Continual Prototype Evolution: Learning Online from Non-Stationary Data Streams

Matthias De Lange
KU Leuven
matthias.delange@kuleuven.be

Tinne Tuytelaars
KU Leuven
tinne.tuytelaars@kuleuven.be

https://arxiv.org/pdf/2009.00919.pdf

Roadmap

- Learner-Evaluator framework
- Data incremental learning
- Prior Work
- CoPE
  - Evolving prototypes
  - Balanced replay
  - PPP-loss
- Future work

Roadmap

- Learner-Evaluator framework
- Data incremental learning
- Prior Work
- CoPE
  - Evolving prototypes
  - Balanced replay
  - PPP-loss
- Future work

Follow along:

https://arxiv.org/pdf/2009.00919.pdf

Slides in Slack!

Learner-evaluator framework

- Current paradigms defined in terms of ‘task’ information

Scenario	*Required at test time*
Task-IL	Solve tasks so far, task-ID provided
Domain-IL	Solve tasks so far, task-ID not provided
Class-IL	Solve tasks so far *and* infer task-ID

[1]

[1] van de Ven, Gido M., and Andreas S. Tolias. "Three scenarios for continual learning." *arXiv preprint 1904.07734* (2019).

Questions? matthias.delange@kuleuven.be

Learner-evaluator framework

- Current paradigms defined in terms of ‘task’ information

Scenario	*Required at test time*
Task-IL	Solve tasks so far, task-ID provided
Domain-IL	Solve tasks so far, task-ID not provided
Class-IL	Solve tasks so far *and* infer task-ID

[1]

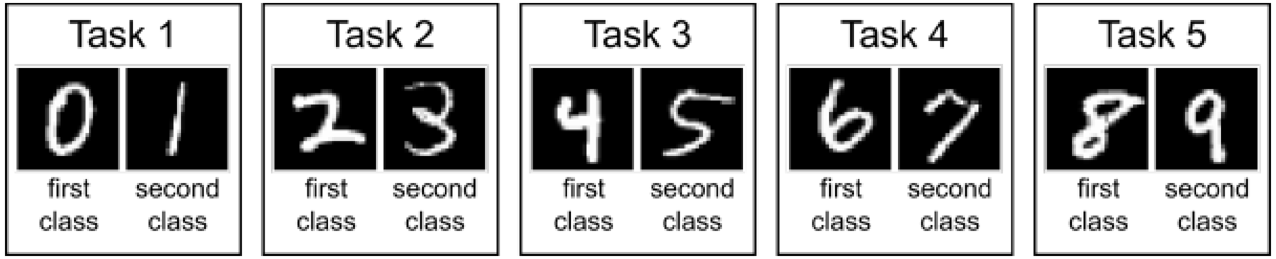
What exactly is a task?

[1] van de Ven, Gido M., and Andreas S. Tolias. "Three scenarios for continual learning." *arXiv preprint 1904.07734* (2019).

Questions? matthias.delange@kuleuven.be

What exactly is a task?

- Grouped data subset by designer → Explicit bias by design



[1]

- Algorithmically, e.g. every K new classes → Implicit bias by design

How to define task-free setups?

Learner-evaluator framework

- Current paradigms defined in terms of ‘task’ information

Scenario	*Required at test time*
Task-IL	Solve tasks so far, task-ID provided
Domain-IL	Solve tasks so far, task-ID not provided
Class-IL	Solve tasks so far *and* infer task-ID

[1]

What exactly is a task?

How to define task-free settings?

[1] van de Ven, Gido M., and Andreas S. Tolias. "Three scenarios for continual learning." *arXiv preprint 1904.07734* (2019).
Questions? matthias.delange@kuleuven.be

Learner-evaluator framework

- Current paradigms defined in terms of ‘task’ information

Scenario	*Required at test time*
Task-IL	Solve tasks so far, task-ID provided
Domain-IL	Solve tasks so far, task-ID not provided
Class-IL	Solve tasks so far *and* infer task-ID

[1]

What exactly is a task?

What resources are available?

How to define task-free settings?

[1] van de Ven, Gido M., and Andreas S. Tolias. "Three scenarios for continual learning." *arXiv preprint 1904.07734* (2019).

Questions? matthias.delange@kuleuven.be

Learner-evaluator framework

- Current paradigms defined in terms of 'task' information

Scenario	*Required at test time*
Task-IL	Solve tasks so far, task-ID provided
Domain-IL	Solve tasks so far, task-ID not provided
Class-IL	Solve tasks so far *and* infer task-ID

[1]

What exactly is a task?

What resources are available?

How to define task-free settings?

What information is available when?

[1] van de Ven, Gido M., and Andreas S. Tolias. "Three scenarios for continual learning." *arXiv preprint 1904.07734* (2019).

Questions? matthias.delange@kuleuven.be

PSI

KU LEUVEN

Learner-evaluator framework

- Current paradigms defined in terms of ‘task’ information

Scenario	*Required at test time*
Task-IL	Solve tasks so far, task-ID provided
Domain-IL	Solve tasks so far, task-ID not provided
Class-IL	Solve tasks so far *and* infer task-ID

[1]

What exactly is a task?

What resources are available?

How to define task-free settings?

How are training & testing interacting?

What information is available when?

[1] van de Ven, Gido M., and Andreas S. Tolias. "Three scenarios for continual learning." *arXiv preprint 1904.07734* (2019).

Questions? matthias.delange@kuleuven.be

How are training and testing interacting?

- Standard ML
  - Static training (iid)
  - Static testing (iid)

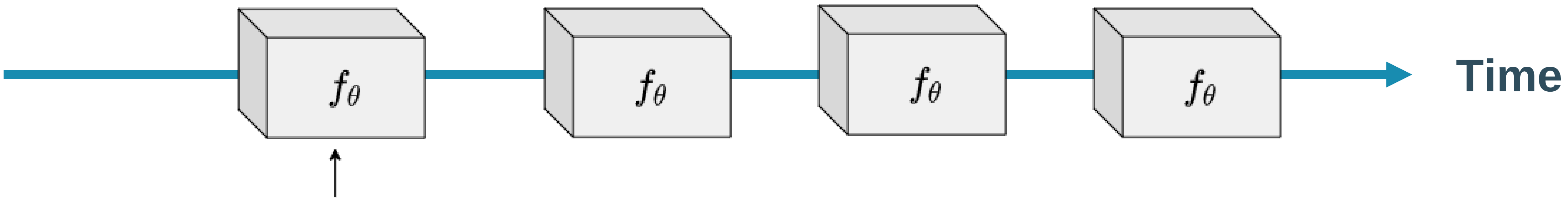
- Continual learning
  - **Continual training** from non-stationary data (non-iid)
  - **Static evaluation** (iid)
    → Is evaluation sequential? = Undefined
    → "Stop training for evaluation?" = Undefined
    → Wait? How is it 'continual' training then? = Undefined?

    → What about drifting concepts? (continual evaluation)

Questions? matthias.delange@kuleuven.be

Concept drift 1-on-1

Testing phase:



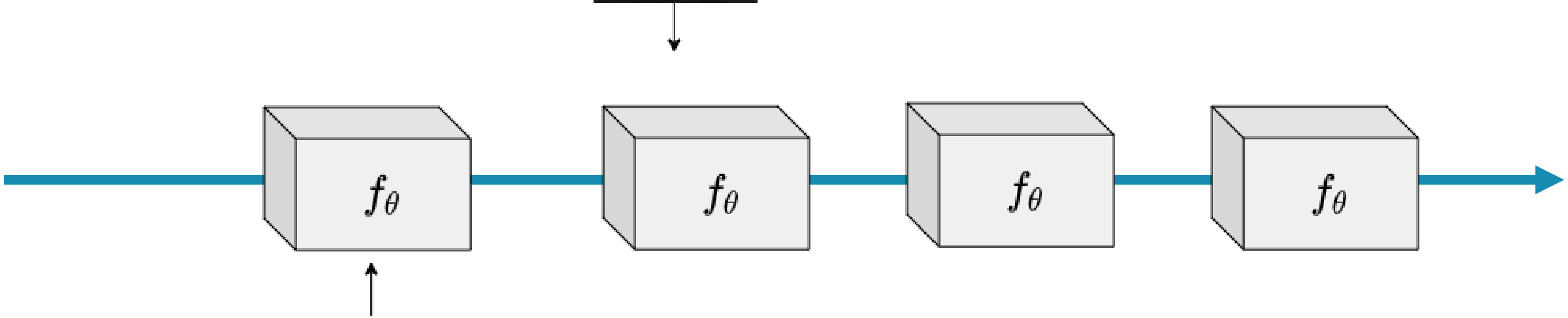
Training phase:

Concept:
Felix

Questions? matthias.delange@kuleuven.be

“You still know my pal Felix?”

Testing phase:



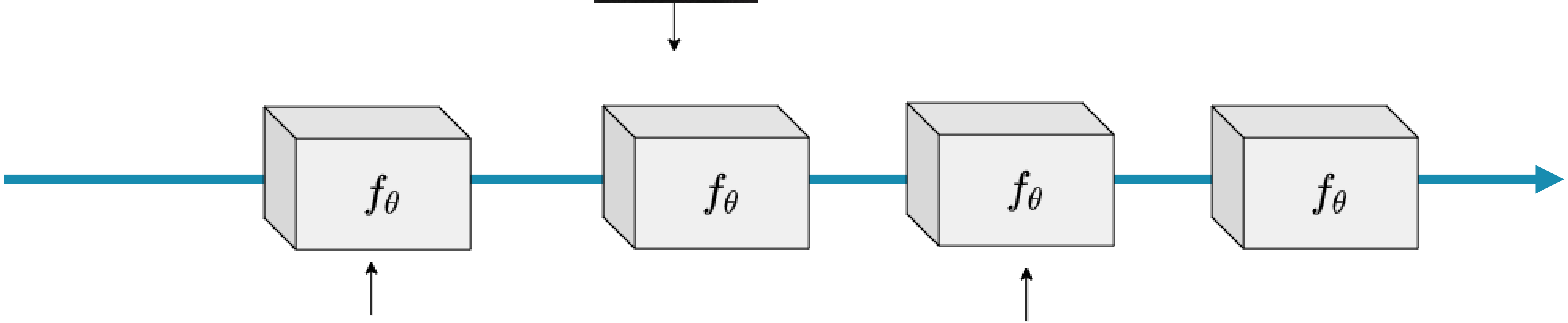
Time

Training phase:

Concept:
Felix

"You still know
my pal Felix?"

Testing phase:



Time

Training phase:

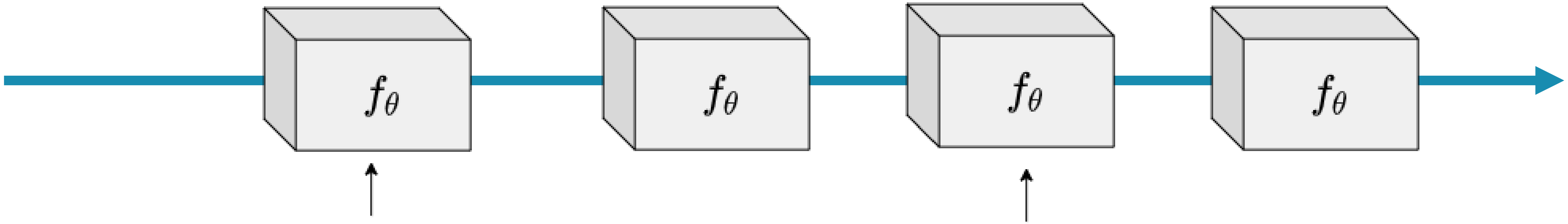
Concept:
Felix

Learn other cats

Testing phase:

“You still know my pal Felix?”

“What about now?”



Time

Training phase:

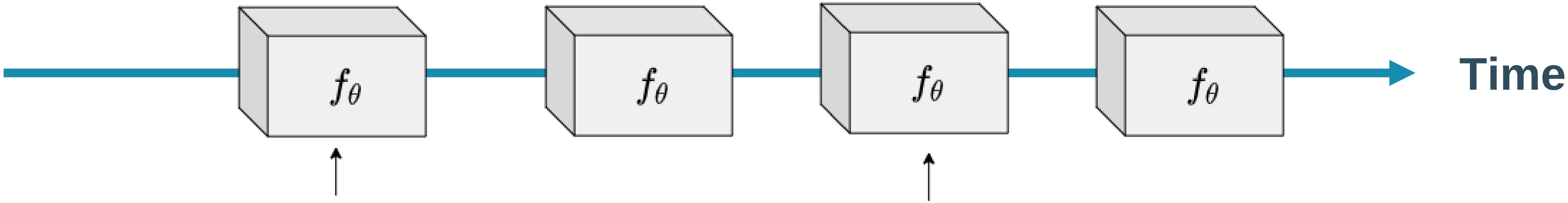
Concept: Felix

Learn other cats

“You still know my pal Felix?”

“What about now?”

Testing phase:



Training phase:

Concept:
Felix

Learn other cats

Learner-evaluator framework

- Current paradigms defined in terms of 'task' information

Scenario	*Required at test time*
Task-IL	Solve tasks so far, task-ID provided
Domain-IL	Solve tasks so far, task-ID not provided
Class-IL	Solve tasks so far *and* infer task-ID

[1]

What exactly is a task?

What resources are available?

How to define task-free settings?

How are training & testing interacting?

What information is available when?

What about drifting concepts?

[1] van de Ven, Gido M., and Andreas S. Tolias. "Three scenarios for continual learning." *arXiv preprint 1904.07734* (2019).

Questions? matthias.delange@kuleuven.be

Answers

- Two-agent framework: Learner & Evaluator

  - Operate independently
  - Generalizable to any data stream
    - No notion of task required
  - Generalizable to any evaluation
    - Concept drift ❤️ Continual Learning
  - Horizon $\mathcal{D}_t$
    Operational memory $\mathcal{M}$

How are training & testing interacting?

What exactly is a task?
How to define task-free settings?

What about drifting concepts?

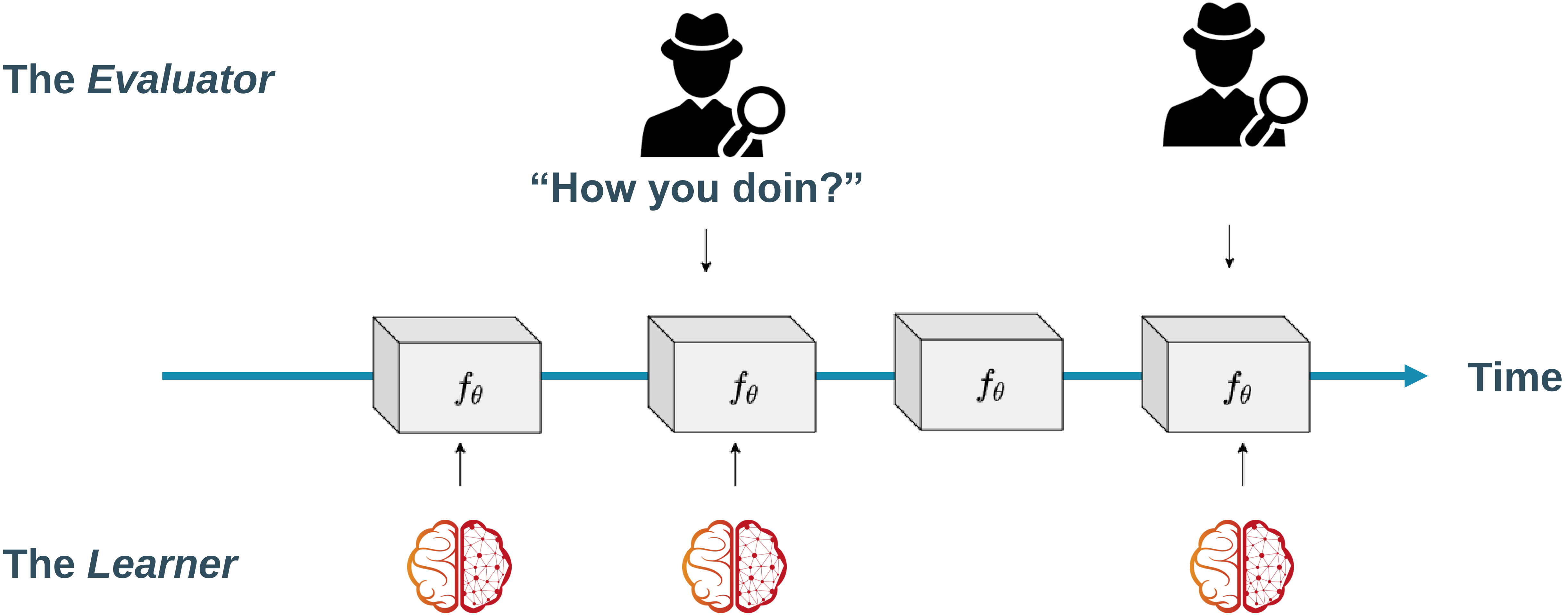
What information is available when?
What resources are available?

Learner-evaluator framework

The *Learner*

The *Evaluator*

Questions? matthias.delange@kuleuven.be



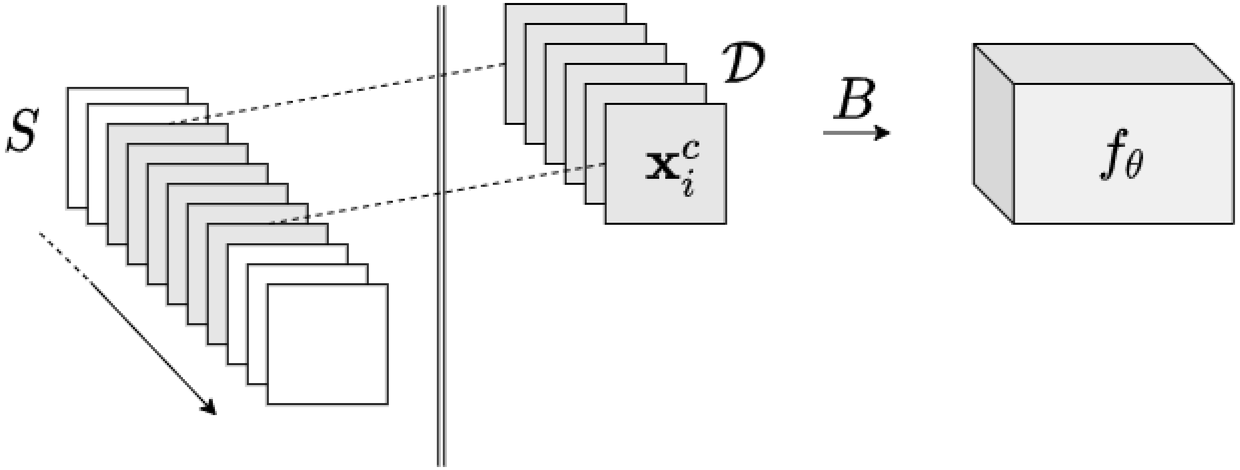
The *Evaluator*
"How you doin?"
f_θ
f_θ
f_θ
f_θ
Time
The *Learner*

Learner-evaluator framework

The *Learner*

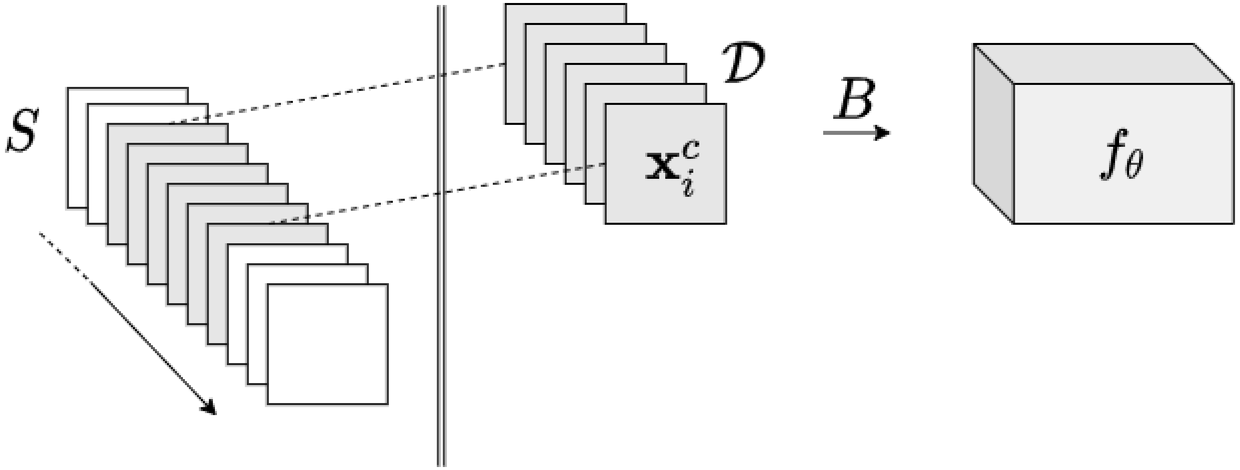
The *Evaluator*

The *Learner*



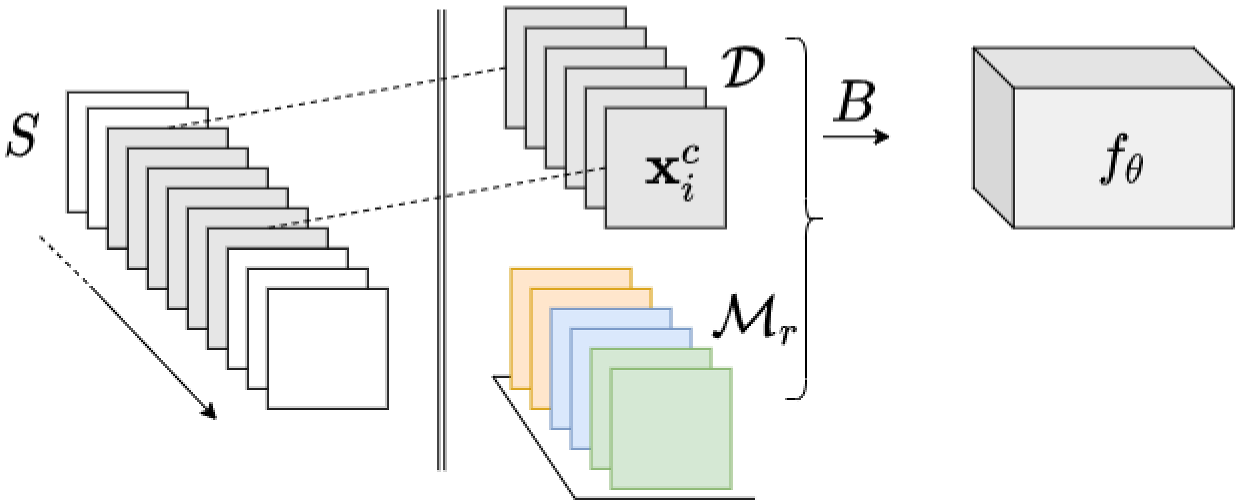
- Observes non-stationary data stream *S* with data samples $(\mathbf{x}_i, \mathbf{y}_i)$
- Horizon $\mathcal{D}$: The observable subset of *S*
  - Simultaneously available to the learner
- Processing batch *B*: Small-scale sampling for stochasticity/multiple updates

The *Learner*



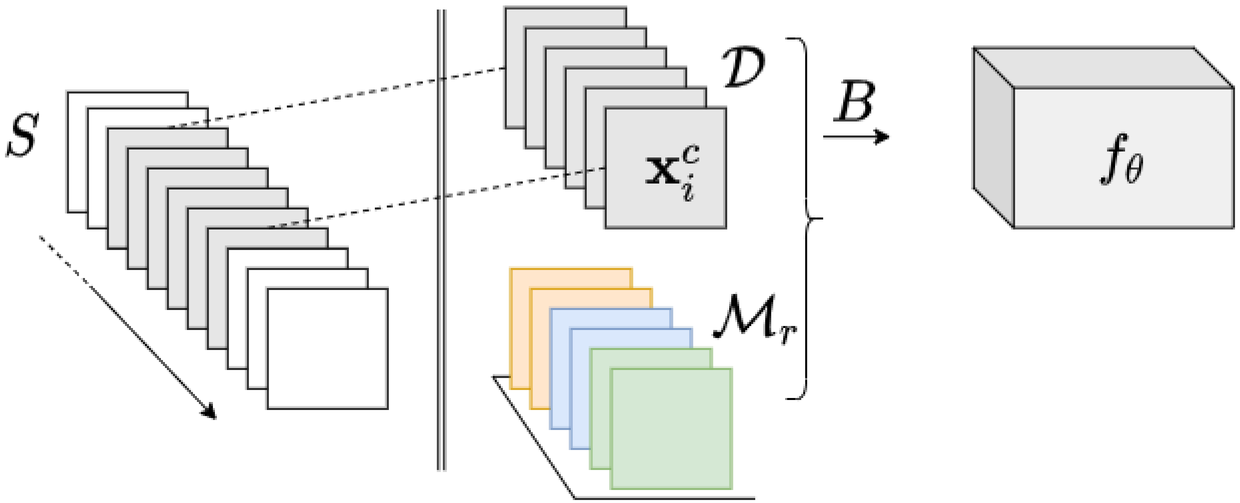
- Application/setup determine horizon $\mathcal{D}$
  - Offline (standard iid ML) → $\mathcal{D} = S$
  - Online (non-iid) → $\mathcal{D} = B$

The *Learner*



- Horizon $\mathcal{D}$ → Memory for observable data of *S*
- Operational Memory $\mathcal{M}$ → Memory for operation of CL algorithm
  - Memory after processing the data
  - E.g. episodic memory

The *Learner*



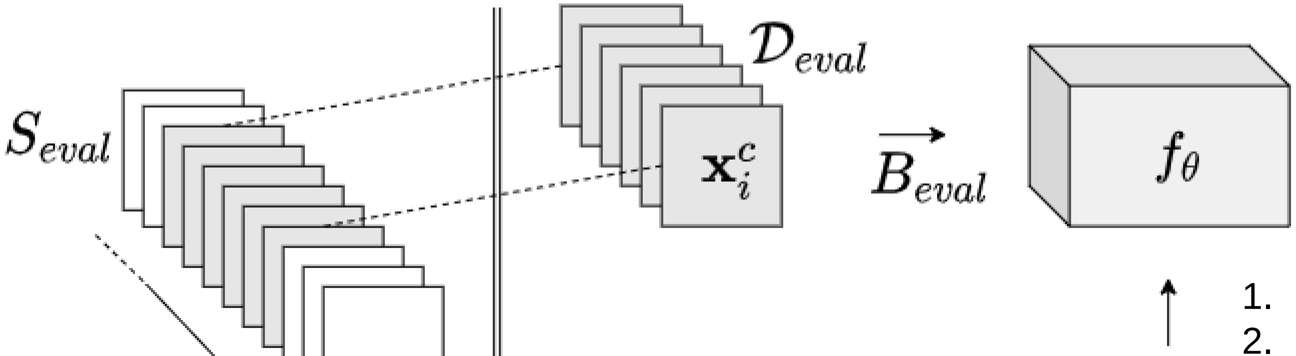
- Memory usage
  - Horizon $\mathcal{D}$ → Dependent setup/application
  - Operational Memory $\mathcal{M}$ → Dependent CL algorithm

Learner-evaluator framework

The *Learner*

The *Evaluator*

The *Evaluator*

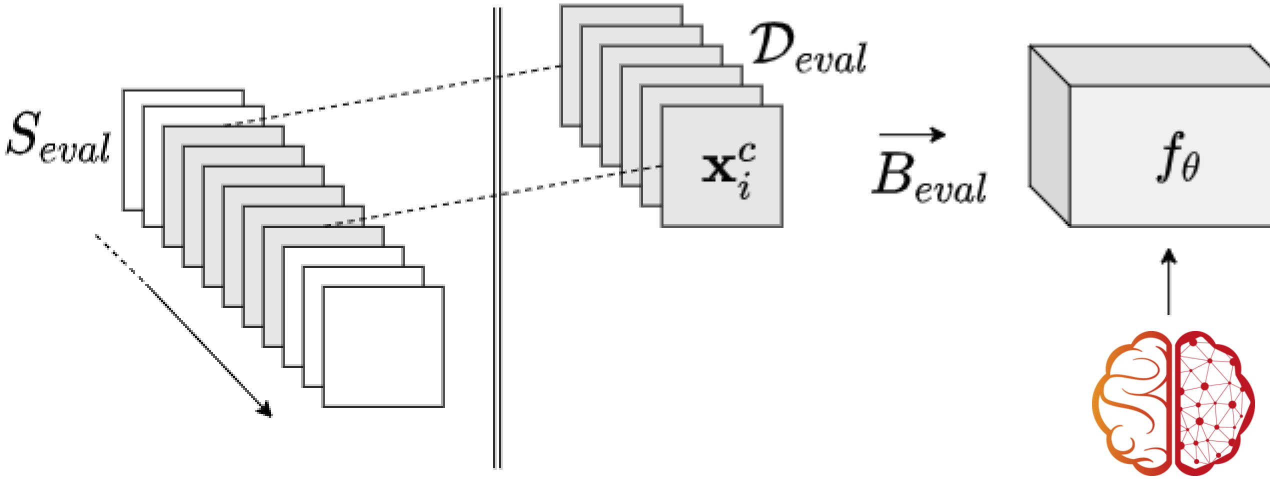


1. Simultaneously
2. Independently

The *Learner*

- Evaluation data stream S_{eval}
- Evaluating horizon $\mathcal{D}_{eval}$, e.g. subset of seen concepts in S_{eval}
- Evaluate $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$
  - Asynchronously on-demand
  - Or with periodicity ρ

The *Evaluator*



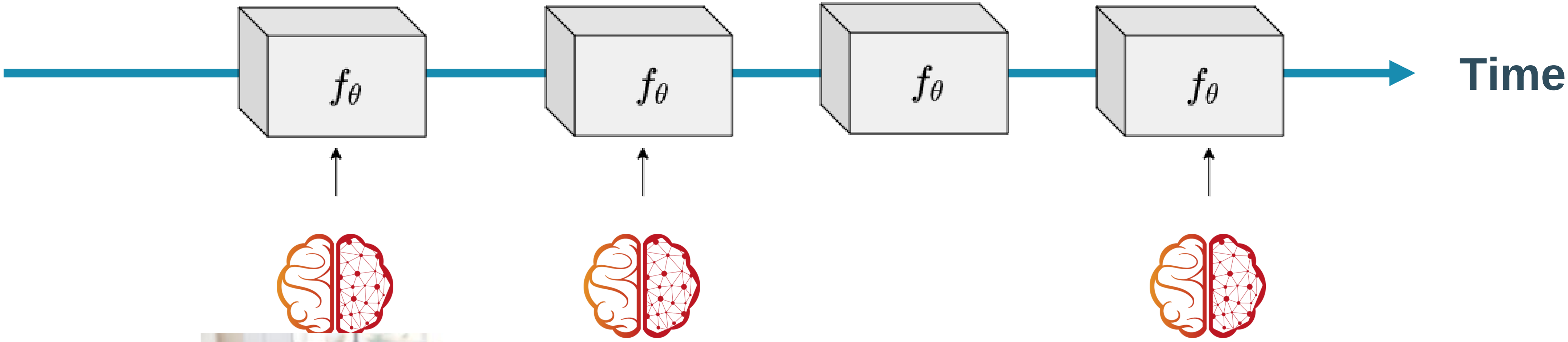
The *Learner*

- Concept distributions in S_{eval} can be
  - Static → Measure forgetting in CL
  - Dynamic → Concepts drift, performance on current distribution?

Questions? matthias.delange@kuleuven.be

Rewind:
Concept drift

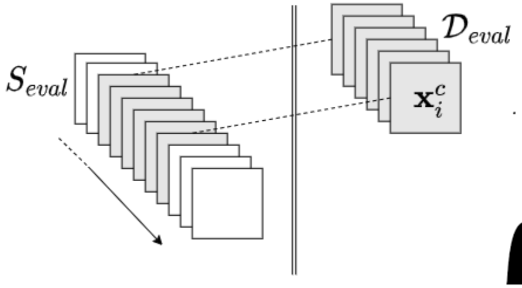
Cat identification system

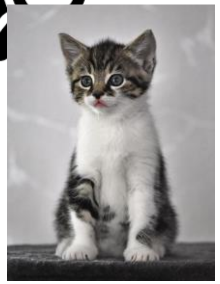


Concept: Felix

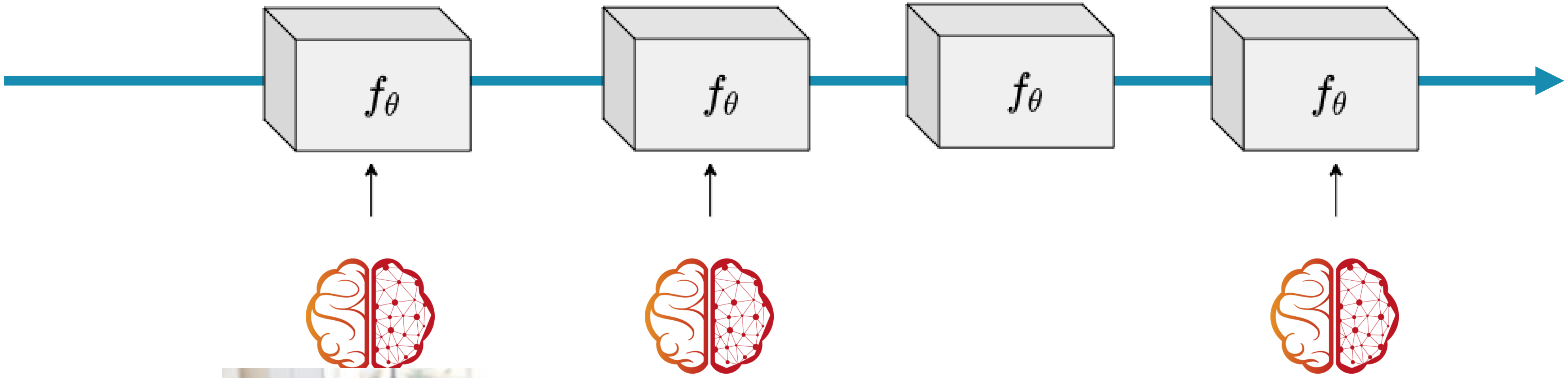
Learn about some other cats

Learn about some other cats



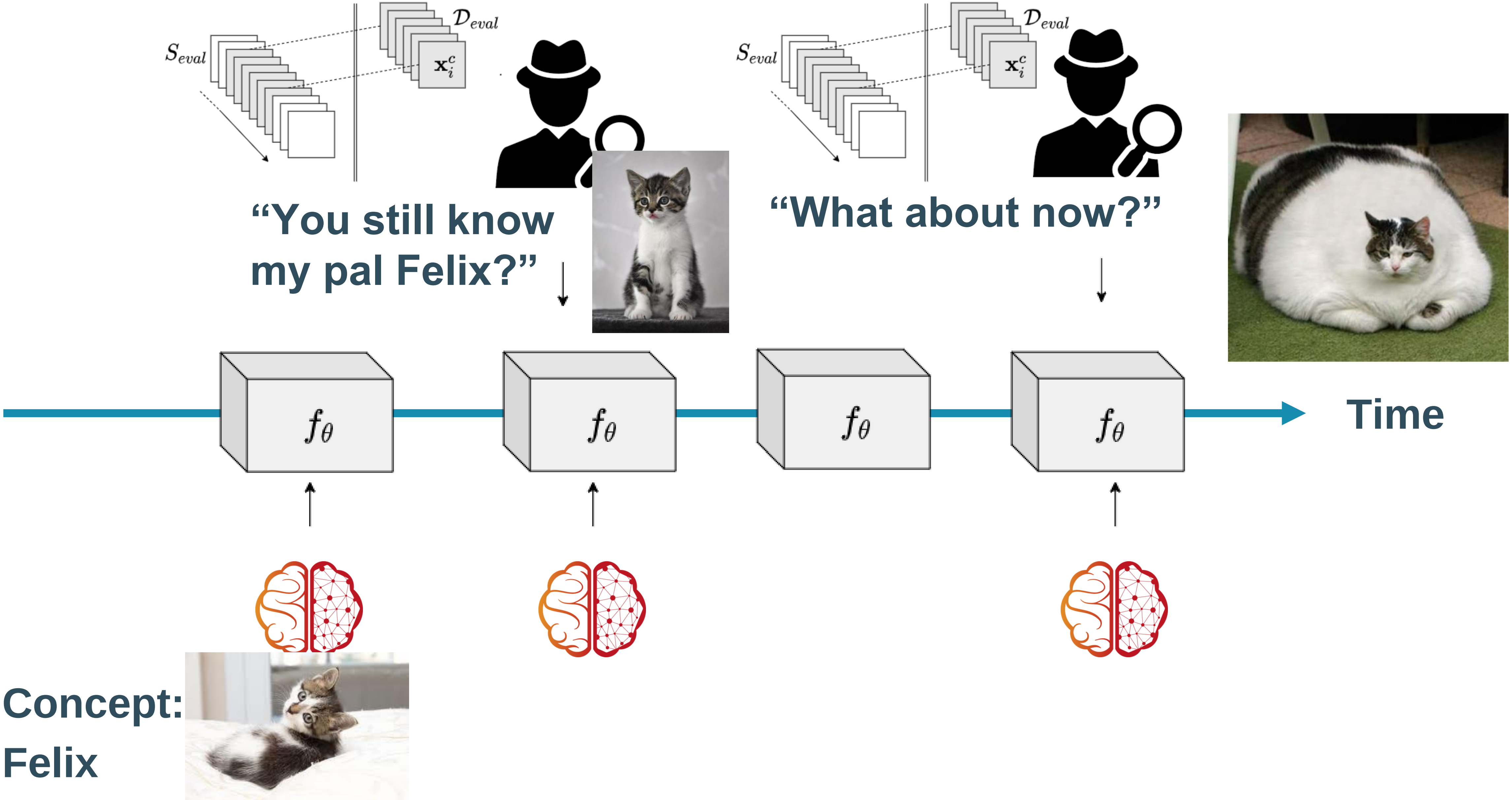


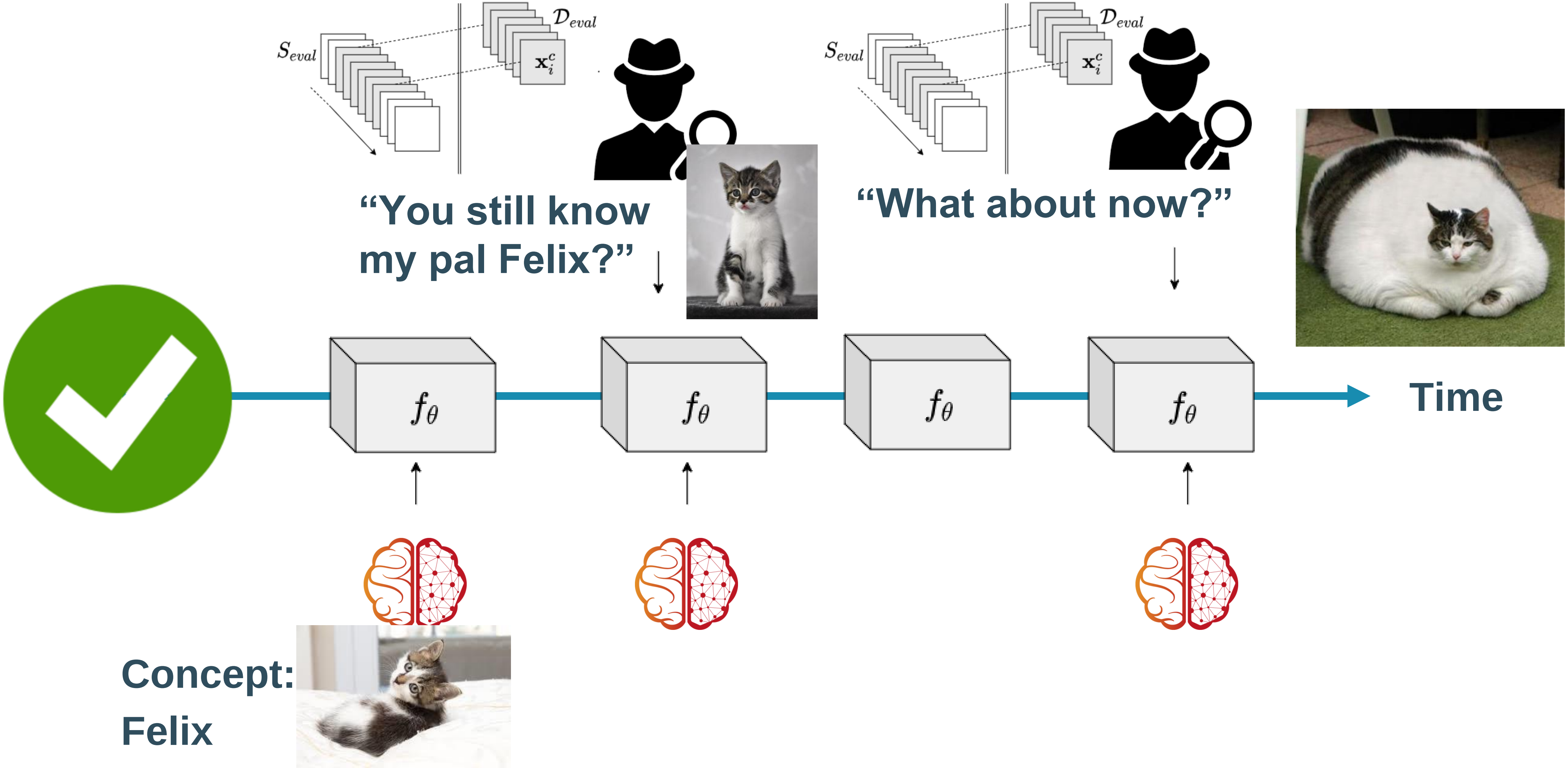
“You still know my pal Felix?”



Time

Concept: Felix


S_{eval}
$\mathcal{D}_{eval}$
$\mathbf{x}_i^c$
“You still know my pal Felix?”
“What about now?”
f_θ
Time
Concept: Felix



S_{eval}
$\mathcal{D}_{eval}$
$\mathbf{x}_i^c$
"You still know my pal Felix?"
"What about now?"
f_θ
f_θ
f_θ
f_θ
Time
Concept:
Felix

Roadmap

- Learner-Evaluator framework
- **Data incremental learning**
- Prior Work
- CoPE
  - Evolving prototypes
  - Balanced replay
  - PPP-loss
- Future work

Data incremental learning

- Redefine current paradigms:

  When is the horizon $\mathcal{D}_t$ replaced? → Application/setup

	information presented to	
	learner	**evaluator**
task incremental	$(\mathbf{x}_i, \mathbf{y}_i, t_i)$	$(\mathbf{x}_i, \mathbf{y}_i, t_i)$
class incremental	$(\mathbf{x}_i, \mathbf{y}_i, t_i)$	$(\mathbf{x}_i, \mathbf{y}_i)$
domain incremental	$(\mathbf{x}_i, \mathbf{y}_i, t_i)$	$(\mathbf{x}_i, \mathbf{y}_i)$
data incremental	$(\mathbf{x}_i, \mathbf{y}_i)$	$(\mathbf{x}_i, \mathbf{y}_i)$

→ Task transitions

→ Class-subset transitions

→ Domain transitions

→ Data stream subsets, no assumptions

Roadmap

- Learner-Evaluator framework
- Data incremental learning
- **Prior Work**
- CoPE
  - Evolving prototypes
  - Balanced replay
  - PPP-loss
- Future work

Prior Work

- Online data incremental learning ($\mathcal{D} = B$)
  - Replay: Reservoir, GSS , MIR
  - Parameter isolation methods: CURL, CN-DPM

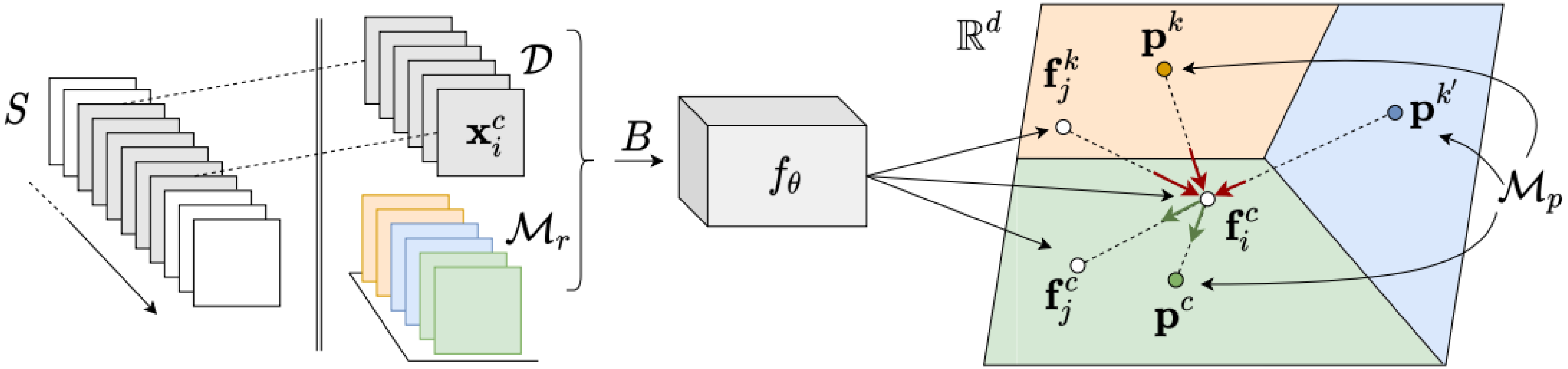
- Class incremental: iCaRL, GEM

Roadmap

- Learner-Evaluator framework
- Data incremental learning
- Prior Work
- **CoPE**
  - Evolving prototypes
  - Balanced replay
  - PPP-loss
- Future work

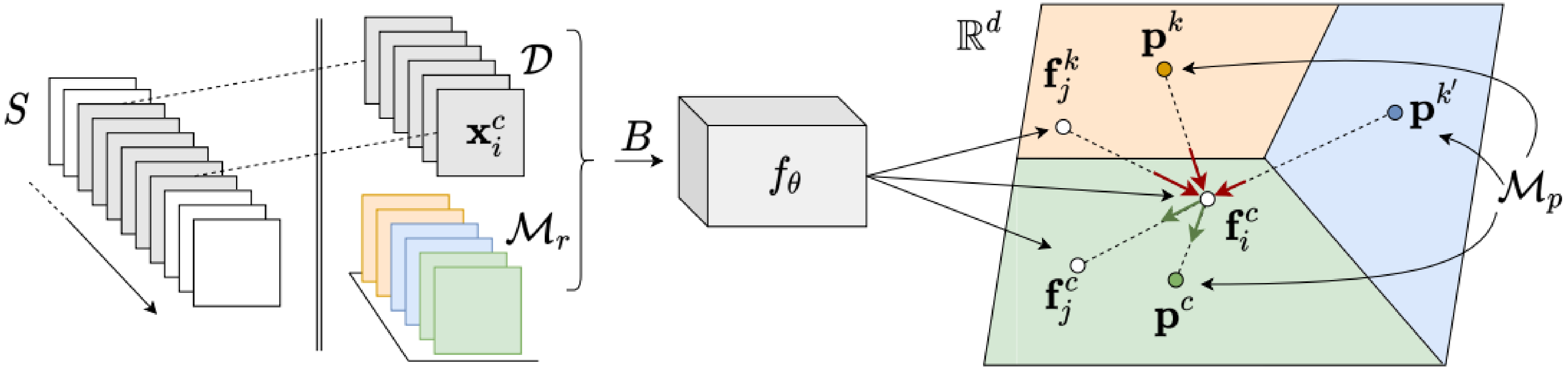
CoPE: Continual Prototype Evolution

- Operates
  - Online
  - Data incremental
  - Imbalanced data



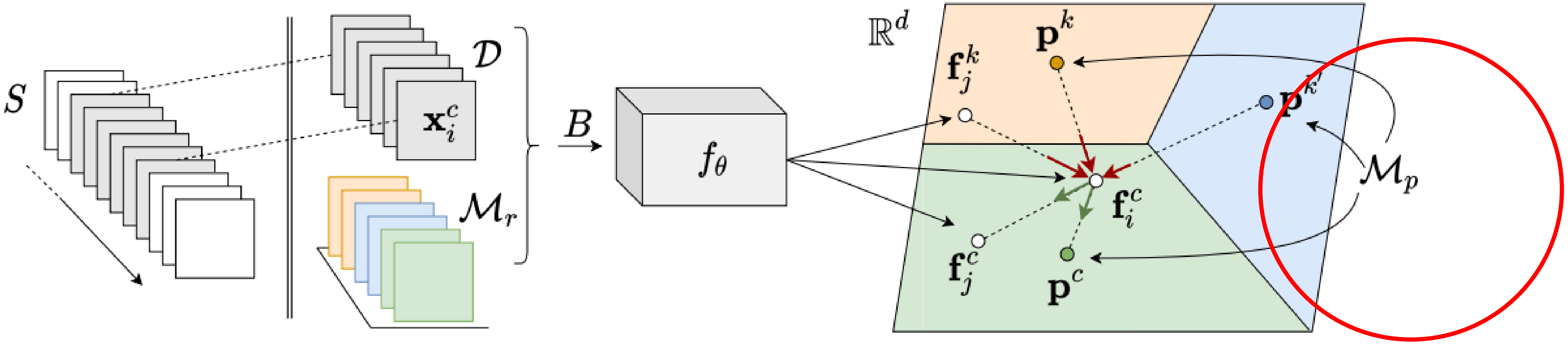
CoPE: Continual Prototype Evolution

- 3 components
  - continually evolving prototypes
  - Balanced replay
  - Pseudo-prototypical proxy loss (PPP-loss)



CoPE: Component 1, Prototypes

Prototypical Memory



CoPE: Component 1, Prototypes

- Prototypes → Nearest Neighbour classifier

- CL literature: recalculated <u>on task transitions</u> with the FULL memory
  ✗ Exhaustive recalculation
  ✗ Dependent on task transitions
  ✗ Static and outdated between task transitions!

- CoPE updates <u>online batch-wise</u> with high momentum
  ✓ Low resource usage
  ✓ Only dependent batch transition
  ✓ Always representative!

CoPE: Component 1, Prototypes

- CoPE updates online batch-wise with high momentum

$$\mathbf{p}^c \leftarrow \alpha \mathbf{p}^c + (1-\alpha)\bar{\mathbf{p}}^c, \text{ s.t.} \quad \bar{\mathbf{p}}^c = \frac{1}{|B^c|} \sum_{\mathbf{x}^c \in B^c} f_\theta(\mathbf{x}^c)$$

	Prototype Momentum			
	0.1	0.9	0.95	0.99
Split-MNIST	93.49 ± 0.70	94.11 ± 0.34	93.96 ± 0.30	93.94 ± 0.20
Split-CIFAR10	44.48 ± 3.19	48.02 ± 2.49	47.98 ± 3.14	48.92 ± 1.32
Split-CIFAR100	15.79 ± 1.16	21.62 ± 0.69	21.56 ± 0.58	20.01 ± 1.81

CoPE: Component 1, Prototypes

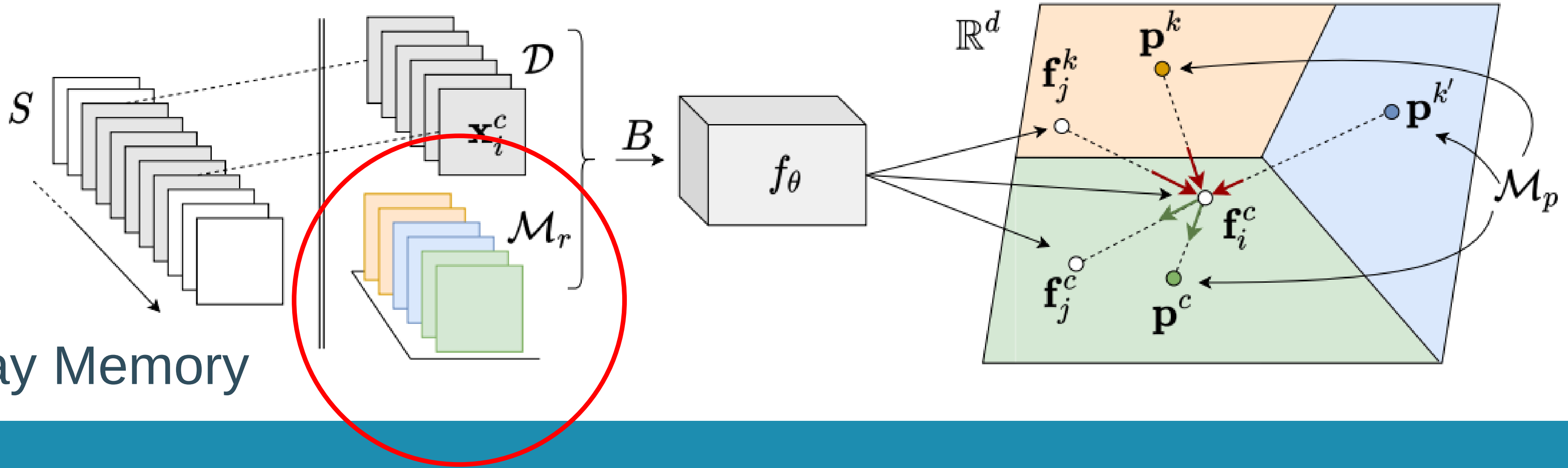
- But, how do the prototypes remain representative?

  → Ever evolving latent space with each update

  → Non-stationary data → Catastrophic forgetting

- Other 2 components:
  - Balanced replay
  - PPP-loss

CoPE: Component 2, Balanced replay

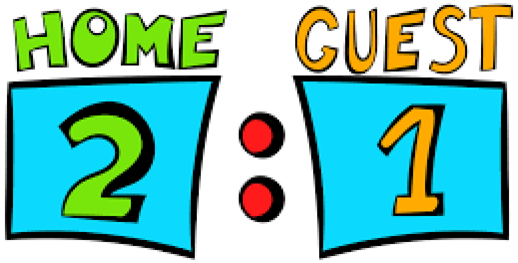


Replay Memory

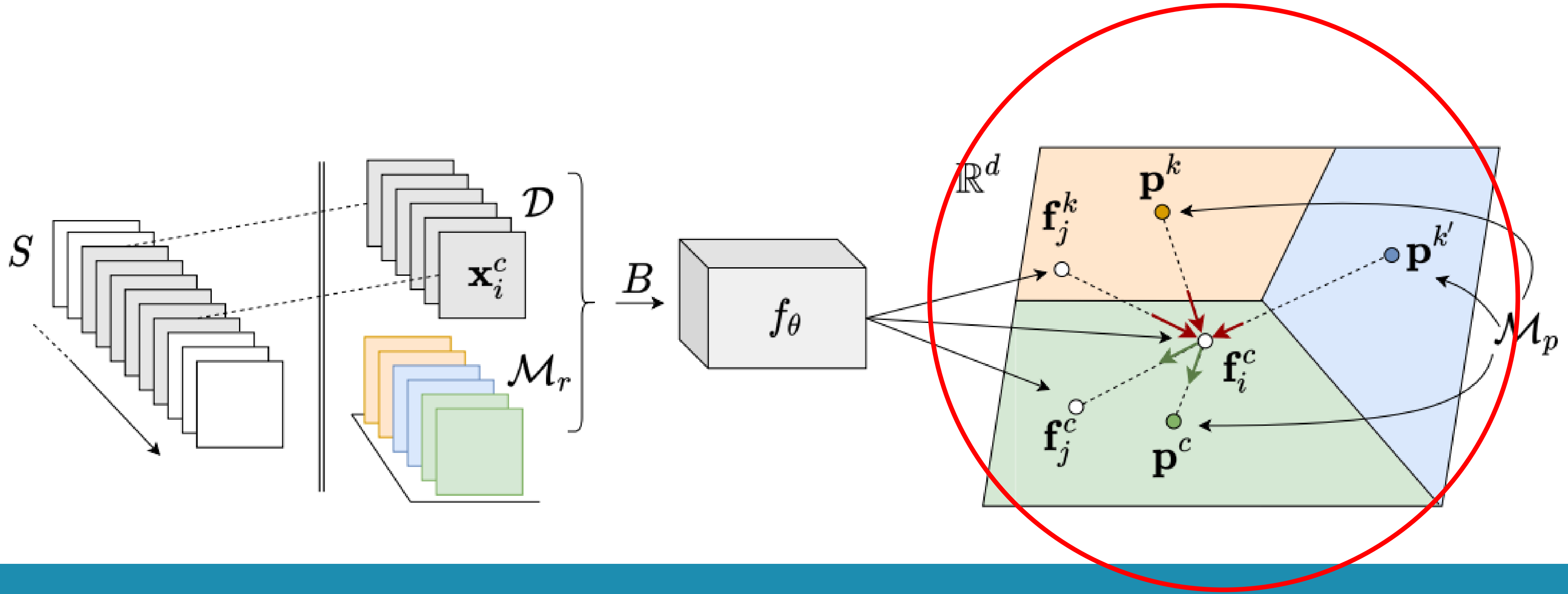
CoPE: Component 2, Balanced replay

- Prior: deem each class equally important
  - **Storage**: Dynamic class memory $\mathcal{M}_r^c$ based on reservoir sampling
  - **Easy Retrieval**: Uniform = class-balanced batch
    → Keeps all class-prototypes up-to-date

- Replay benefits:
  1. Combat catastrophic forgetting
  2. Latent batch information for all classes



CoPE: Component 3, PPP-loss



CoPE: Component 3, PPP-loss

- Pseudo-Prototypical Proxy loss

- Batch *B* not only gives supervision about instance category
  → Also relational information in the latent space!

- Construct per instance, one-against-all subsets:

$$B^c = \{(\mathbf{x}_i, y_i = c) \in B\} \quad \text{and} \quad B^k$$

CoPE: Component 3, PPP-loss

- Construct per instance, one-against-all subsets:

$B^c = \{(\mathbf{x}_i, y_i = c) \in B\}$ and B^k

- Define prototype proxy sets:
  - Attractor $\mathbb{P}_i^c = \{\mathbf{p}^c\} \cup \{\hat{\mathbf{p}}_j^c = f_\theta(\mathbf{x}_j^c) \mid \forall \mathbf{x}_j^c \in B^c, \, i \neq j\}$
  - Repellor $\mathbb{U}_i^c = \{\mathbf{p}^c, \, \hat{\mathbf{p}}_i^c = f_\theta(\mathbf{x}_i^c)\}$

Pseudo-prototypes

CoPE: Component 3, PPP-loss

$$P_i = P(c|\mathbf{x}_i^c) \prod_{\mathbf{x}_j^k} (1 - P_i(c|\mathbf{x}_j^k))$$

Similar to [2], reformulate as binary classification problem.

$$P(c|\mathbf{x}_i^c) = \mathop{\mathbb{E}}_{\tilde{\mathbf{p}}^c \in \mathbb{P}_i^c} \left[P(c|\mathbf{f}_i^c, \tilde{\mathbf{p}}^c)\right], \quad P_i(c|\mathbf{x}_j^k) = \mathop{\mathbb{E}}_{\tilde{\mathbf{p}}^c \in \mathbb{U}_i^c} \left[P(c|\mathbf{f}_j^k, \tilde{\mathbf{p}}^c)\right]$$

[2] Ye, Mang, et al. "Unsupervised embedding learning via invariant and spreading instance feature." *CVPR* 2019.

CoPE: Component 3, PPP-loss

$$P_i = P(c|\mathbf{x}_i^c) \prod_{\mathbf{x}_j^k} (1 - P_i(c|\mathbf{x}_j^k))$$

$$P(c|\mathbf{x}_i^c) = \mathop{\mathbb{E}}_{\tilde{\mathbf{p}}^c \in \mathbb{P}_i^c} \left[P(c|\mathbf{f}_i^c, \tilde{\mathbf{p}}^c) \right], \quad P_i(c|\mathbf{x}_j^k) = \mathop{\mathbb{E}}_{\tilde{\mathbf{p}}^c \in \mathbb{U}_i^c} \left[P(c|\mathbf{f}_j^k, \tilde{\mathbf{p}}^c) \right]$$

with $\tilde{\mathbf{p}}^c$ a proxy for the latent mean of class c in

$$P(c|\mathbf{f}, \tilde{\mathbf{p}}^c) = \frac{\exp(\mathbf{f}^T \tilde{\mathbf{p}}^c / \tau)}{\exp(\mathbf{f}^T \tilde{\mathbf{p}}^c / \tau) + \sum_{k \neq c} \exp(\mathbf{f}^T \mathbf{p}^k / \tau)}$$

CoPE: Component 3, PPP-loss

$$P_i = P(c|\mathbf{x}_i^c) \prod_{\mathbf{x}_j^k} (1 - P_i(c|\mathbf{x}_j^k))$$

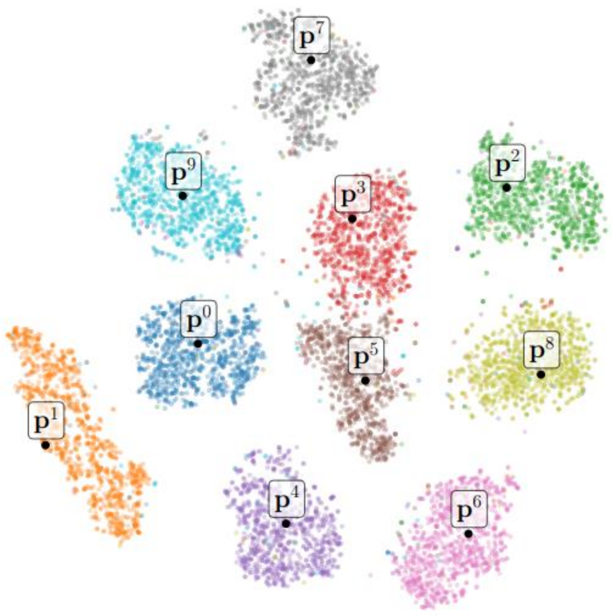
$$\downarrow$$

$$\mathcal{L} = -\frac{1}{|B|} \left[\sum_i \log P(c|\mathbf{x}_i^c) + \sum_i \sum_{\mathbf{x}_j^k} \log(1 - P_i(c|\mathbf{x}_j^k)) \right]$$

PPP-loss ablation

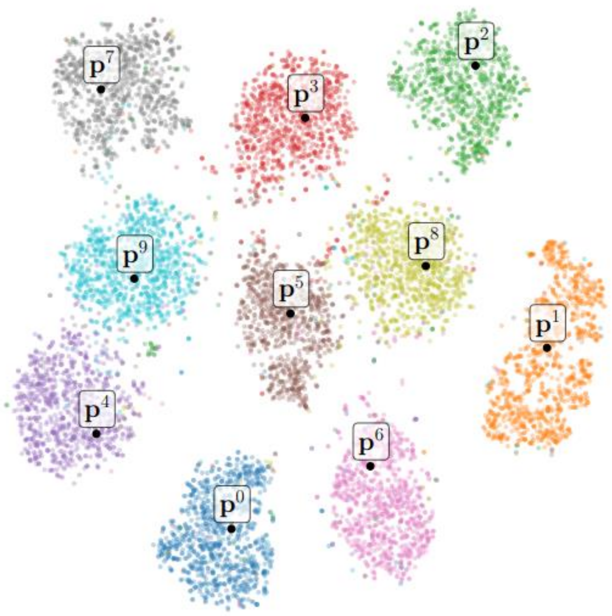
- Including/excluding pseudo-prototypes in PPP-loss

94.52%



(a) PPP-loss – *incl.* $\hat{\mathbf{p}}$

90.86%



(b) PPP-loss – *excl.* $\hat{\mathbf{p}}$

Optimal prototypes?

- We approximate the mean of the latent distribution
  → Optimal for Bregman divergences

$$d_\varphi(\mathbf{f}_i, \mathbf{f}_j) = \varphi(\mathbf{f}_i) - \varphi(\mathbf{f}_j) - (\mathbf{f}_i - \mathbf{f}_j)^T \nabla\varphi(\mathbf{f}_j)$$

- E.g. squared Euclidian distance $\varphi(\mathbf{f}) = ||\mathbf{f}||^2$

- We constrain $||\mathbf{f}_i|| = ||\mathbf{f}_j|| = 1$, resulting in $\frac{1}{2}||\mathbf{f}_i - \mathbf{f}_j||^2 = 1 - \cos\angle(\mathbf{f}_i, \mathbf{f}_j)$
- We need similarity → $\cos\angle(\mathbf{f}_i, \mathbf{f}_j) = \mathbf{f}_i^T \mathbf{f}_j$

Experiments

- Learner:
  - Online processing with |B|=10
  - S subdivided in task-like sequence (to compare with iCaRL/GEM)
    → CoPE learner is unaware of this! (not provided)

- Evaluator:
  - held-out dataset of static concepts in S_eval, evaluating with the subset of seen concepts Y in D_eval using the accuracy metric.

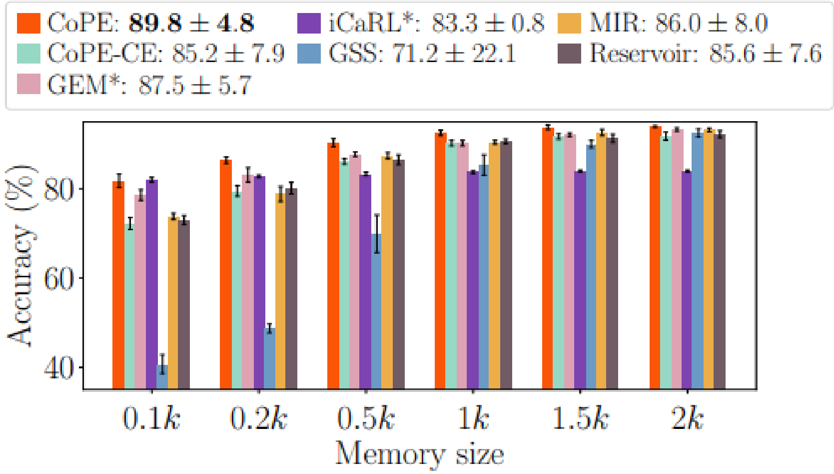
Questions? matthias.delange@kuleuven.be

Balanced data streams

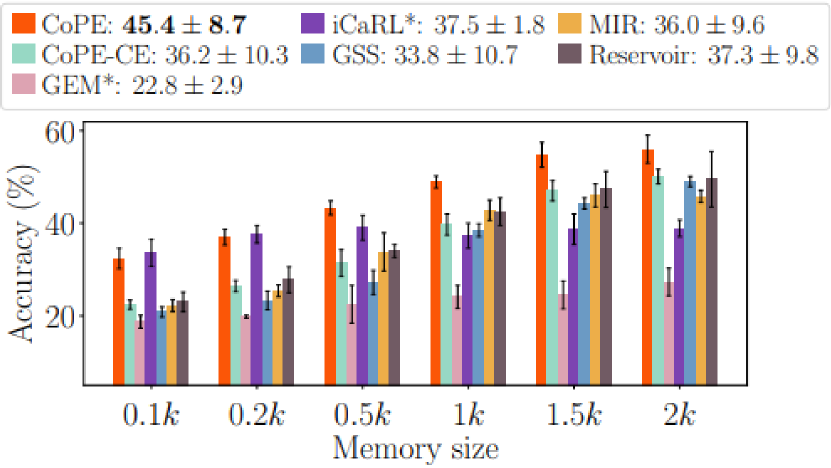
	Split-MNIST	Split-CIFAR10	Split-CIFAR100
iid-offline	98.44 ± 0.02	83.02 ± 0.60	50.28 ± 0.66
iid-online	96.57 ± 0.14	62.31 ± 1.67	20.10 ± 0.90
finetune	19.75 ± 0.05	18.55 ± 0.34	3.53 ± 0.04
GEM	93.25 ± 0.36	24.13 ± 2.46	11.12 ± 2.48
iCARL	83.95 ± 0.21	37.32 ± 2.66	10.80 ± 0.37
CURL (Rao et al., 2019)	92.59 ± 0.66	–	–
DN-CPM (Lee et al., 2020)	93.23 ± 0.09	45.21 ± 0.18	20.10 ± 0.12
reservoir	92.16 ± 0.75	42.48 ± 3.04	19.57 ± 1.79
MIR	93.20 ± 0.36	42.80 ± 2.22	20.00 ± 0.57
GSS	92.47 ± 0.92	38.45 ± 1.41	13.10 ± 0.94
CoPE-CE	91.77 ± 0.87	39.73 ± 2.26	18.33 ± 1.52
CoPE (ours)	$\mathbf{93.94 \pm 0.20}$	$\mathbf{48.92 \pm 1.32}$	$\mathbf{21.62 \pm 0.69}$

“Sure dude, but you just tweaked the buffer size?”

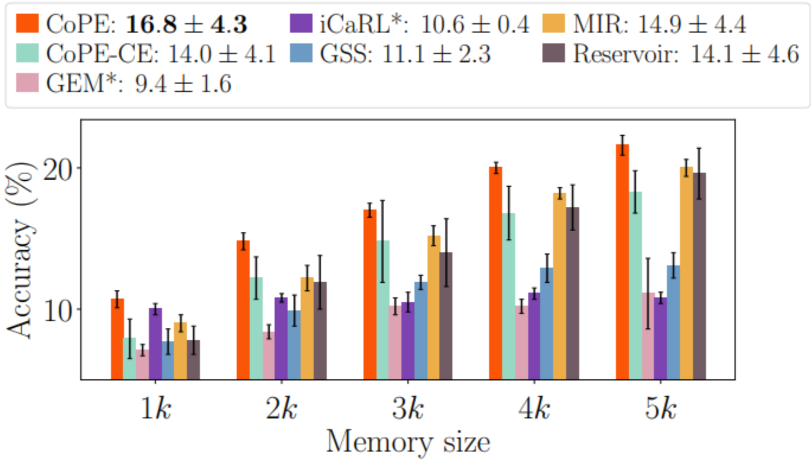
Split-MNIST



Split-CIFAR10



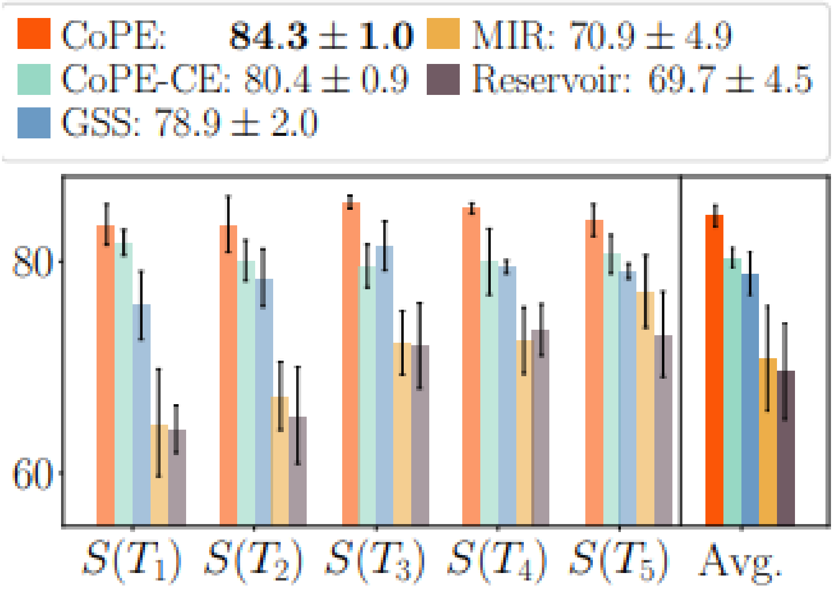
Split-CIFAR100



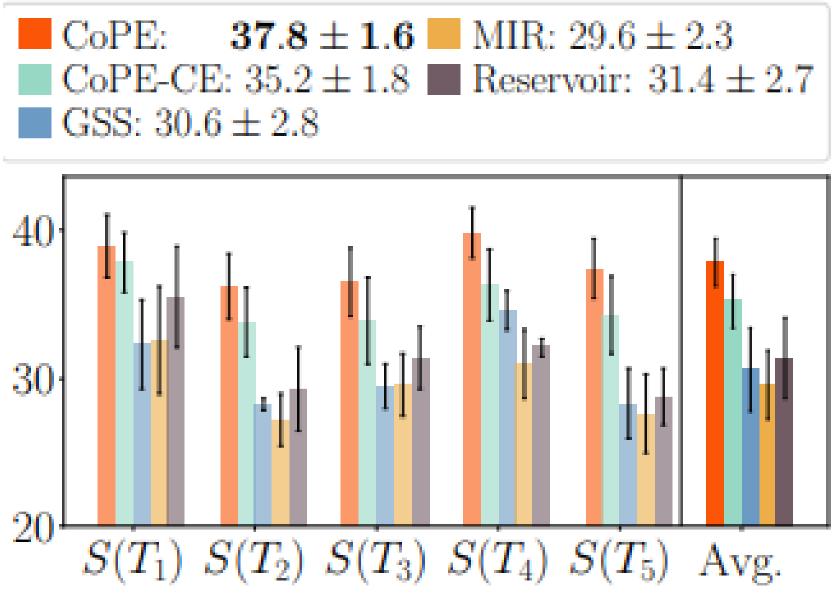
Imbalanced experiments

- Not just the balancing memory scheme (CoPE-CE)
- The PPP-loss encourages prototype-based clusters each update

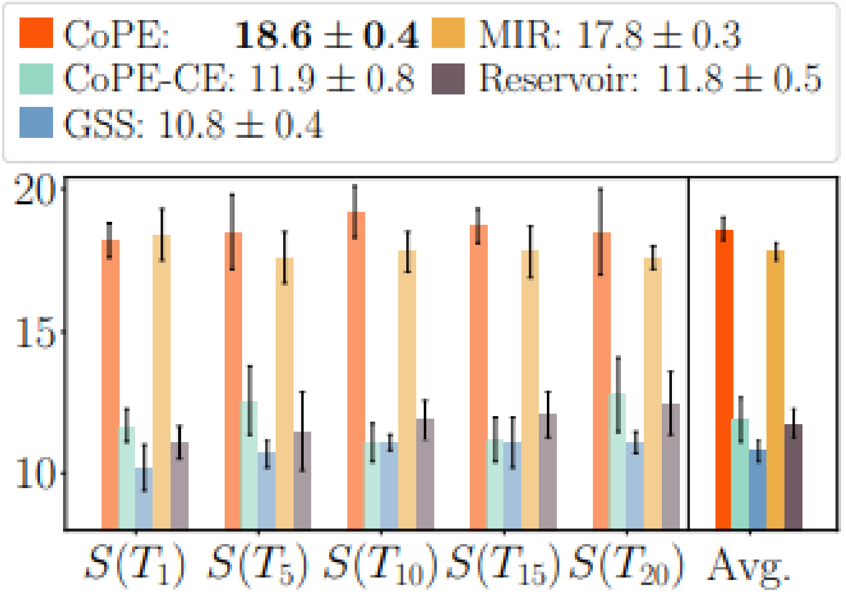
Split-MNIST



Split-CIFAR10



Split-CIFAR100



Summary

- Learner-evaluator framework → 2 agents, horizon (~~task~~), concept drift
- Data incremental learning → Any data stream (~~task info~~)
- CoPE → Online data incremental
  → Continually evolving prototypes
  → Balanced replay
  → PPP-loss
- Future? → Apply for concept drift, beyond classification/supervised learning

Code

https://github.com/mattdl

Questions?

matthias.delange@kuleuven.be