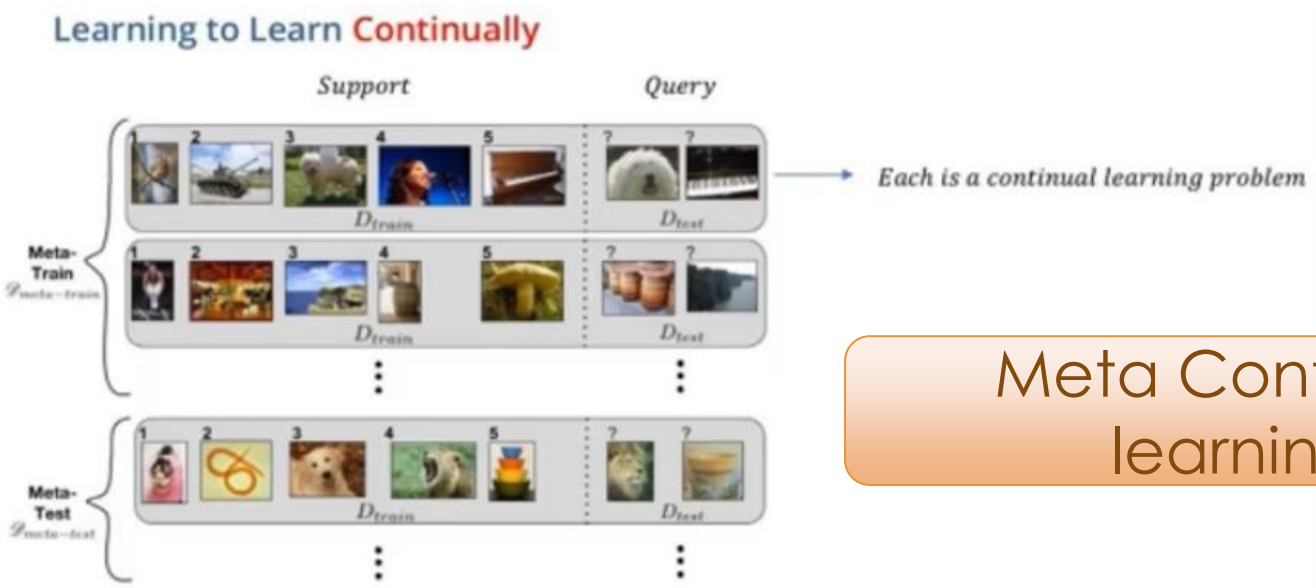
In contrast, many real world settings look like:



Let's combine them

Edge of Knowledge *(cont.)*

Meta Continual Learning

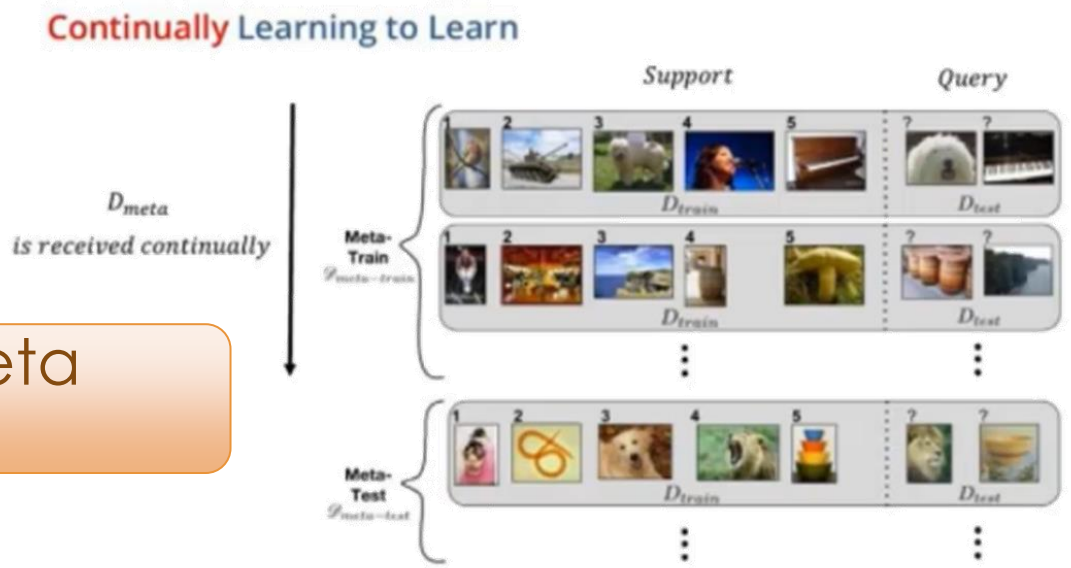


Meta Continual learning

Edge of Knowledge *(cont.)*

Continual Meta Learning

Continual Meta learning



Conclusion

- Where to use Continual Learning?
  - Constant memory
  - No task boundary info
  - Online Learning
  - Forward transfer
  - Backward transfer
  - Problem agnostic
  - Adaptively learning from partial data
  - No test time oracle
  - Task revisiting to strengthen prior knowledge
  - Graceful forgetting to balance stability and plasticity

CEDRA
Center of Excellence in Design, Robotics & Automation

Thanks for your attention