Efficient Training

EECE695D: Efficient ML Systems

Spring 2025

Recap

- **W1–4.** Inference Efficiency
  - Making large models smaller (Sparsity, Low-Precision)
  - Building small models (NAS)
  - Training small models well (Knowledge Distillation)

- **W5–.** Training Efficiency

Overview

- Many ways to achieve efficiency
  - **Optimizer.** SGD, Adam, Shampoo, Muon, …
  - **Initialization.** Xavier, Kaiming, Orthogonal, …
  - **Shared Knowledge** ← **W5-6**
  - **Parallelism.** (W8)
  - **Data.** (W9)
  - **Matmul.** (W12)

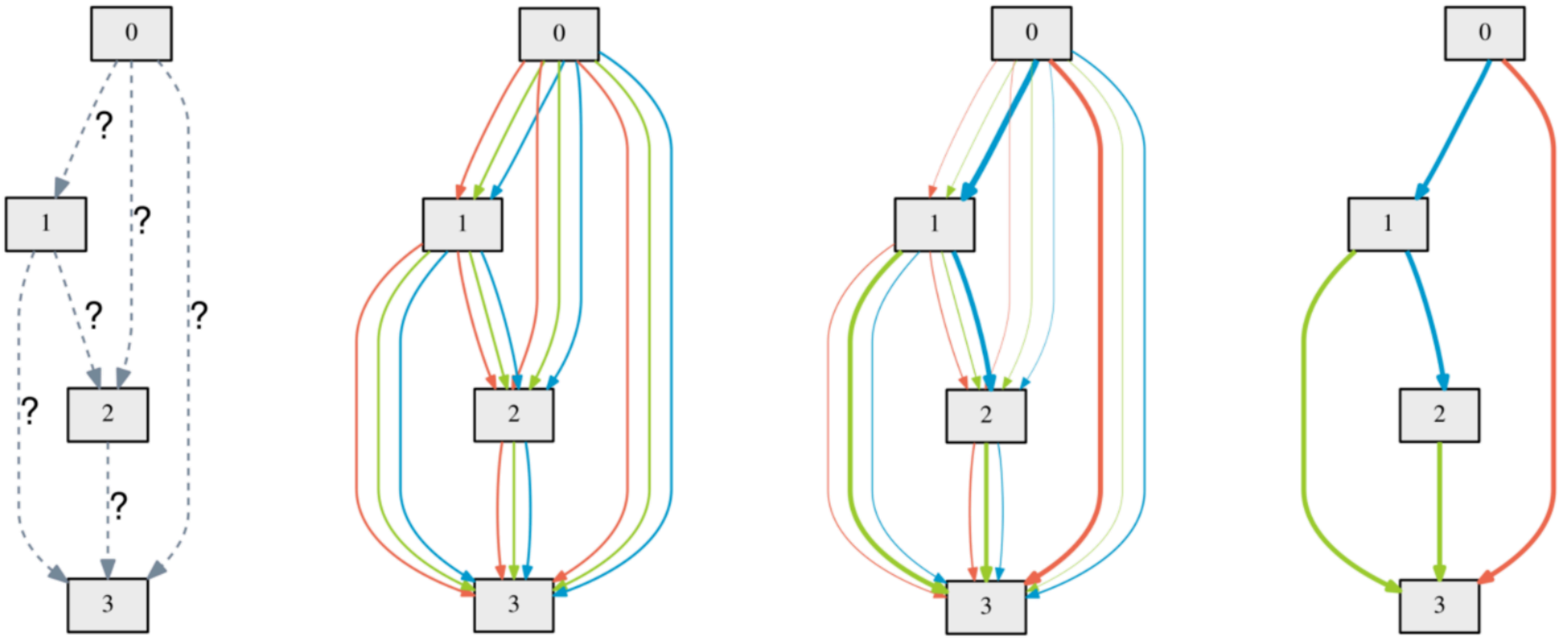
Shared Knowledge

- **Idea.** Transfer knowledge from related training episodes (experience)
- Intuition. Human can learn tasks faster, if already familiar with similar task



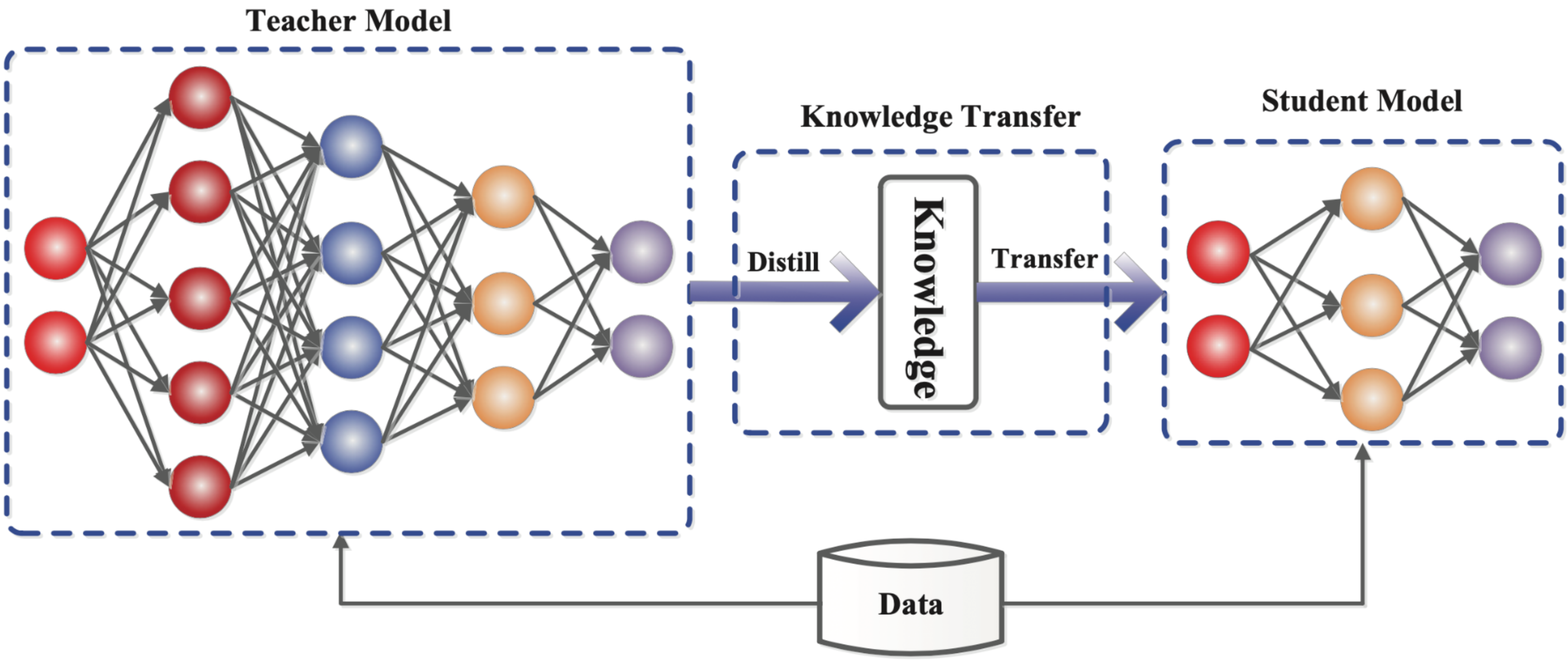
Shared Knowledge

- Note that we have seen this already
- Example. **DARTS** shares weights over architectures to cut training cost



Shared Knowledge

- Example. **Knowledge distillation** transfers teacher knowledge to student



Agenda

- **Today.** Continual Learning

- **Coming next.**
  - Meta-Learning
  - Test-Time Adaptation
  - Parameter-Efficient Fine-Tuning
  - Hyperparameter Transfer
  - Model merging / editing

Basic idea

Motivating example

- **Goal.** Want to build a machine that learns and acts like a human
- **Dumb way.** Run the following algorithm
  - Build a robot with human-like sensory devices
    - Vision / Audio / …
  - Deploy the robot
  - Store all the data it sees
  - Train a model <u>from scratch</u>
    - Frequently, e.g., every hour

Motivating example

- **Problem.** Data explodes!
  - Consider the input data stream:
    - Vision. $\approx$ 10MegaPixels x 24bit RGB x 60Hz $\approx$ 1.8GB/s
    - Audio. $\approx$ 44.1kHz x 16bit scale x 2 channels $\approx$ 0.19 MB/s
    - (...)
  - After a year, we accumulate 56 PB of data (⇔ GPT3 used 570GBs)
  - Out of storage, out of compute!

Motivation

- **Idea.** Learn like human!
  - DON'T: Accumulate all data
    Retrain from scratch every time
  - DO: Keep one model
    Continuously acquire, fine-tune, and transfer new knowledge/skills

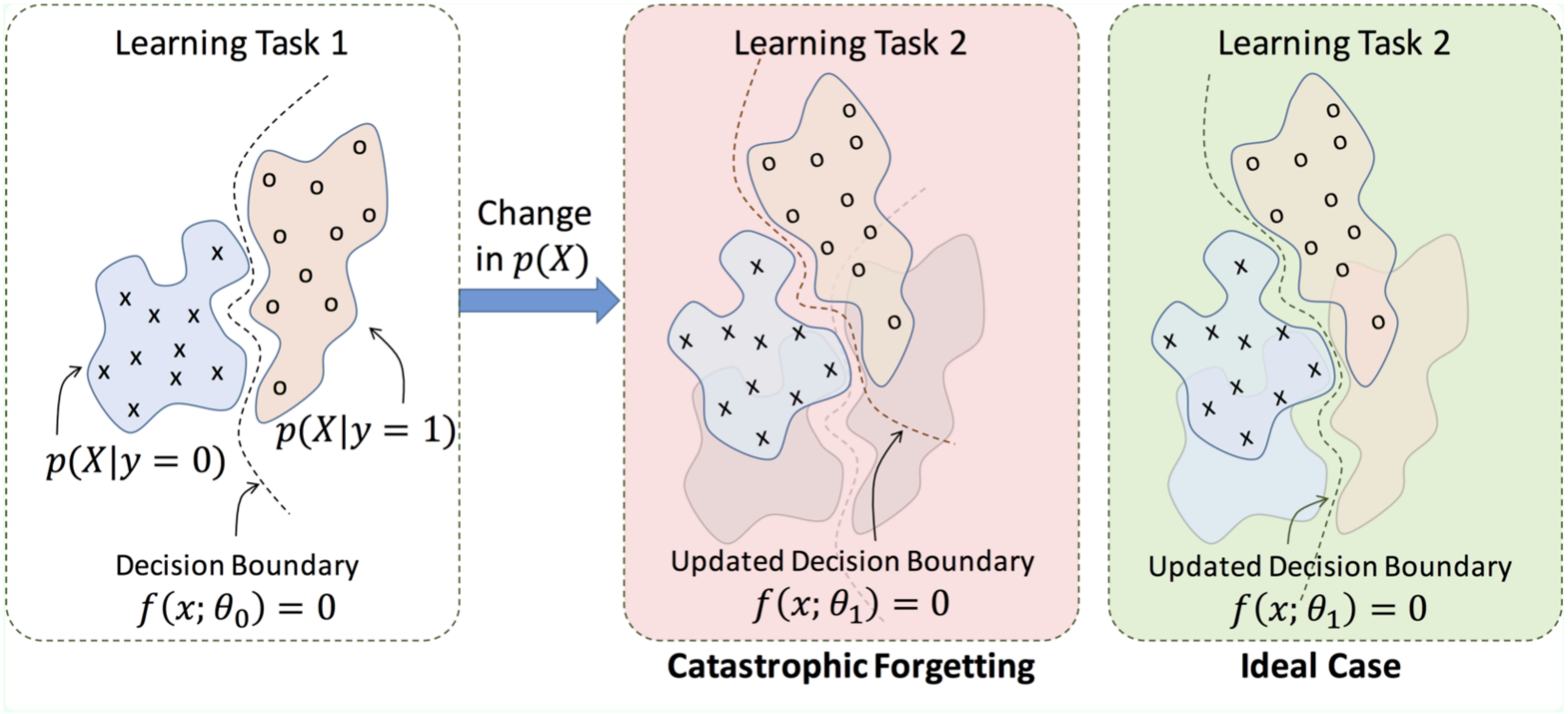
Motivation

- **Naïve.** Run the online learning algorithm
  - Initialize some f_0
  - Repeat: At time t,
    - Collect the data $(\mathbf{x}_t, y_t)$
    - Update the model as $f_{t+1} = f_t - \nabla \ell(f_t(\mathbf{x}_t), y_t)$
    - Discard the data $(\mathbf{x}_t, y_t)$

(recall: classic perceptron algorithm)

Catastrophic forgetting

- **Question.** Are we good?
  - No, if data distribution changes over time!
    - Very degraded performance on previously learned tasks (a.k.a., "catastrophic forgetting")



Catastrophic forgetting

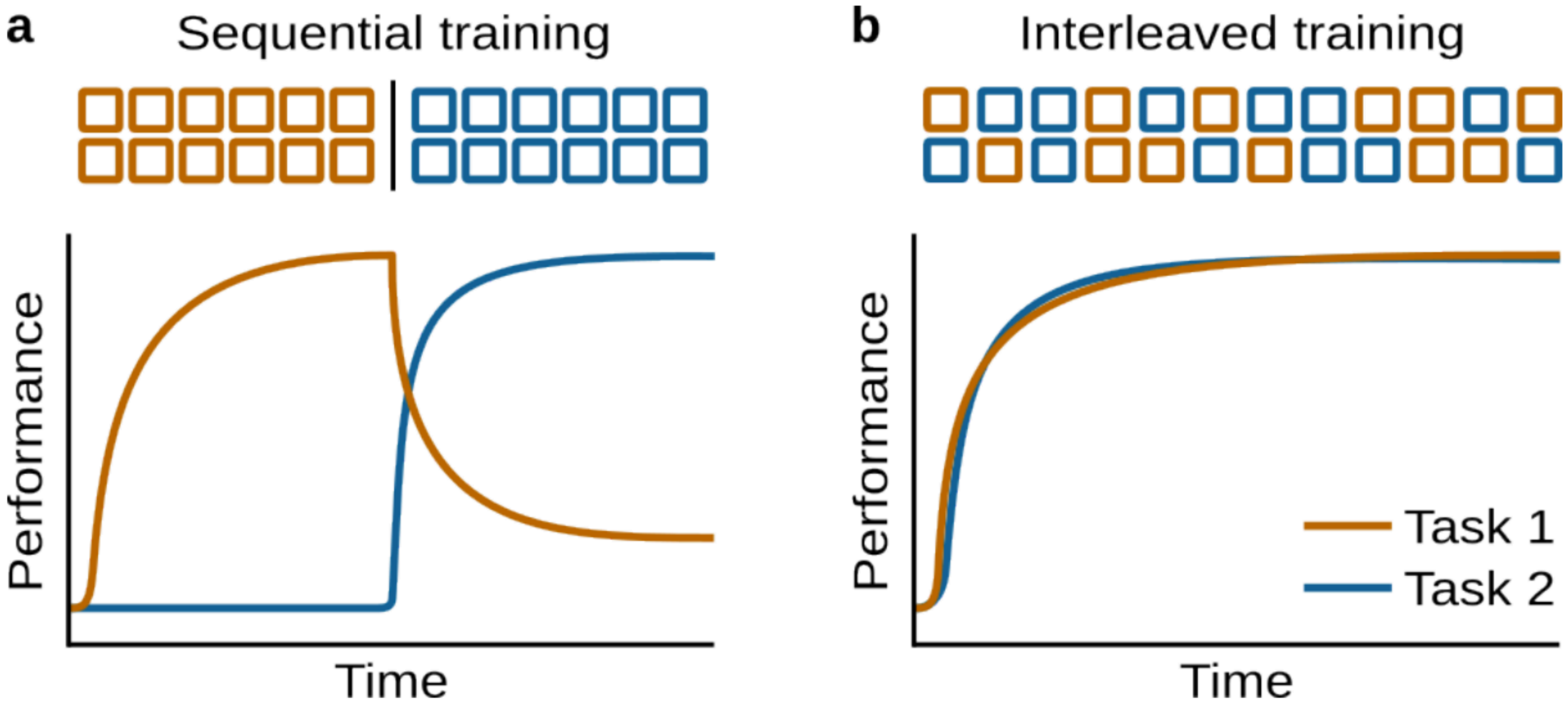
- Example. A clothes recommendation model forgetting everything about prior user preferences from the same season of the previous year

Catastrophic forgetting

- Why does catastrophic forgetting happen?

- Rough Intuition. Training a model with new information interferes with previously learned knowledge

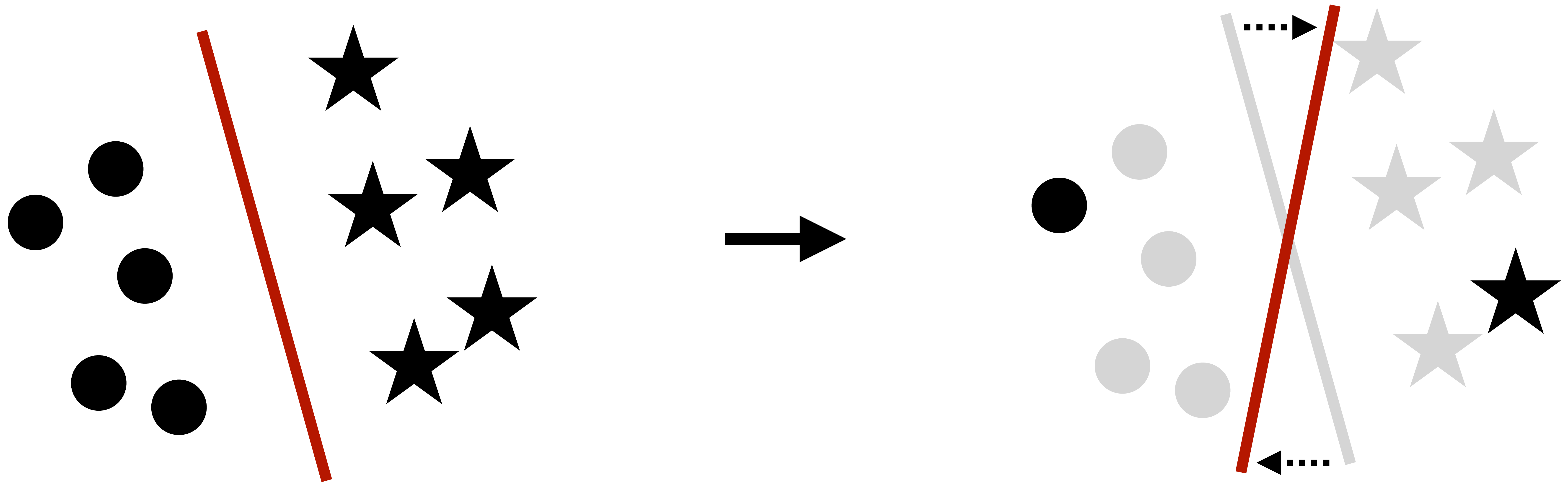
  - Abrupt accuracy drop, or old knowledge completely replaced by new



Van de Ven et al., "Continual learning and catastrophic forgetting," arXiv 2024

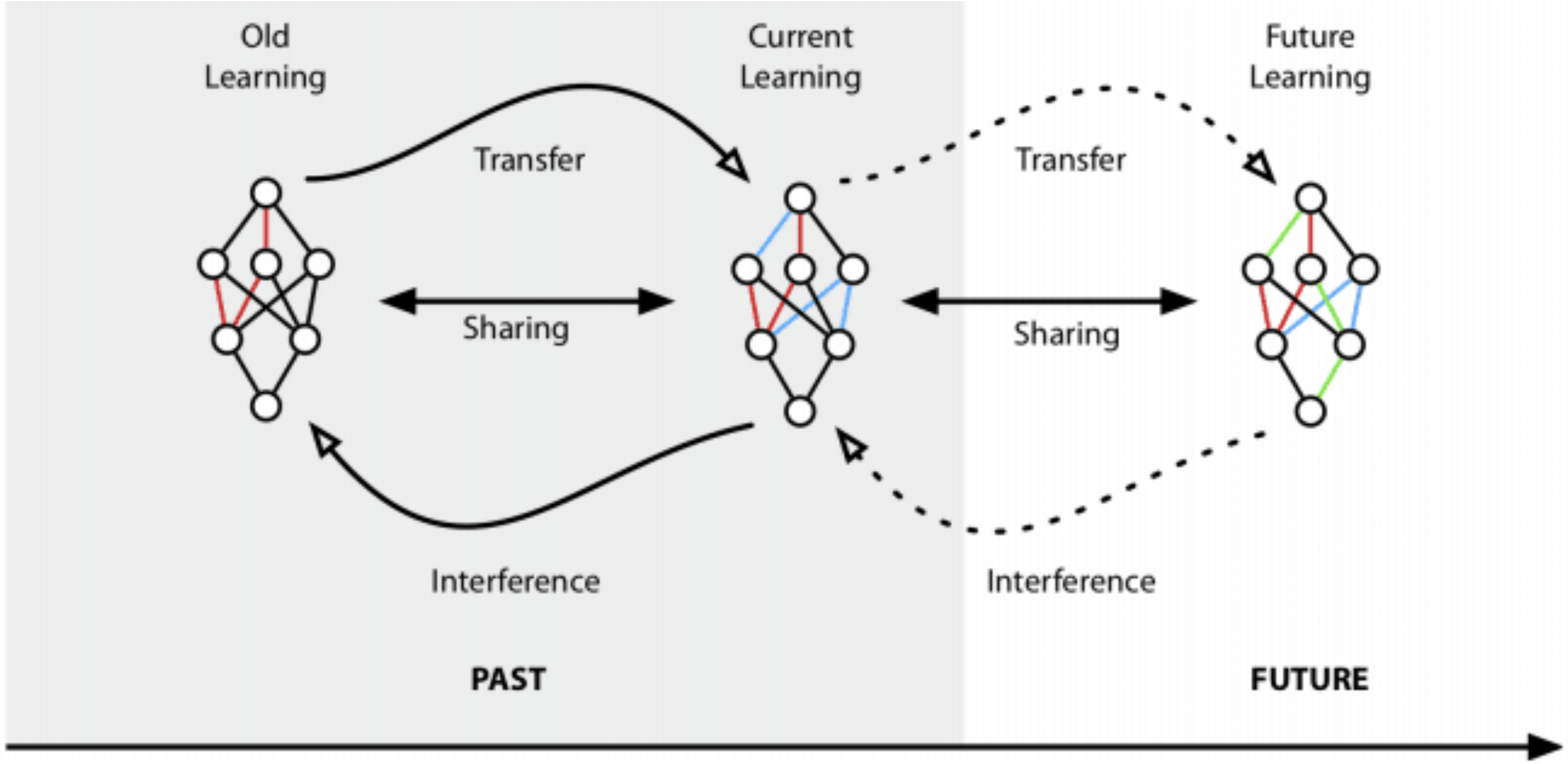
Catastrophic forgetting

- Can be attributed to the nature of SGD
    - Example. Logistic regression
        - Theoretically, always tries to find margin-maximizing solution
          $\Rightarrow$ Departs from previously discovered solution



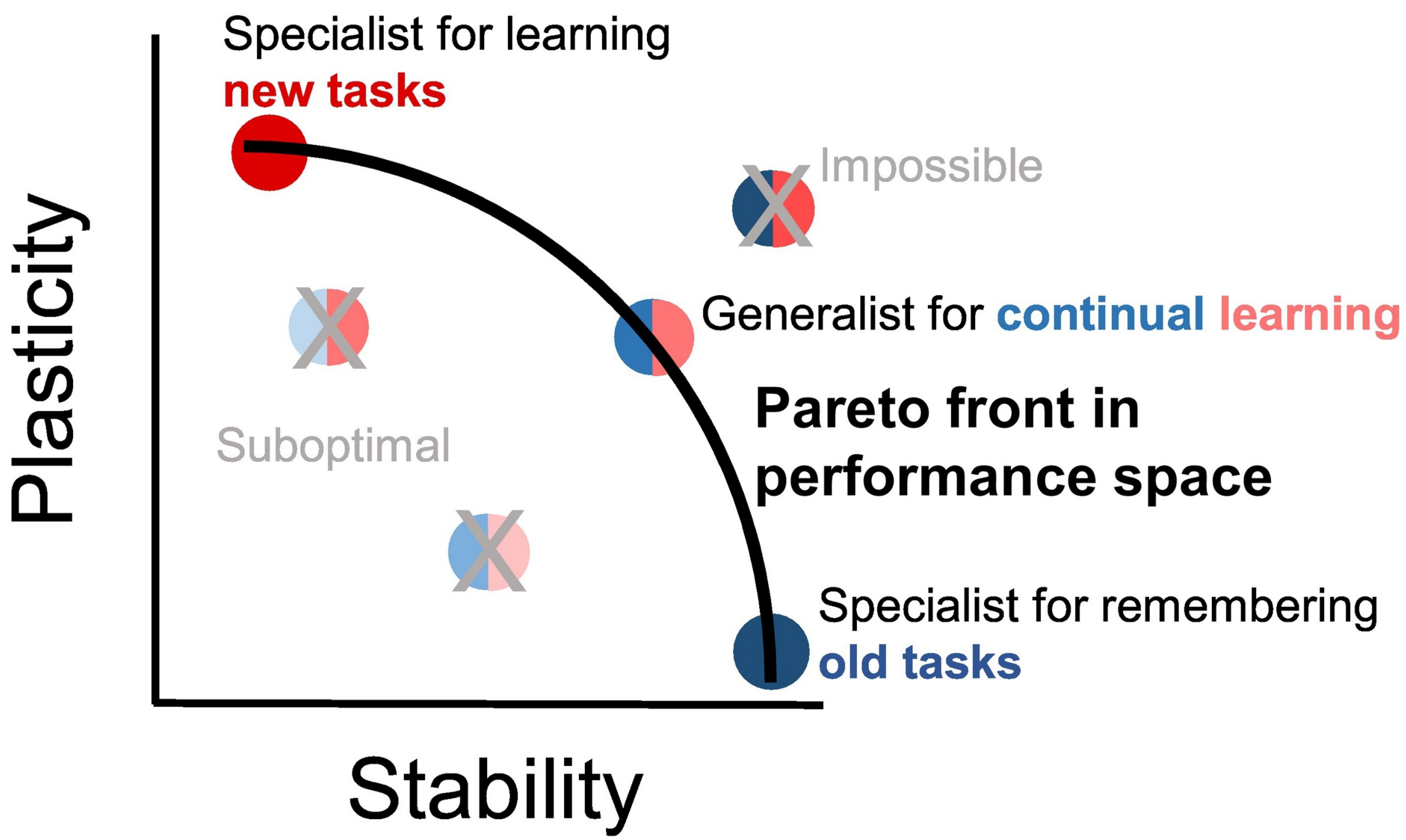
Continual learning

- **Idea.** Share some experience (compressed knowledge) w/ past&future selves
    - No need to store all prior data, or train from scratch
    - Ideally, Want to improve performance on past task, using current data



Continual learning

- If we focus too much on remembering the past self
  - Performs poorly on current task
- If we completely forget
  - Performs poorly on past task
- Called "Stability-Plasticity Dilemma"



Formalizations

Categories

- For classification tasks, there are three popular categories:
    - Task-incremental
    - Domain-incremental
    - Class-incremental

Formalization

- Consider a supervised learning setup, with non-stationary data stream:
  - Each datum consists of a triplet

$$(\mathbf{x}_i, y_i, t_i)$$

    - Feature $\mathbf{x} \in \mathscr{X}$
    - Label $y \in \mathscr{Y}$
    - Task $t \in \mathscr{T}$
      - denotes which distribution the datum belongs

De Lange et al., "A continual learning survey: …," TPAMI 2021

Formalization

- **Task-incremental learning**
  - For training data, we know $(\mathbf{x}_i, y_i, t_i)$
  - For test data, we know $(\mathbf{x}, t)$
  - Example. Clothes recommender, with seasonal info
    - $\mathbf{x}$: Clothes info
    - y: Preference
    - t: Current season
- **Focus.** Achieve positive transfer between tasks

Formalization

- **Domain-incremental learning**
  - For training data, we know $(\mathbf{x}_i, y_i, t_i)$
  - For test data, we know $(\mathbf{x})$
  - Example. Self-driving car w/o weather info
    - $\mathbf{x}$: Visual input from camera
    - y: Driving actions
    - t: Weather information

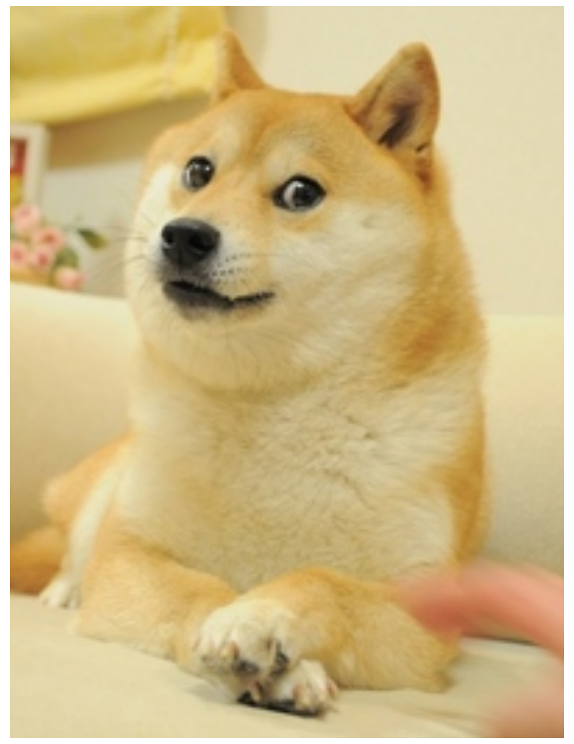
- **Focus.** Alleviate catastrophic forgetting

Formalization

- **Class-incremental learning**
  - DIL, but we need to infer task information as well
    (e.g., $\mathscr{Y}$ differs from task to task)
  - Example. Learning to classify animals, with streaming classes
  - **Focus.** Classifying classes not observed together
    - Can we use “folded ears” as a decisive feature?

Task 1

Task 2

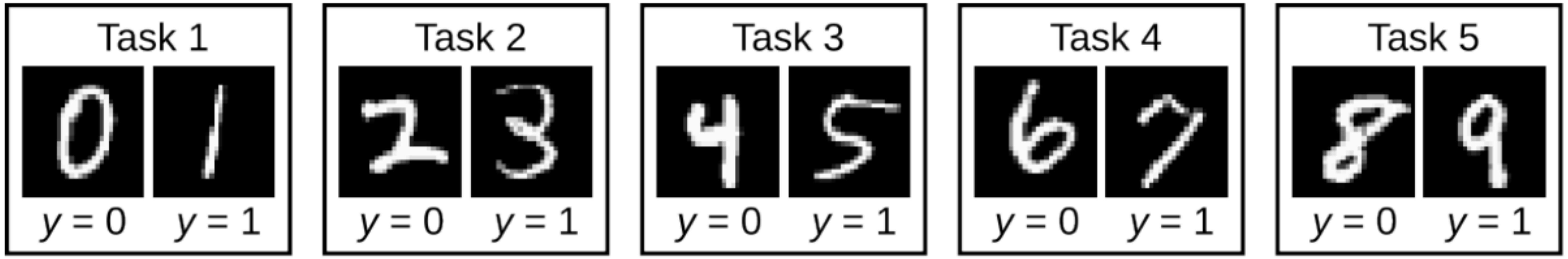


Test

Formalization

- Typically the difficulty goes as:

$$\text{TIL} < \text{DIL} < \text{CIL}$$



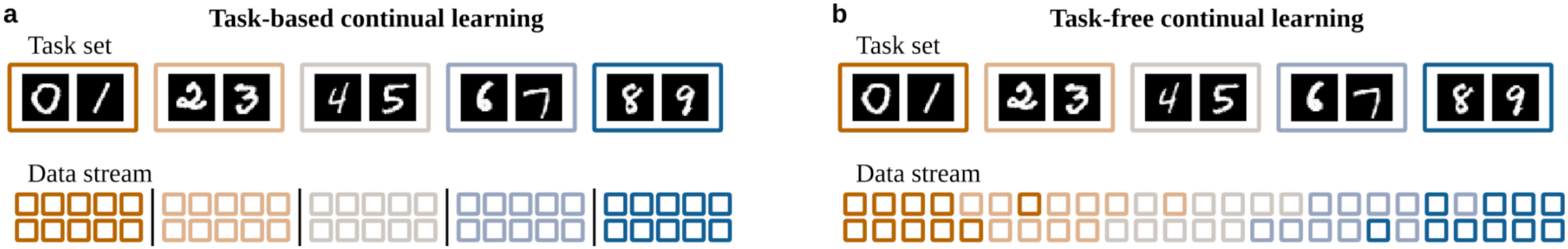
$\mathcal{X}$ = image pixel space
$\mathcal{T}$ = task set = {1,2,3,4,5}
$\mathcal{Y}$ = within-task label space = {0,1}

Scenario	*Type of choice*	*Mapping to learn*
Task-incremental learning	Choice between two digits of same task (e.g., 0 or 1?)	$f: \mathcal{X} \times \mathcal{T} \rightarrow \mathcal{Y}$
Domain-incremental learning	Is the digit odd or even?	$f: \mathcal{X} \rightarrow \mathcal{Y}$
Class-incremental learning	Choice between all ten digits	$f: \mathcal{X} \rightarrow \mathcal{T} \times \mathcal{Y}$

Van de Ven et al., "Continual learning and catastrophic forgetting," arXiv 2024

Formalization

- Even further, recent works consider task-free cases
  - No task label available, and unclear boundaries
  - Practical, but extremely challenging



Van de Ven et al., "Continual learning and catastrophic forgetting," arXiv 2024

Strategies

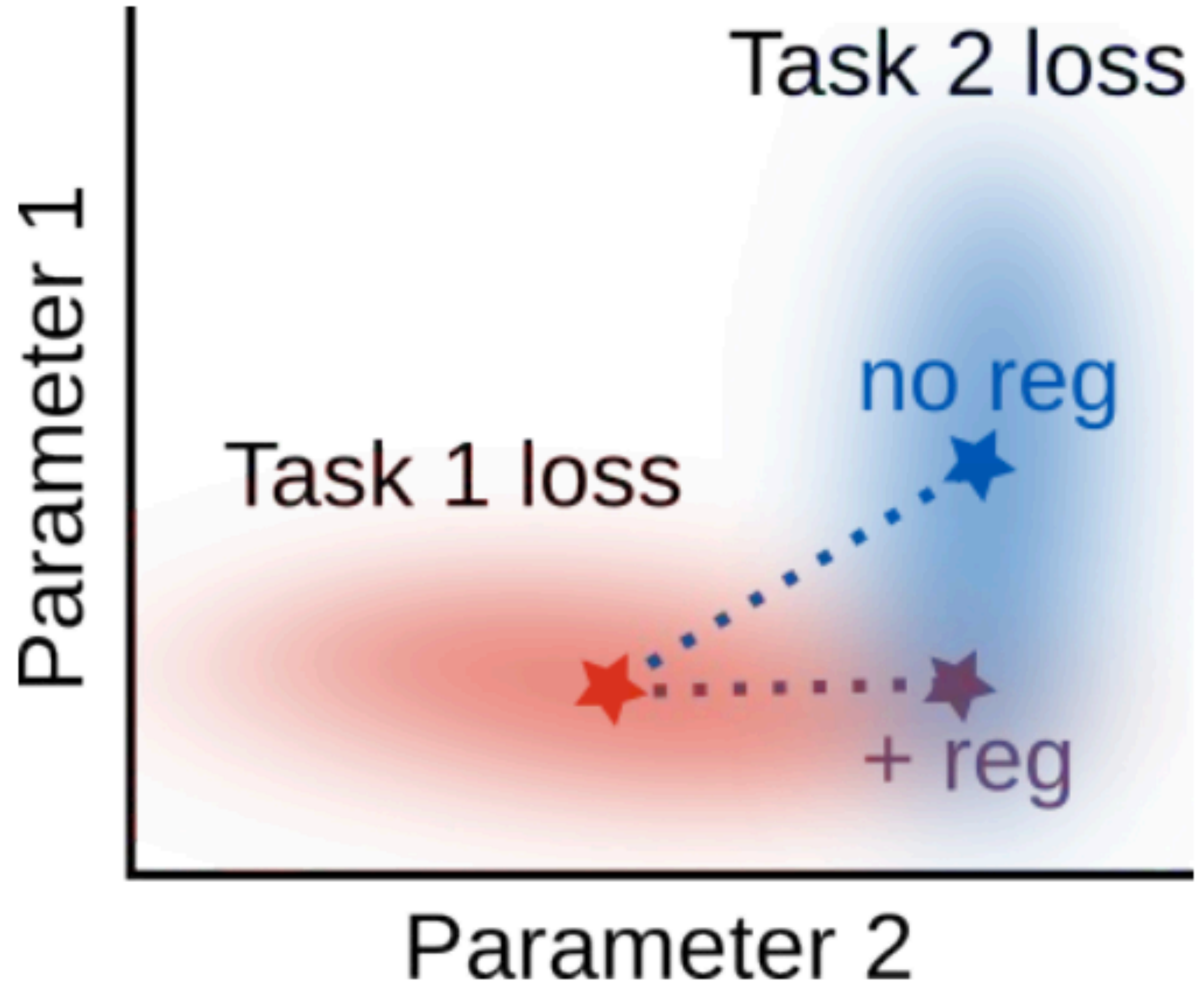
Categories

- Here are some popular options:
  - Regularization-based
  - Replay-based
  - Template-based
  - Context-dependent Processing

Regularization

- **Idea.** Regularize the parameters to prevent shifting much from the past self
    - That is, minimize the loss function

$$L(\theta) = L_{\text{new}}(\theta) + \text{dist}(\theta, \theta_{\text{past}})$$



Regularization

- Example. Elastic Weight Consolidation
    - Suppose that we want to minimize:

$$L_{\text{new}}(\theta) + L_{\text{past}}(\theta)$$

    - Apply the Taylor approximation:

$$L_{\text{new}}(\theta) + L_{\text{past}}(\theta_{\text{past}}) + G_{\text{past}}^\top(\theta - \theta_{\text{past}}) + (\theta - \theta_{\text{past}})^\top H_{\text{past}}(\theta - \theta_{\text{past}})$$

        - Remove unnecessary terms to get:

$$L_{\text{new}}(\theta) + \boxed{(\theta - \theta_{\text{past}})^\top H_{\text{past}}(\theta - \theta_{\text{past}})}$$

(actual version uses Fisher information matrix, which is easier to compute)

Kirkpatrick et al., "Overcoming catastrophic forgetting in neural networks," PNAS 2017

Regularization

- Synaptic intelligence (SI) measures a similar metric
  - Difference: Measured over the entire learning trajectory
  - https://arxiv.org/abs/1703.04200

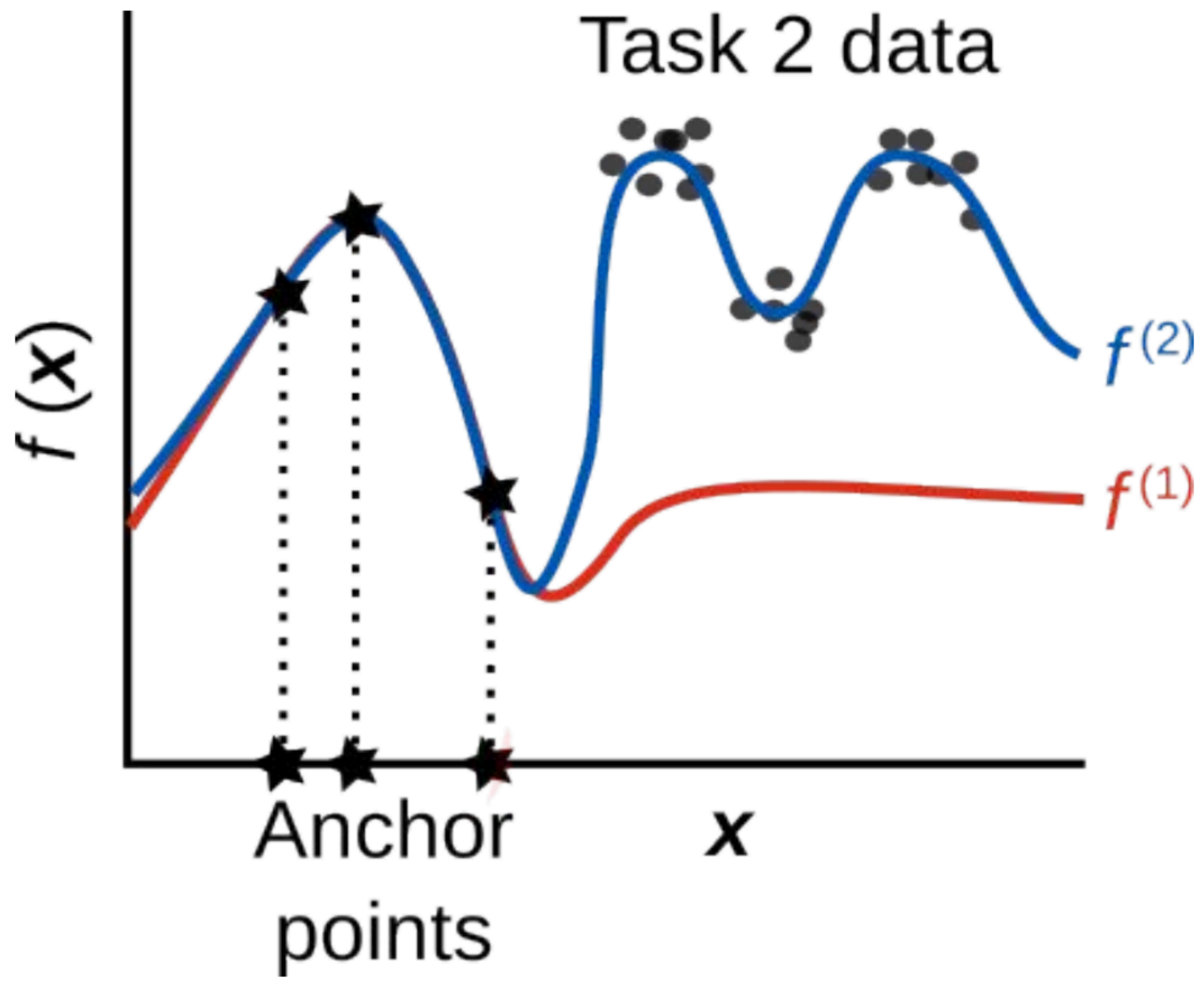
Regularization

- Some works conduct functional regularization, instead of parameter reg.

- **Idea.** Preserve the predictions of past model on anchor points

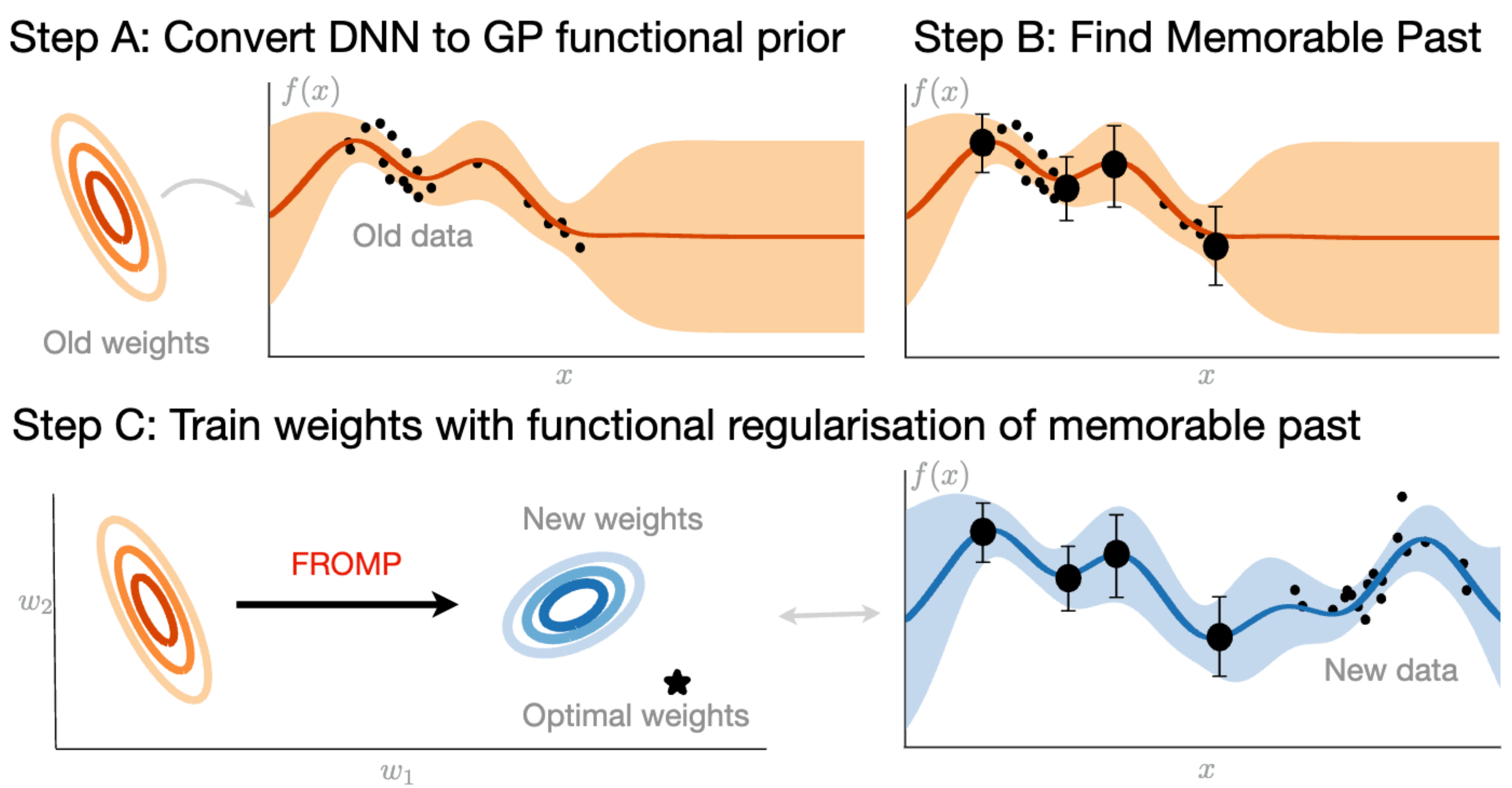
$$L(\theta) = L_{\text{new}}(\theta) + \mathbb{E}[\text{dist}(f_\theta(\tilde{x}), f_{\theta_{\text{past}}}(\tilde{x}))]$$

  - Ideally, anchors should be samples from the past task



Regularization

- Example. Learning without forgetting (LwF)
  - Use the samples from the new task as the anchor points
- Example. FROMP
  - Recover "memorable past" from the past model

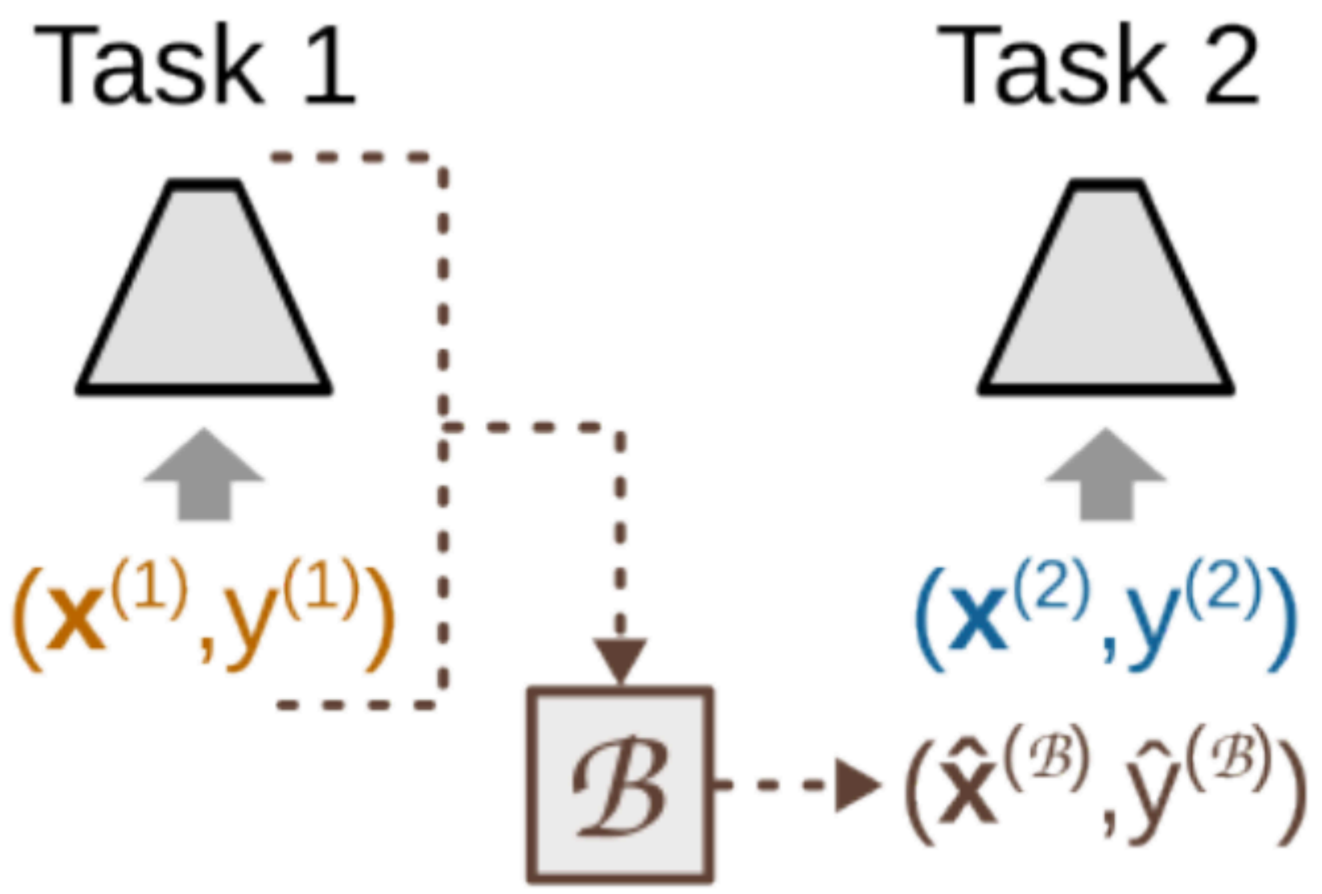


Replay

- **Idea.** Train the model w/ current task samples + representative past samples

$$L(\theta) = L_{\text{new}}(\theta) + L_{\text{past,rep}}(\theta)$$

  - Example. Experience replay (ER) stores random data in buffer
  - Example. Deep Generative Replay (DGR) trains GAN to generate samples

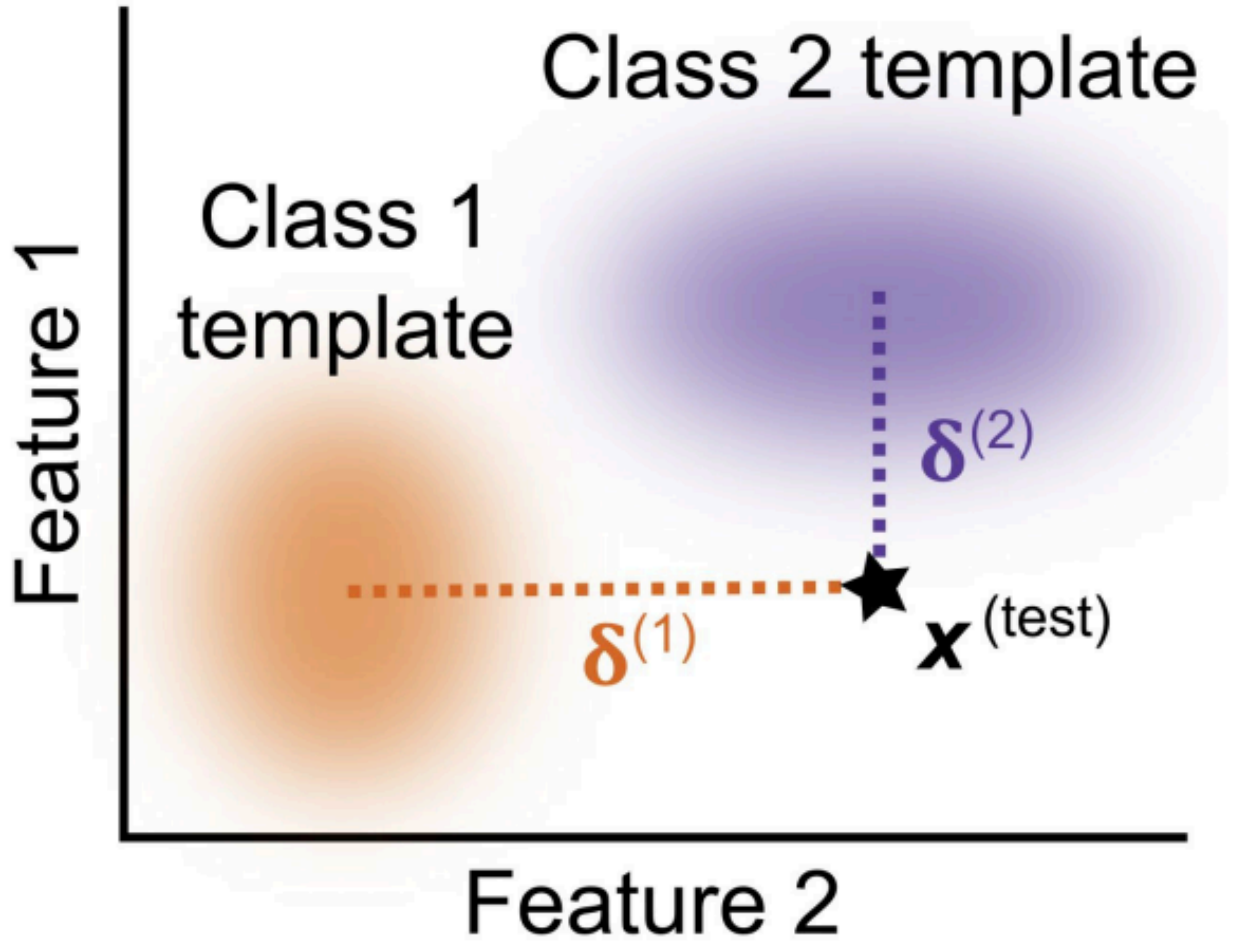


Template-based

- **Idea.** Classify based on "templates" that are kept for each class
  - Example. iCaRL stores some exemplar samples from each class:
    - Classification is done by nearest-mean-of-exemplars
    - Feature extractor can be trained, e.g., via self-supervised learning

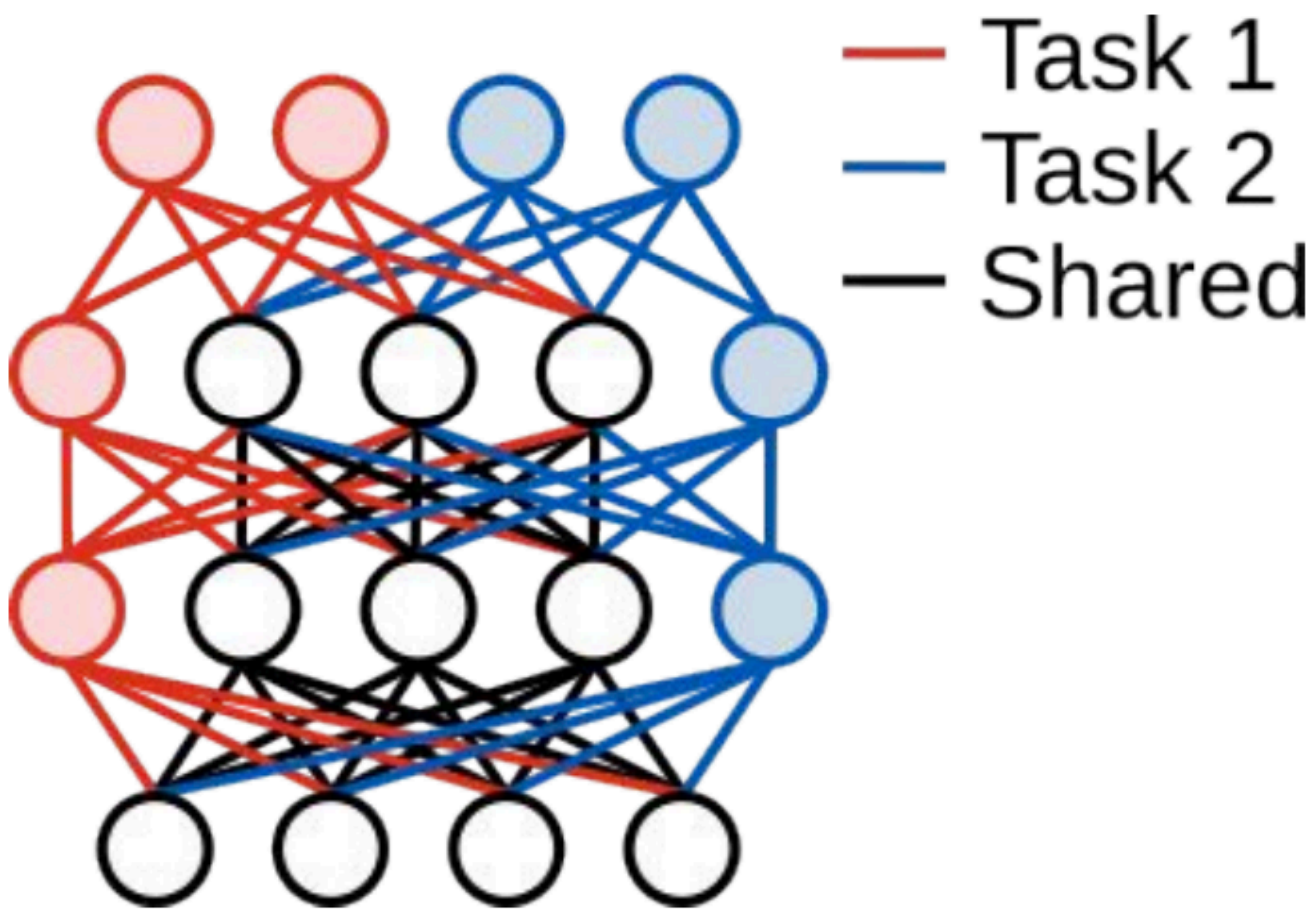
Algorithm 1 iCaRL CLASSIFY

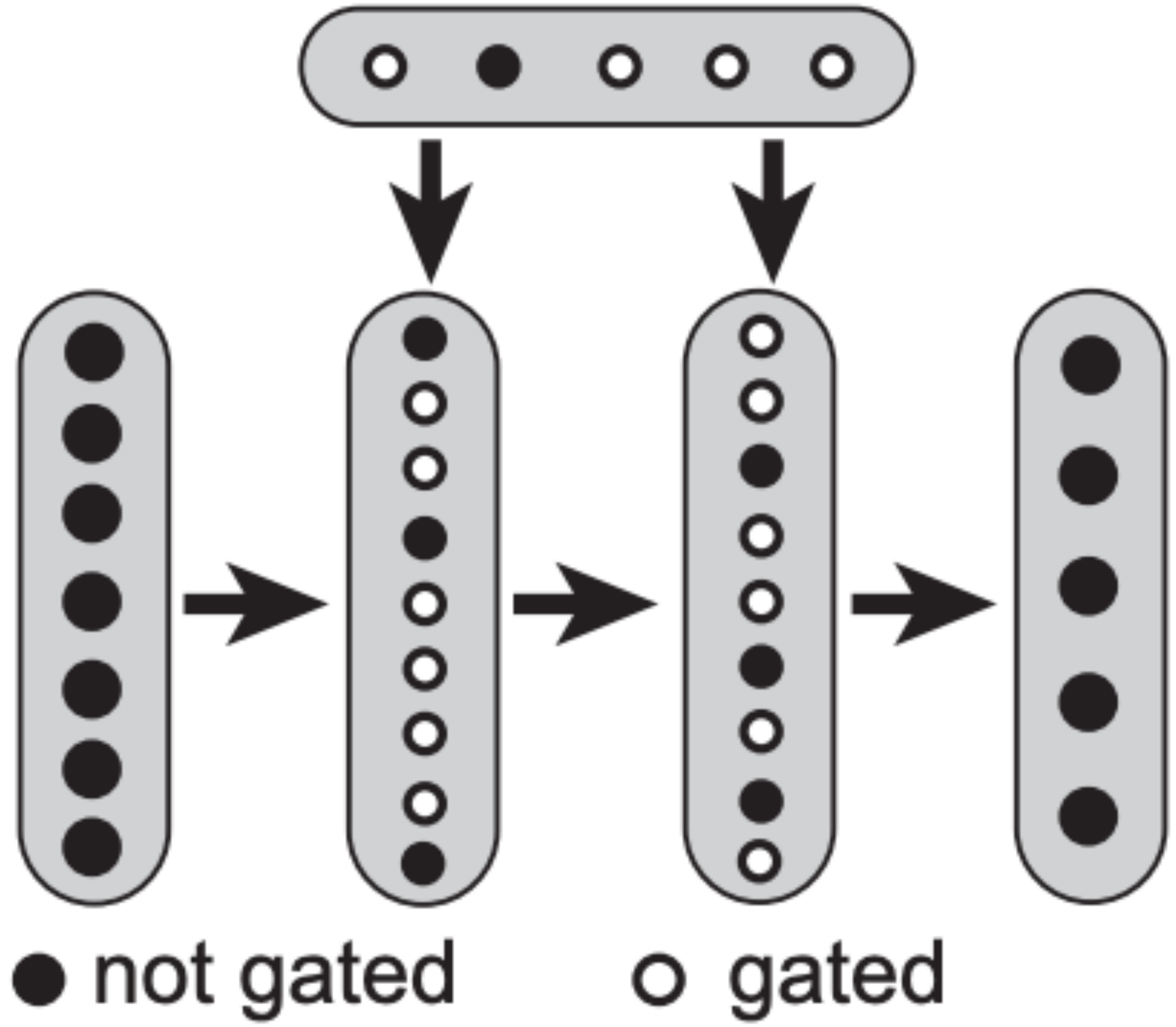
input x // image to be classified
require $\mathcal{P} = (P_1, \dots, P_t)$ // class exemplar sets
require $\varphi : \mathcal{X} \to \mathbb{R}^d$ // feature map
  for $y = 1, \dots, t$ **do**
    $\mu_y \leftarrow \frac{1}{|P_y|} \sum_{p \in P_y} \varphi(p)$ // mean-of-exemplars
  end for
  $y^* \leftarrow \underset{y=1,\dots,t}{\operatorname{argmin}} \|\varphi(x) - \mu_y\|$ // nearest prototype
output class label y^*



Context-dependent processing

- **Idea.** Assign some parameters exclusively for a specific task
- <u>Example.</u> Context-dependent gating jointly trains a gating function with the model parameters
  - Used together with EWC





Further readings

- In the context of LLMs, continual learning is done in various stages
  - Pre-training
    - https://arxiv.org/abs/2403.08763
  - Instruction tuning
    - https://arxiv.org/pdf/2205.12393
  - Alignment
    - https://arxiv.org/abs/2407.05342

That’s it for today 🙌