Continual Learning

Gido van de Ven

E-mail: gido.vandeven@kuleuven.be

Website: https://gmvandeven.github.io

INVICTA Spring School, Porto

19 March 2024

Overview

Introduction

➔ What is continual learning?

➔ Why is continual learning difficult?

➔ Why is continual learning important?

Main part

➔ Different types of continual learning – *“the problem”*

➔ Approaches for continual learning – *“the solutions”*

Time-depending

➔ The road forward: open topics in continual learning

Further reading

This lecture is for a large part based on this recent book chapter:

Continual Learning and Catastrophic Forgetting

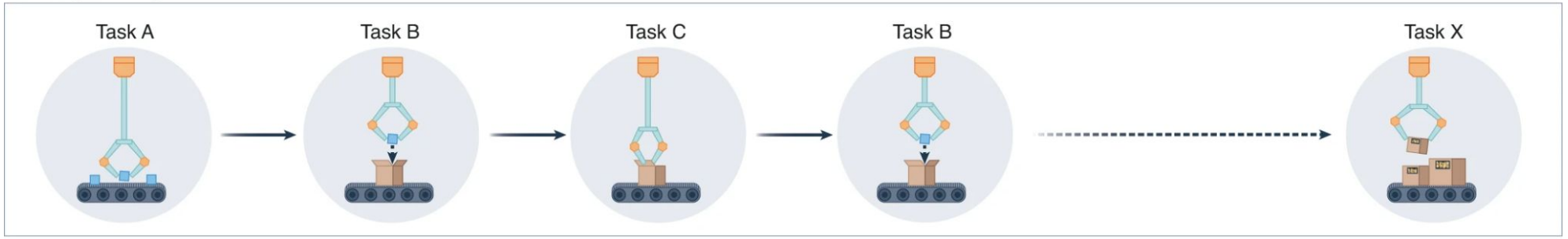
Gido M. van de Ven, Nicholas Soures*, Dhireesha Kudithipudi*

Preprint available at https://arxiv.org/abs/2403.05175

What is continual learning?

What is continual learning?

- Continual learning is the problem of incrementally learning from a non-stationary stream of data
  - Non-stationary: distribution of data from which is learned changes over time
  - Incrementally: new learning should not overwrite what was learned before, but knowledge should be accumulated



Source: Kudithipudi et al. (2022, Nature Machine Intelligence)

What is continual learning?

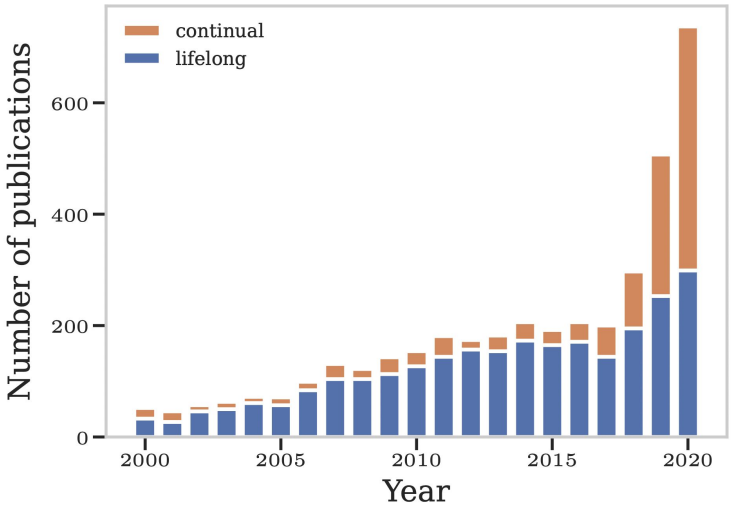
- Continual learning is the problem of incrementally learning from a non-stationary stream of data
  - Non-stationary: distribution of data from which is learned changes over time
  - Incrementally: new learning should not overwrite what was learned before, but knowledge should be accumulated

- Continual learning is a ***key aspect of intelligence***

- Deep neural networks largely lack continual learning abilities, especially compared to their biological counterparts

The term ‘continual learning’

- ‘Continual learning’ *vs*. ‘lifelong learning’
  - Often used interchangeably
  - Popularity of ‘continual learning’ more recent



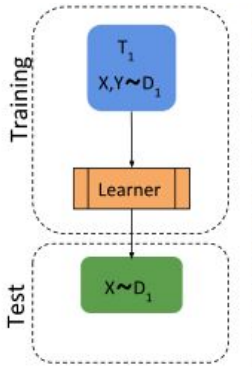
Number of machine learning publications per year, based on keyword occurrence in abstract.
Source: Mundt et al. (2022, ICLR)

- Especially in recent years, the ‘continual learning’ literature tends to have a more narrow focus:
  - Traditional ML: all training data available at same time
  - Continual learning: - training data arrive incrementally
    - there is non-stationarity

These terms seem roughly to be used as follows:
- **Continual learning** *narrow* → how to deal with non-stationarity in training data
- **Lifelong learning** *broad* → everything relevant for an agent learning throughout its lifetime

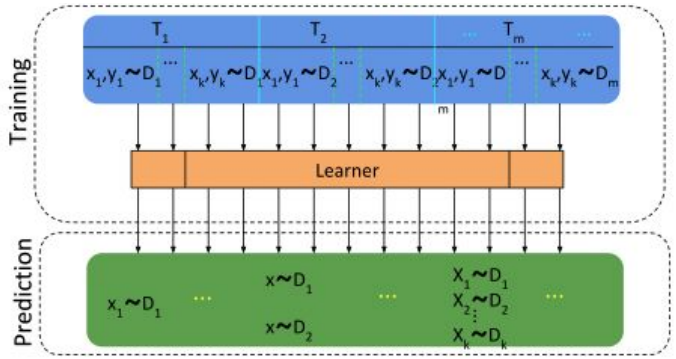
Continual learning in relation to other fields

Standard ML



- One task
- Data available at same time

Continual Learning

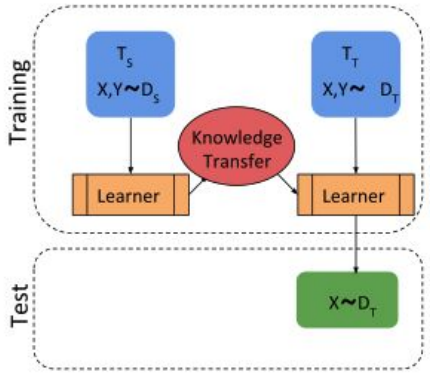


- Multiple tasks
- Data arrive incrementally

} ~~non~~-stationarity

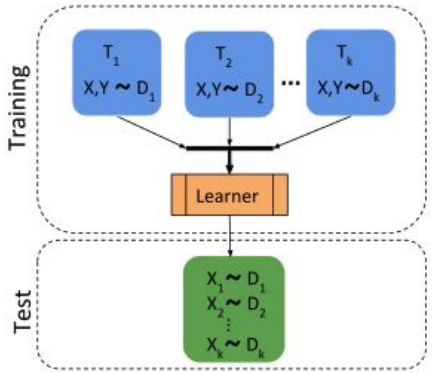
- Goal: all tasks

Transfer Learning



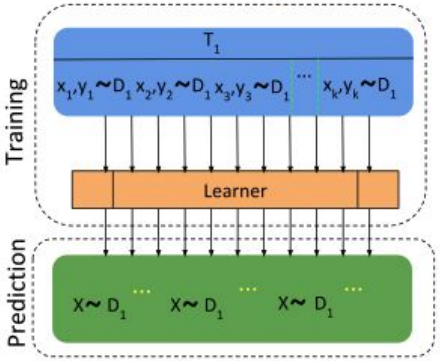
- Multiple tasks
- Data arrive incrementally
- Goal: last task

Multitask Learning



- Multiple tasks
- Data available at same time
- Goal: all tasks

Online Learning

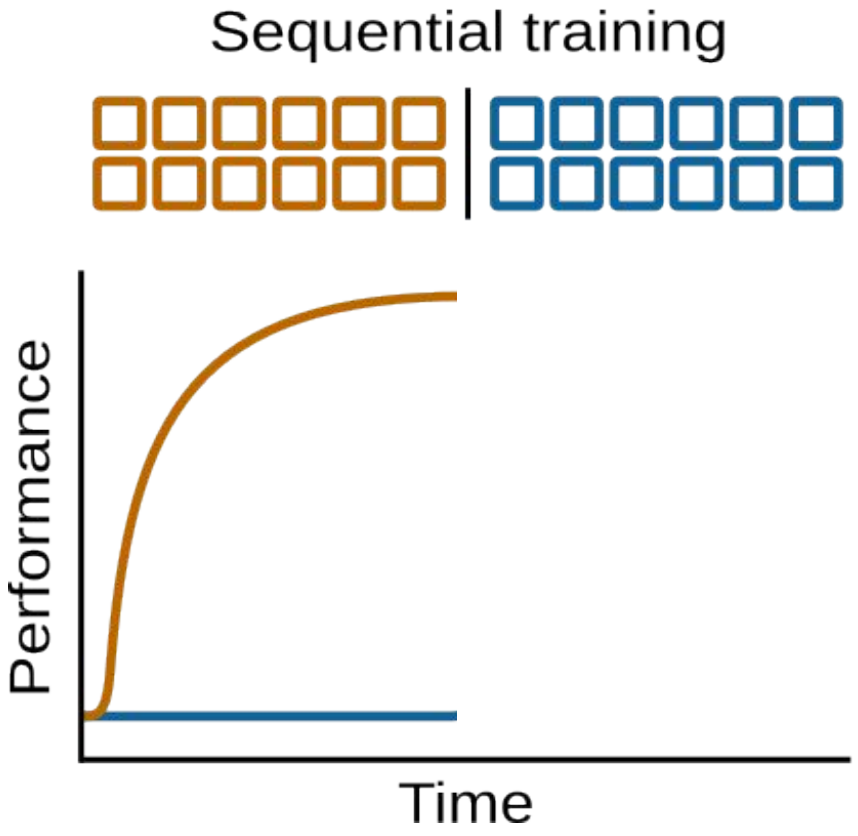


- One task
- Data arrive incrementally

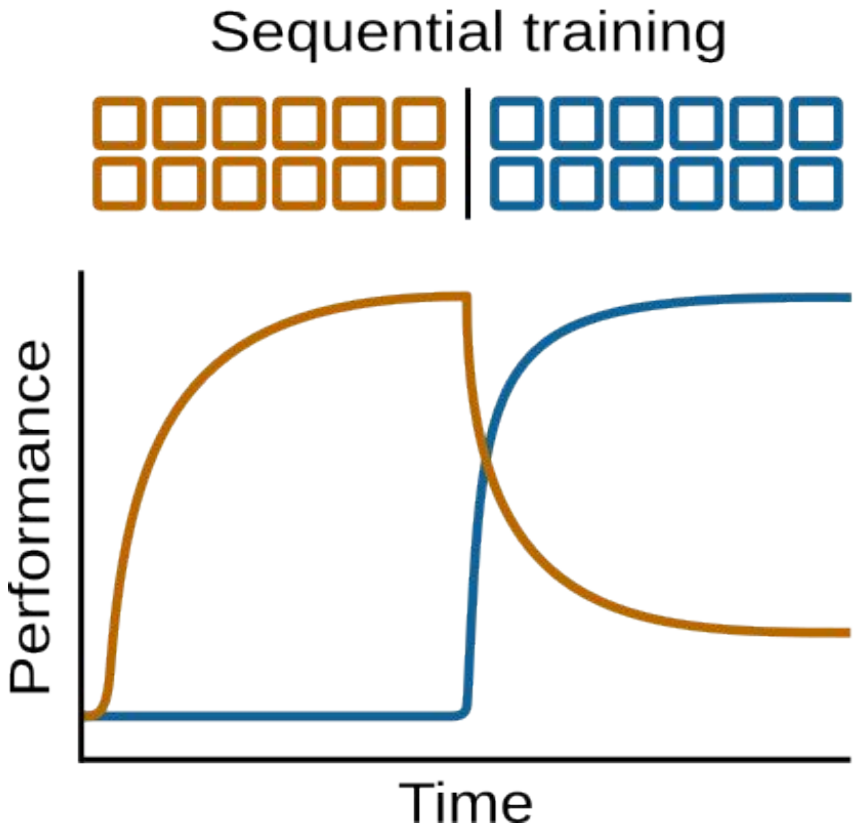
Source: De Lange et al. (2021, TPAMI)

Why is continual learning difficult?

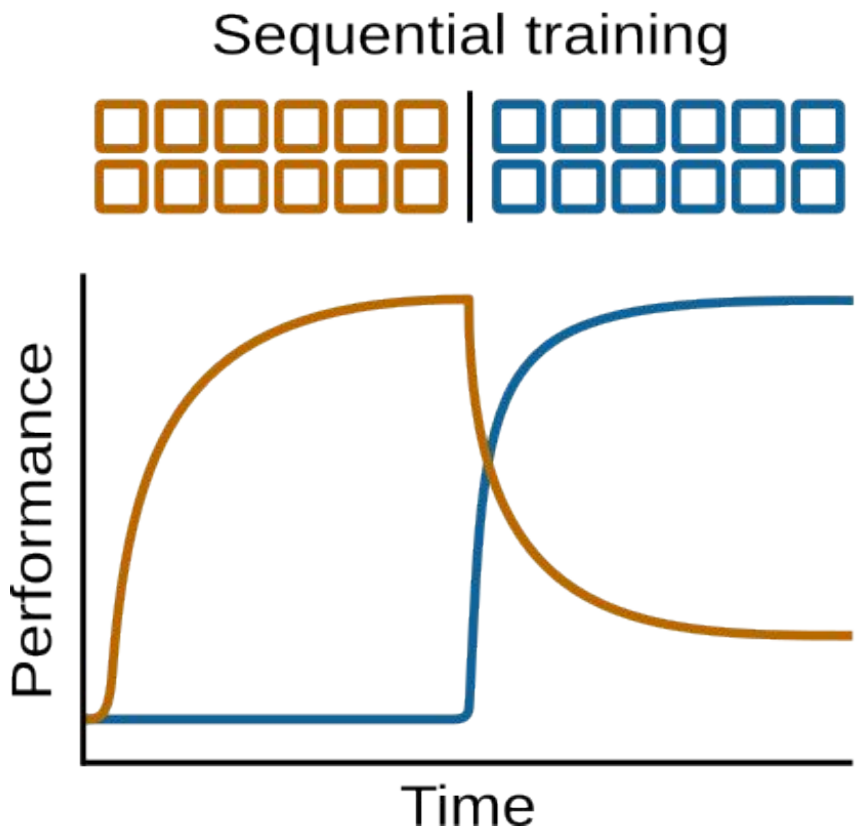
Catastrophic forgetting

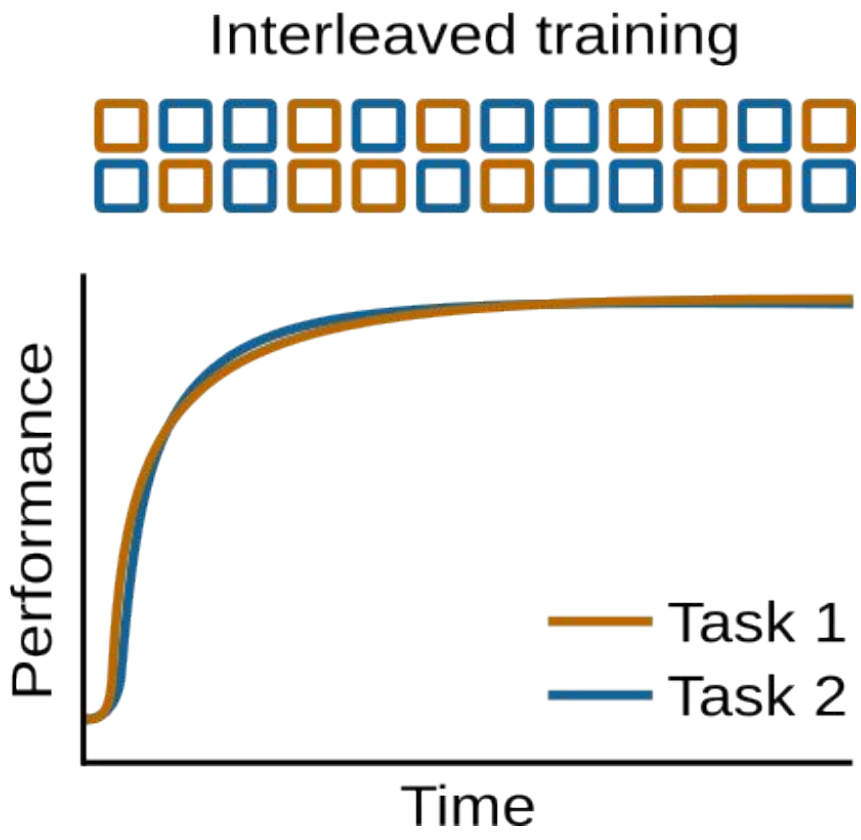


Catastrophic forgetting



Catastrophic forgetting





Catastrophic forgetting: expected or surprising?

- We first optimize the model for one task, and then we optimize it for another task. Why would the optimal solution for the second task still be good for the first one?

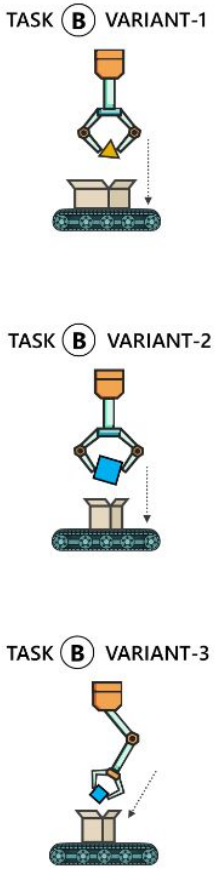
- One reasonable hypothesis:

  ``*If the modifications are all in the direction that helps the pattern that is being stored, there will be a conspiracy effect: The total help for the intended pattern will be the sum of all the small separate modifications. For unrelated patterns, however, there will be very little transfer of effect because some of the modifications will help and some will hinder. Instead of all the small modifications conspiring together, they will mainly cancel out.''* (pp. 81-82; Hinton et al., 1986)

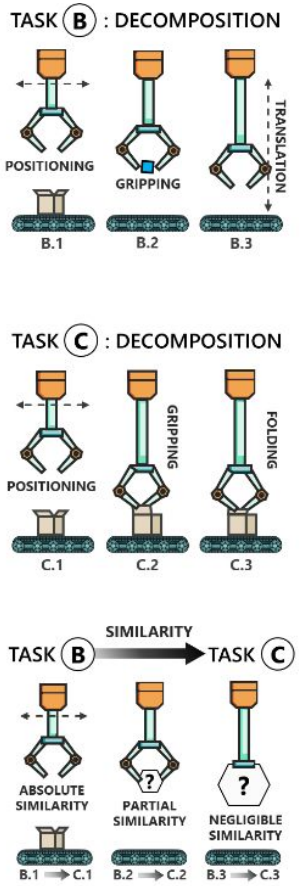
- McCloskey and Cohen (1989) and Ratcliff (1990) were the first to demonstrate catastrophic forgetting

Other features important for continual learning

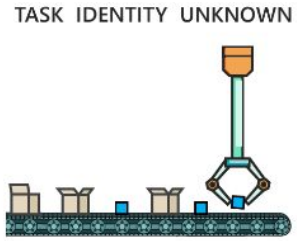
Rapid adaptation



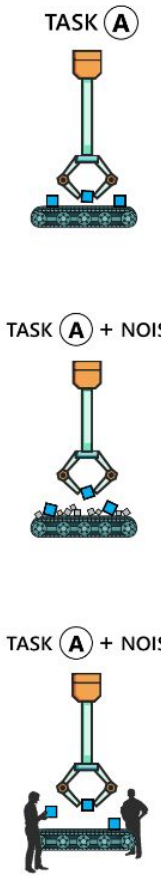
Exploiting task similarity



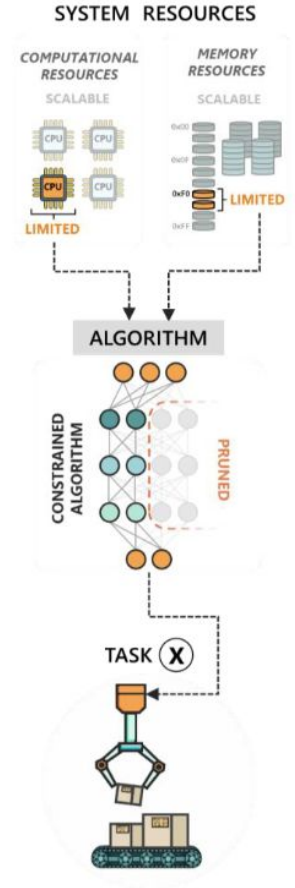
Task agnosticity



Noise tolerance



Resource efficiency and sustainability



Source: Kudithipudi et al. (2022, Nature Machine Intelligence)

Goal of continual learning

on both old and new tasks

e.g., in terms of memory and computation

Achieve an optimal trade-off between ***performance*** and ***resource efficiency***

Fine-tuning
Only train the model on the new data

- Bad performance
- Good resource efficiency

Continual learning strategies

Full retraining
Always fully train the model on all data so far

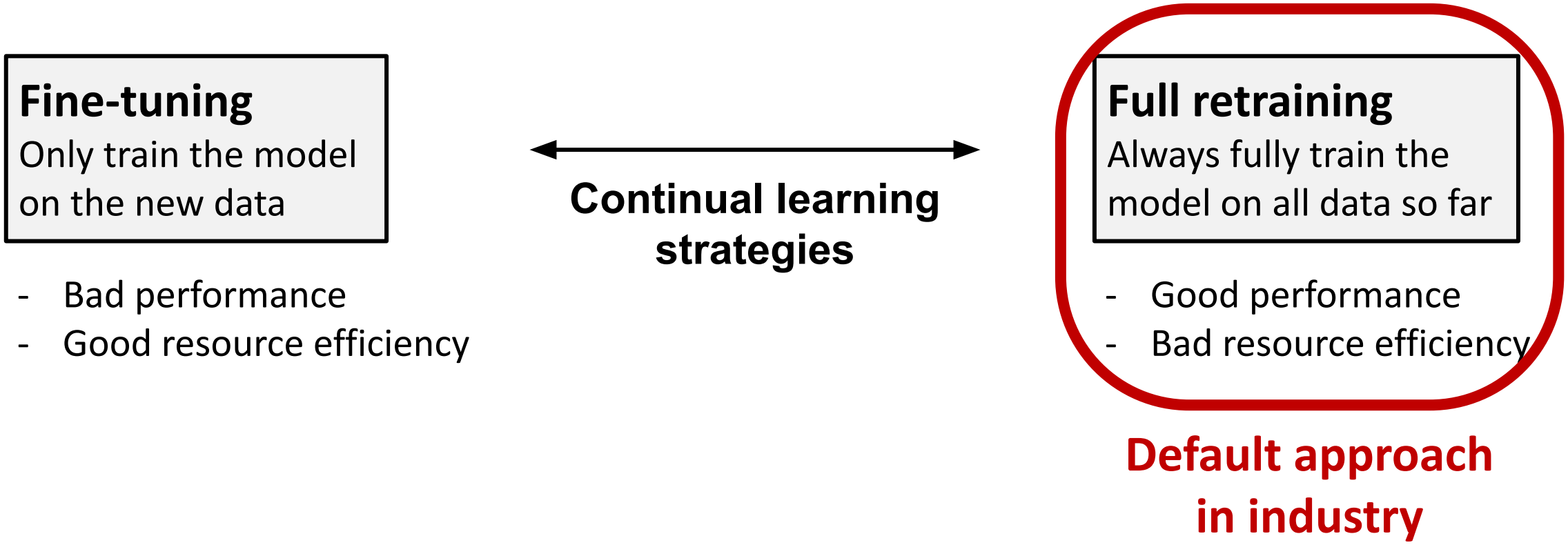
- Good performance
- Bad resource efficiency

Source: Verwimp et al. (2023, arXiv)

Why is continual learning important?

Potential applications of continual learning

1. Improve efficiency and substantially reduce required resources



Potential applications of continual learning

1. Improve efficiency and substantially reduce required resources

2. Correct mistakes or biases in trained networks (adapting locally)

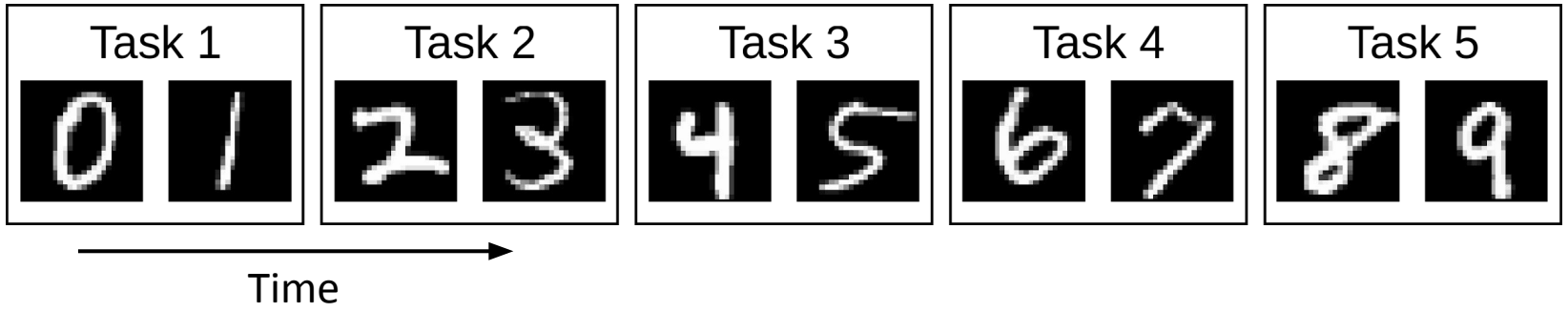
Potential applications of continual learning

1. Improve efficiency and substantially reduce required resources
2. Correct mistakes or biases in trained networks (adapting locally)
3. On-device learning (e.g., personalization)

Different types of continual learning

The canonical continual learning example: Split MNIST

- MNIST dataset is split in multiple tasks(*) that must be learned sequentially
- After all tasks have been learned, the model should be good at all tasks
- Typically, no or only a small amount of data from past tasks can be stored



(*) Sometimes the term “context” or “experience” is used instead of “task”, although there can be subtle differences between these terms.

Going beyond Split MNIST

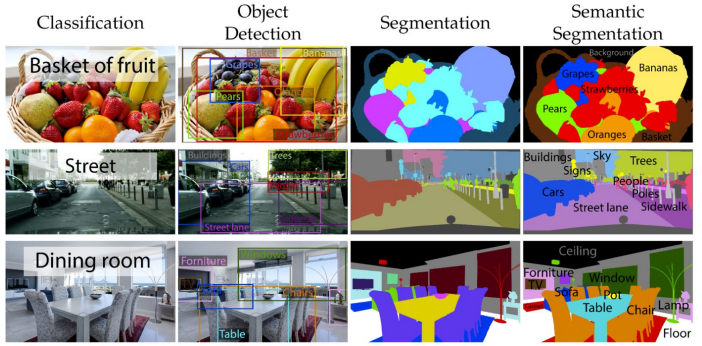
- Splitting up existing image datasets:
  - CIFAR-10
  - CIFAR-100
  - (Tiny)ImageNet
  - …

Source: van de Ven et al. (2020, Nature Communications)

- Datasets specific for continual learning:
  - CORe50
  - Stream-51
  - The CLEAR Benchmark
  - …

Source: Lin et al. (2021, NeurIPS Datasets and Benchmarks Track)

- Beyond classification:
  - Continual reinforcement learning
  - Continual object detection
  - Continual semantic segmentation
  - …



Source: Toldo et al. (2020, Technologies)

CORe50: different types of continual learning

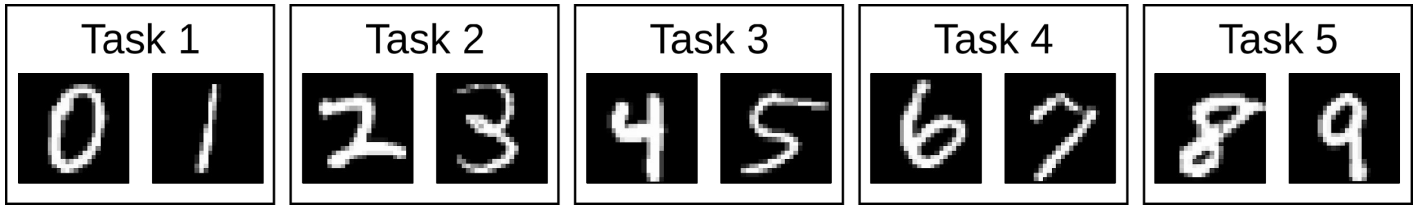
New classes

New instances

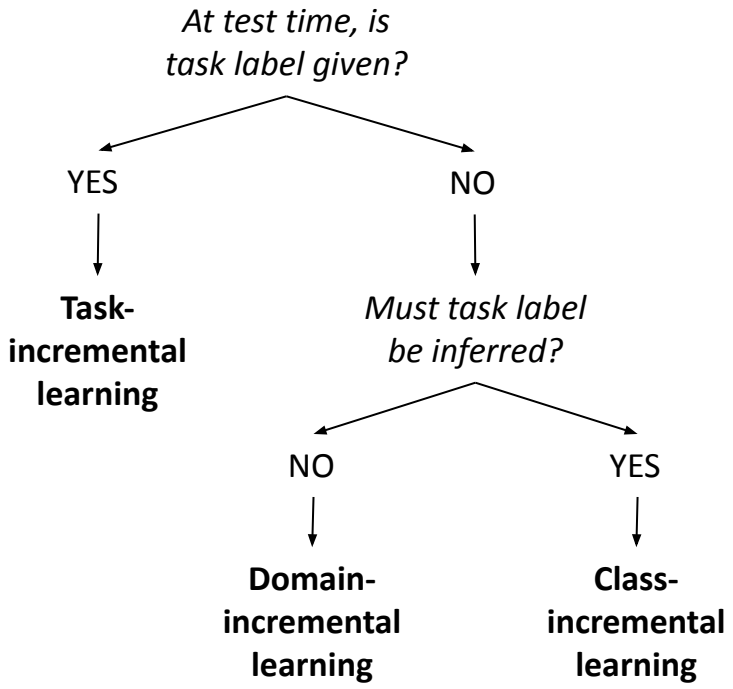
Source: Lomonaco & Maltoni (2017, CoRL)

Back to MNIST: three continual learning scenarios

Split MNIST:



	Type of choice
Task-incremental	Choice between the two digits of the task
Domain-incremental	Is the digit odd or even?
Class-incremental	Choice between all ten digits

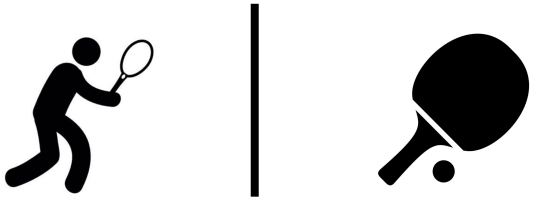


Source: van de Ven & Tolias (2018, NeurIPS workshop)

Three continual learning scenarios: intuitively

- Task-incremental learning *(Task-IL)*
  - Incrementally learn a set of clearly distinguishable tasks

Important challenge: achieve positive transfer between tasks

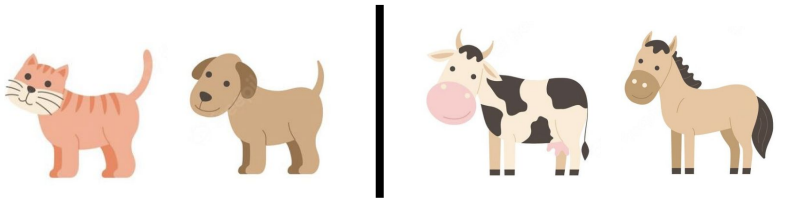


- Domain-incremental learning *(Domain-IL)*
  - Learn the same type of problem in different contexts

Important challenge: alleviate catastrophic forgetting

- Class-incremental learning *(Class-IL)*
  - Incrementally learn a growing number of classes

Important challenge: learn to discriminate between objects not observed together

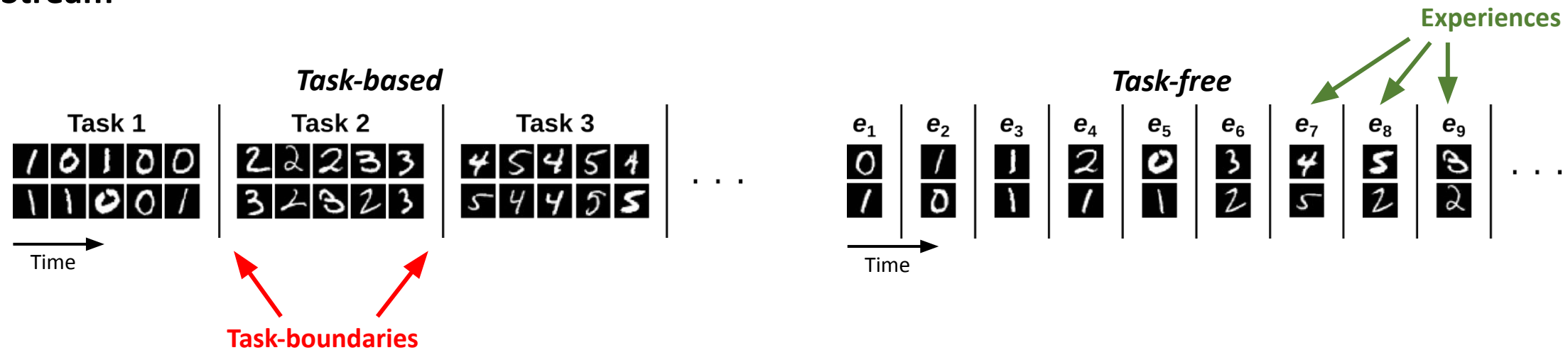


Images designed by Freepik

Sources: van de Ven & Tolias (2018, NeurIPS workshop), van de Ven et al. (2022, Nature Machine Intelligence)

Task-based *vs.* task-free continual learning

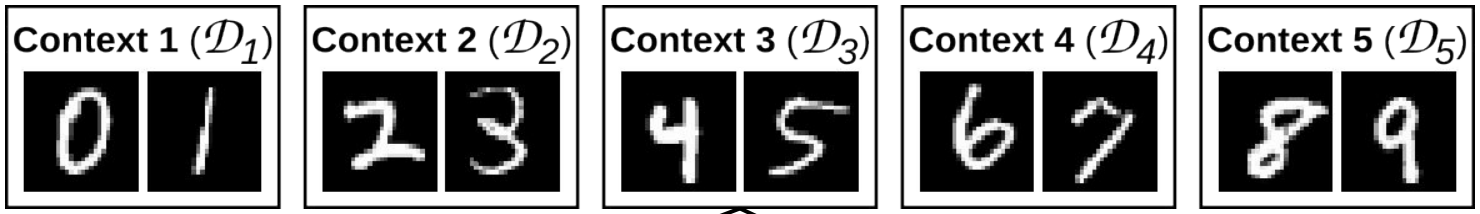
Data Stream



Task-based *vs.* task-free: formalizing non-stationarity

Context Set

Collection of underlying data-distributions



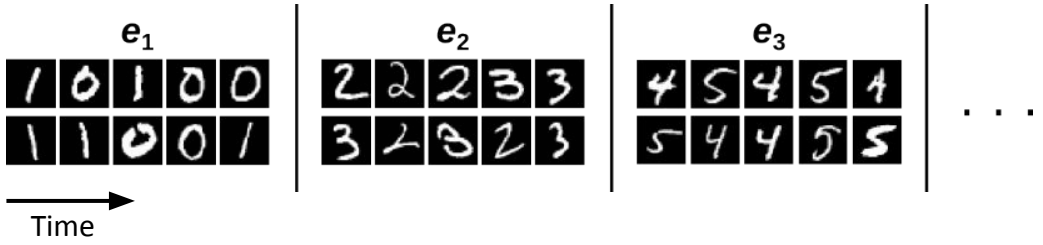
Data Stream

Sequence of 'experiences' presented to algorithm

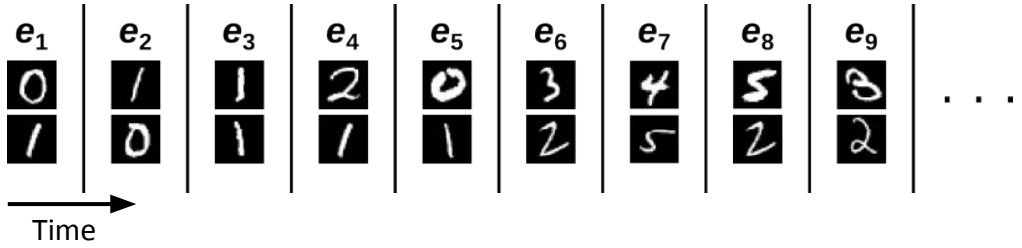
$e_t = \mathcal{D}_t^{\text{train}}$

$e_t[i] \sim \sum_{c \in C} p_c^{t,i} \mathcal{D}_c$

Task-based



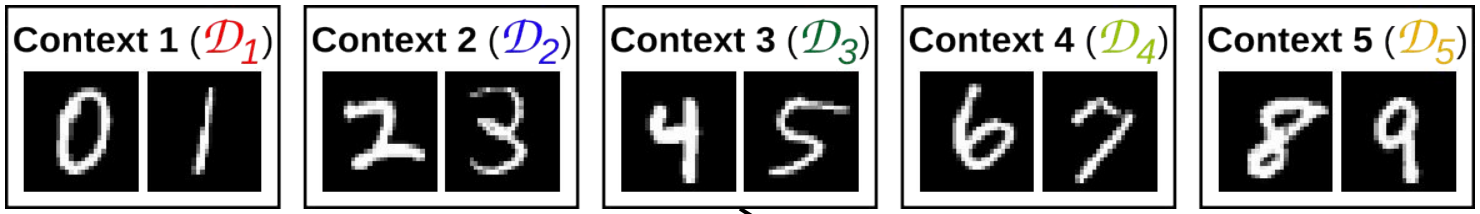
Task-free



Task-based *vs.* task-free: formalizing non-stationarity

Context Set

Collection of underlying data-distributions

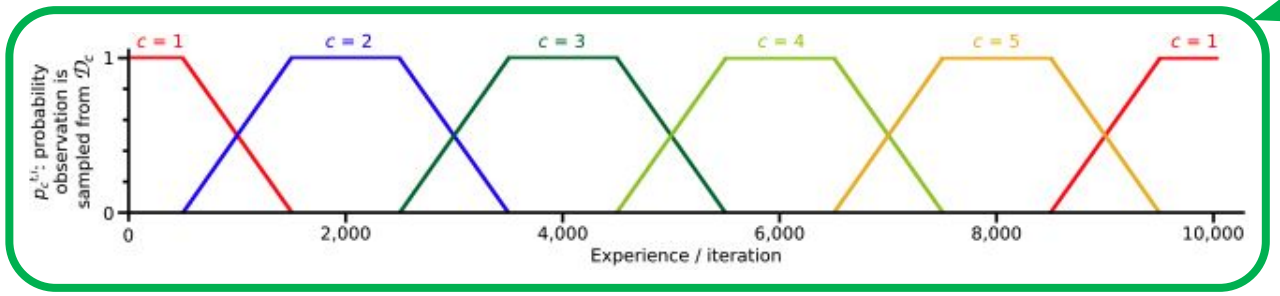


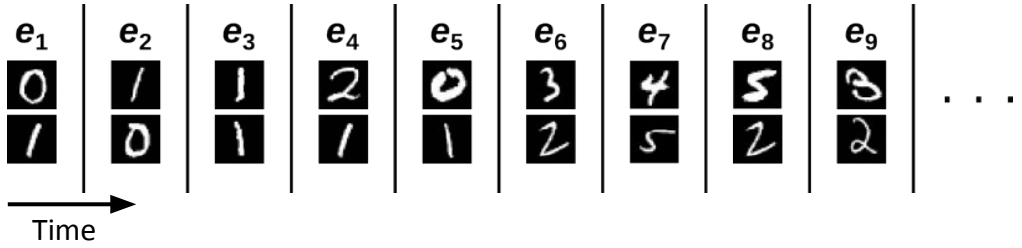
$$e_t[i] \sim \sum_{c \in \mathcal{C}} p_c^{t,i} \mathcal{D}_c$$

Data Stream

Sequence of 'experiences' presented to algorithm

Schedule



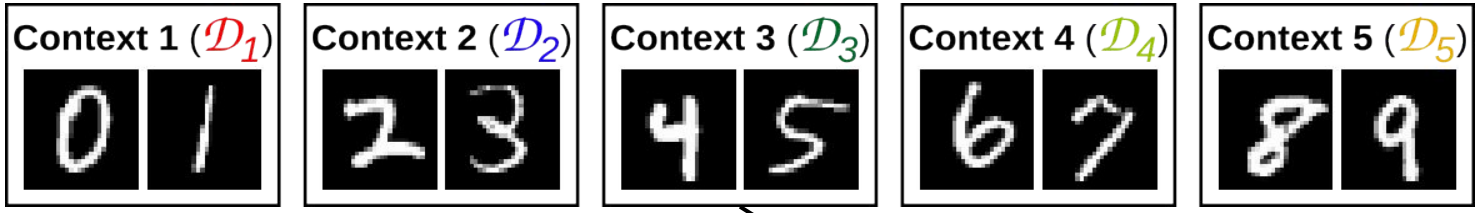


General framework

[1] Context Set

Collection of underlying data-distributions

"What aspect of the data changes over time?"



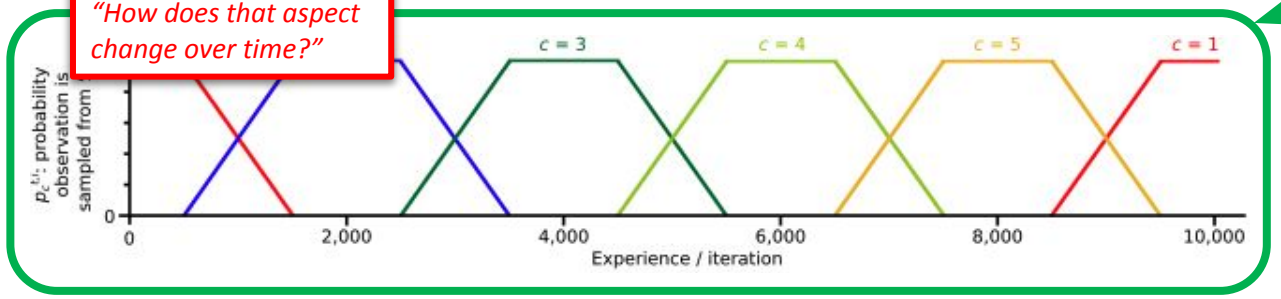
[2] Data Stream

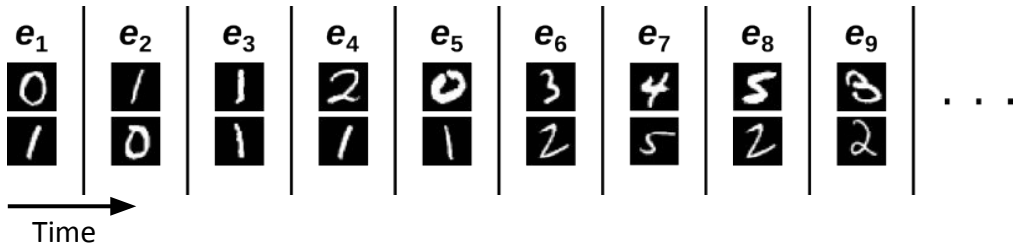
Sequence of 'experiences' presented to algorithm

"How does that aspect change over time?"

$$e_t[i] \sim \sum_{c \in \mathcal{C}} p_c^{t,i} \mathcal{D}_c$$

Schedule





[3] Scenario

What is expected of the algorithm?

"How does that aspect relate to the mapping to learn?"

	Type of choice	*Mapping to learn*
(Generalized) Task-IL	Choice between two digits of same context	$f: \mathcal{X} \times \mathcal{C} \rightarrow \mathcal{Y}$
(Generalized) Domain-IL	Is the digit odd or even?	$f: \mathcal{X} \rightarrow \mathcal{Y}$
(Generalized) Class-IL	Choice between all ten digits	$f: \mathcal{X} \rightarrow \mathcal{C} \times \mathcal{Y}$

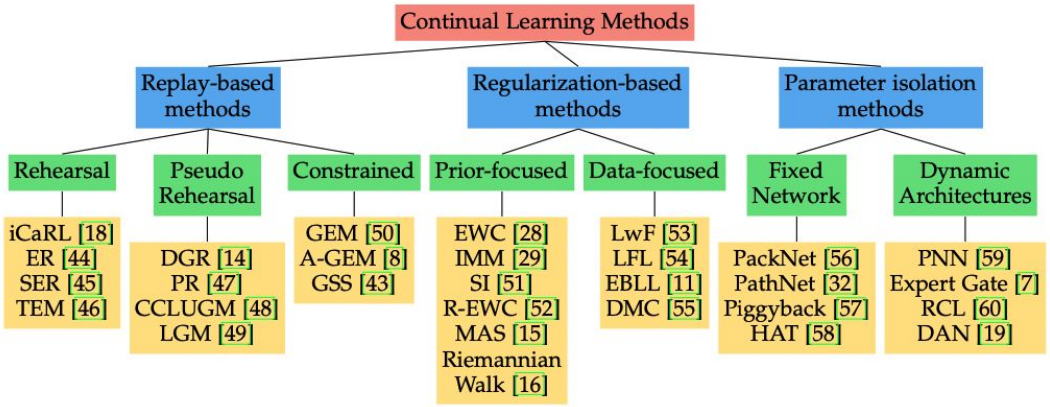
$\mathcal{X}$ = image pixel space

$\mathcal{C}$ = context space = {1,2,3,4,5}

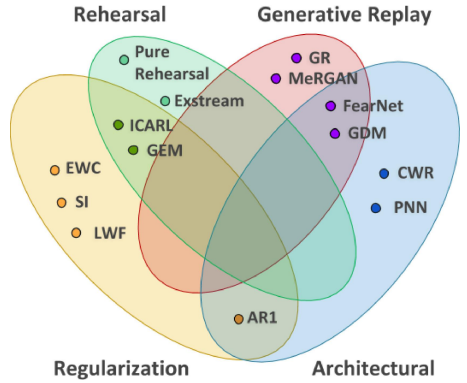
$\mathcal{Y}$ = within-context label space = {0,1}

Approaches for continual learning

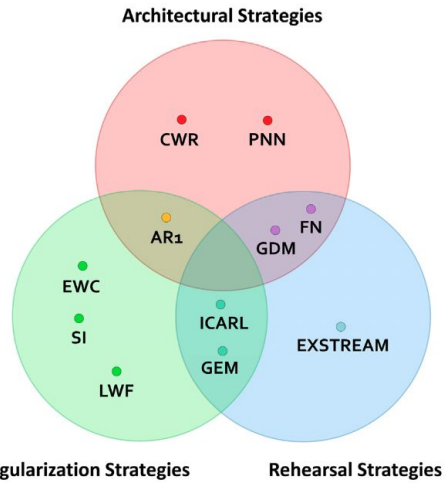
Categorizations of continual learning strategies



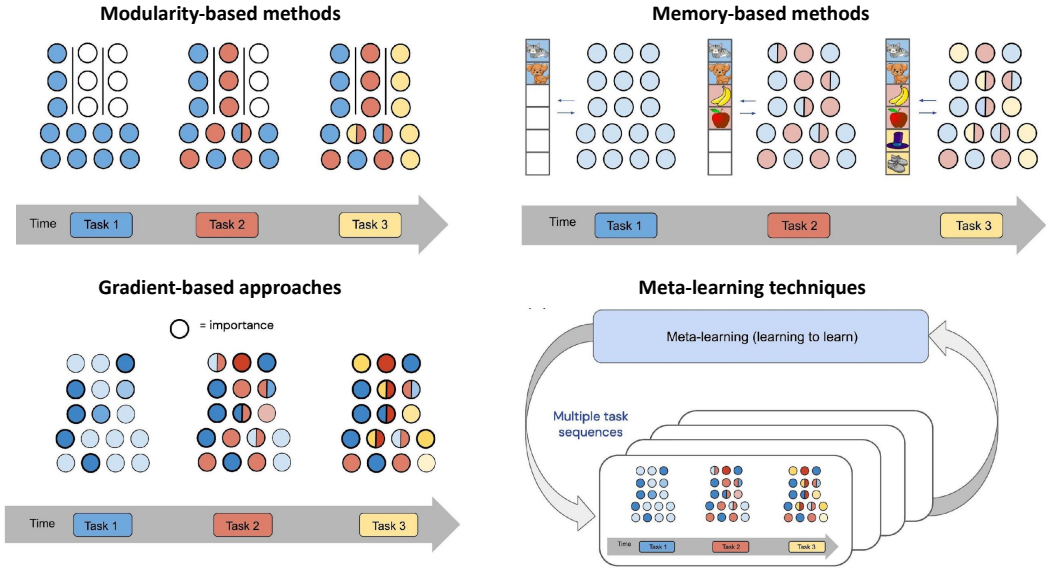
Source: De Lange et al. (2022, TPAMI)



Source: Lesort et al. (2020, Information Fusion)



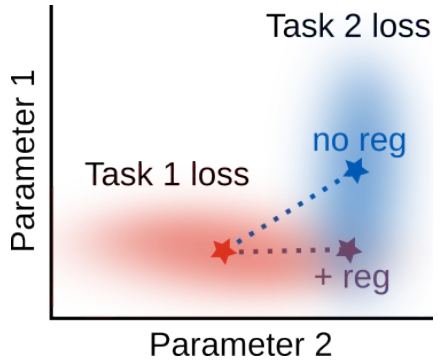
Source: Maltoni & Lomonaco (2019, Neural Networks)



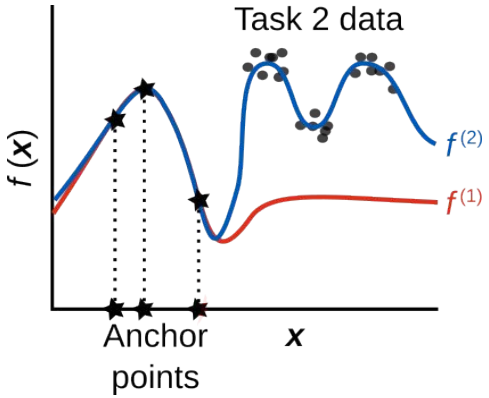
Source: Hadsell et al. (2020, Trends in Cognitive Sciences)

Categorizations of continual learning strategies

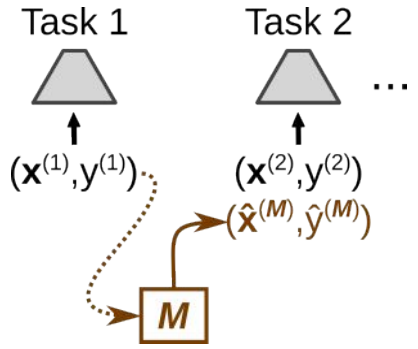
Parameter regularization



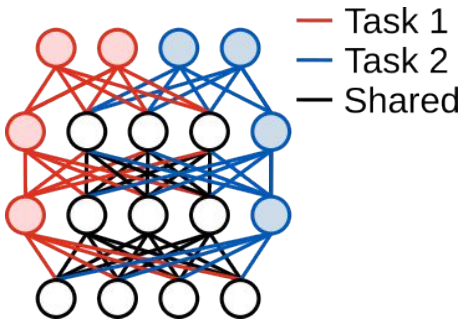
Functional regularization



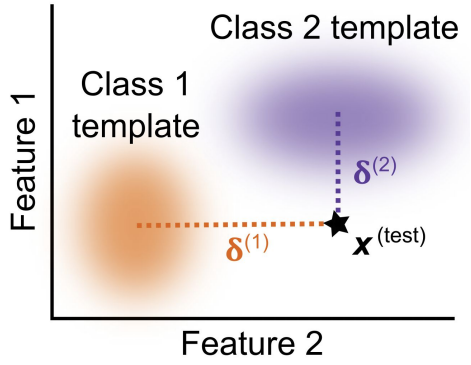
Replay



Context-specific components



Template-based classification



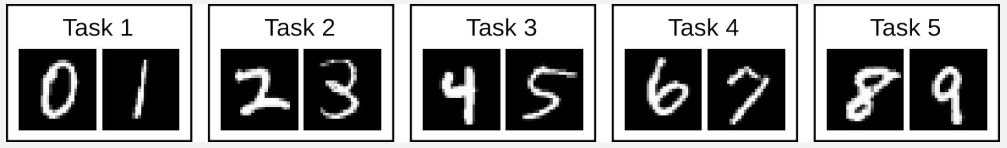
Source: van de Ven et al. (2022, Nature Machine Intelligence)

Baselines: finetuning (*lower target*) & joint training (*upper target*)

None: Network sequentially trained on each task in the standard way (*lower target*)

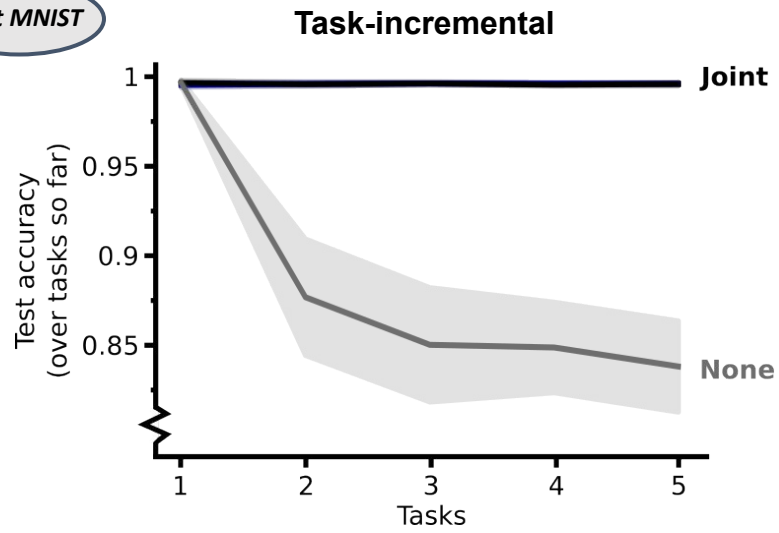
Joint: Network always trained on data of all tasks so far (*upper target*)

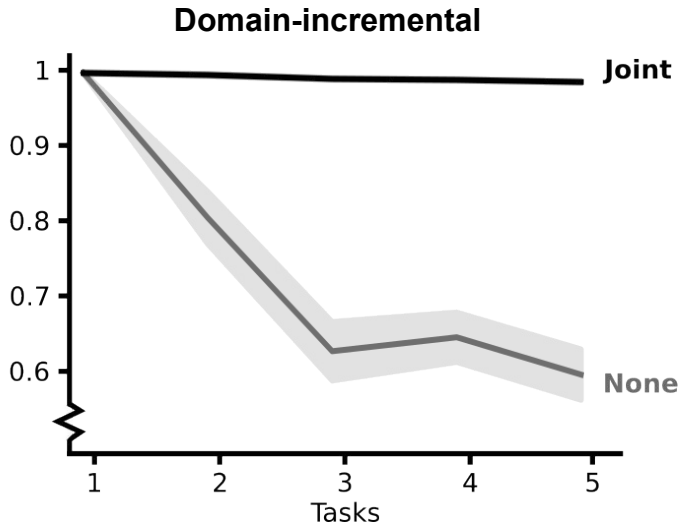
Empirical comparison on Split MNIST according to each scenario

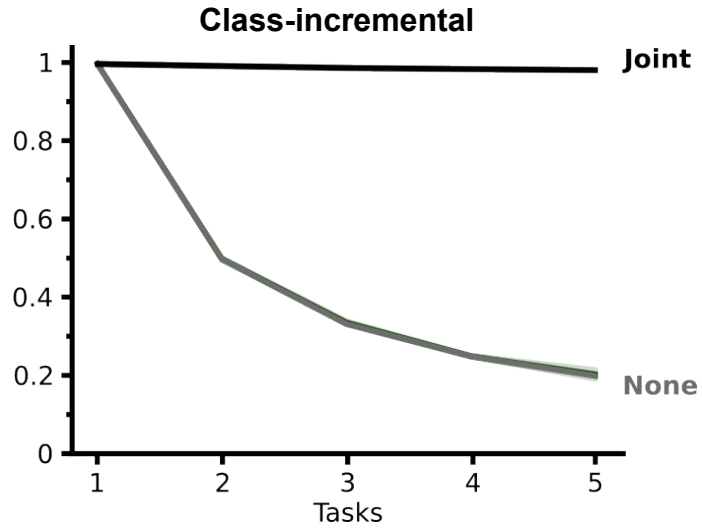


Task-incremental learning	Choice between two digits of same task (*e.g.*, 0 or 1?)
Domain-incremental learning	Is the digit odd or even?
Class-incremental learning	Choice between all ten digits

Split MNIST





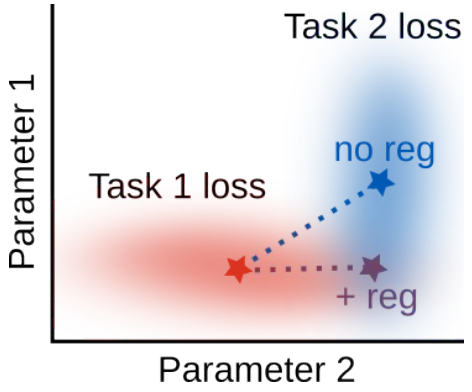


Code for these experiments: https://github.com/GMvandeVen/continual-learning

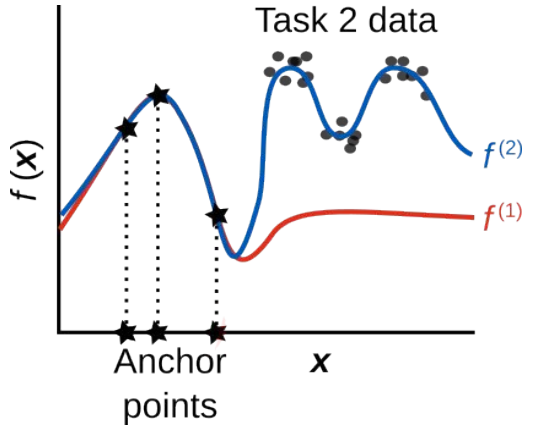
Regularization

- In continual learning, regularization typically means adding a penalty term to the loss function to **encourage the model to stay close to a previous version of itself**.
- Often, the version relative to which changes are penalized is a copy of the model stored after finishing training on the last task
- Two forms of regularization:

Parameter regularization

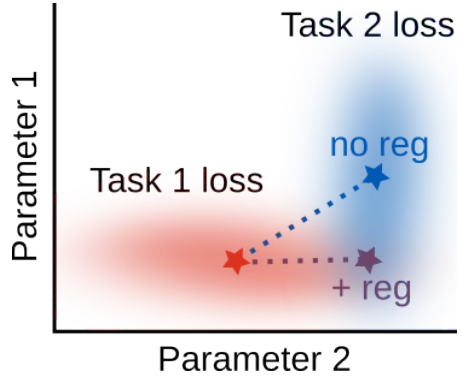


Functional regularization



Parameter regularization

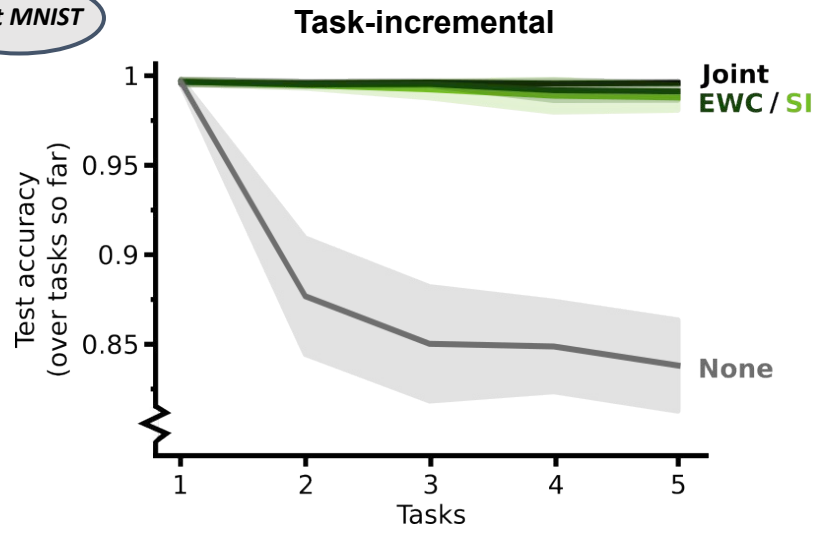
- Parameters important for past tasks are encouraged not to change too much when learning a new task
- Can often be interpreted as sequential approximate Bayesian inference on the network's parameters
- Representative methods:
  - Elastic Weight Consolidation [**EWC**] (Kirkpatrick et al., 2017 PNAS)
  - Synaptic Intelligence [**SI**] (Zenke et al., 2017 ICML)

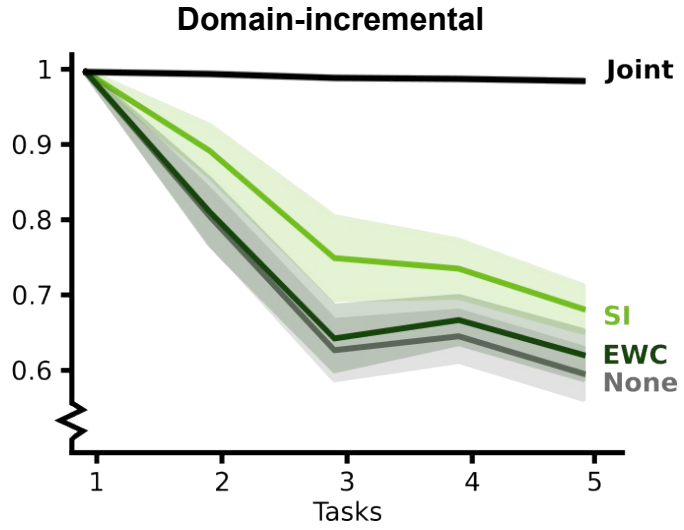


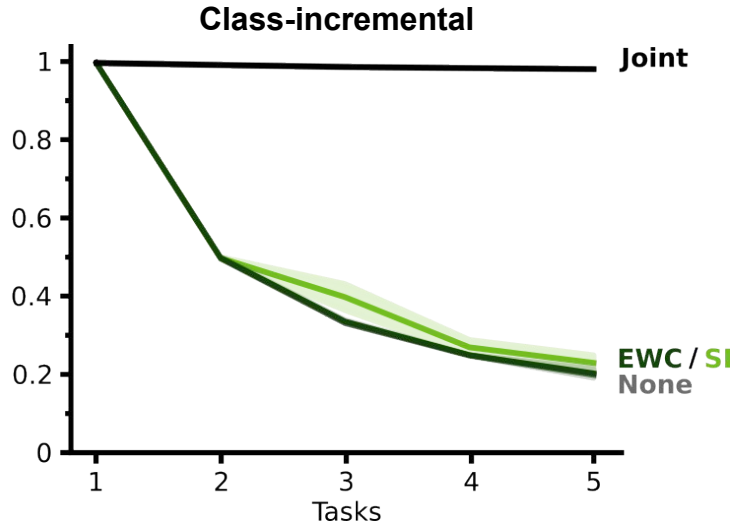
$$\mathcal{L}_{\text{total}} = \mathcal{L} + \|\theta - \theta^*\|_\Sigma$$

θ^*: parameters relative to which changes are penalized
Σ: estimate of how important parameters are
$\|.\|_\Sigma$: weighted norm

Split MNIST



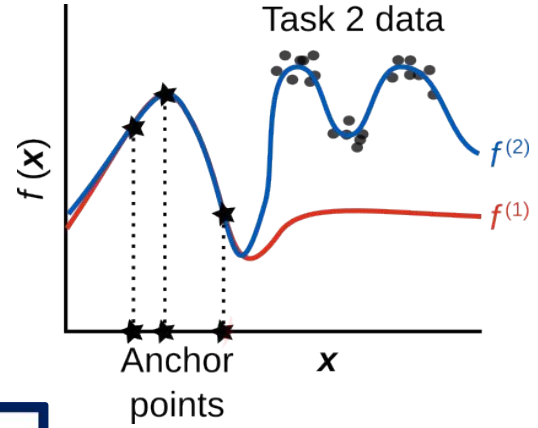




Code for these experiments: https://github.com/GMvandeVen/continual-learning

Functional regularization

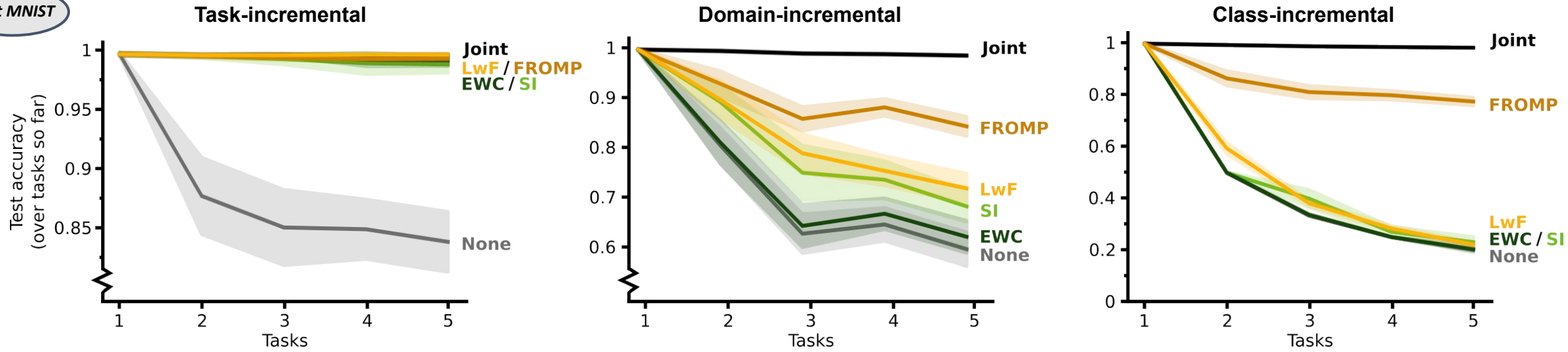
- The input-output mapping learned previously is encouraged not to change too much at a particular set of inputs (the 'anchor points')
- Also referred to as knowledge distillation
- Representative methods:
  - Learning without Forgetting [**LwF**] (Li & Hoiem, 2017 TPAMI)
  - Functional Regularization Of Memorable Past [**FROMP**] (Pan et al., 2020 NeurIPS)



$$\mathcal{L}_{\text{total}} = \mathcal{L} + \langle f_\theta, f_{\theta^*} \rangle_{\mathcal{A}}$$

f_{θ^*}: function relative to which changes are penalized
$\mathcal{A}$: set of 'anchor points' at which the divergence between f_θ and f_{θ^*} is measured

Split MNIST

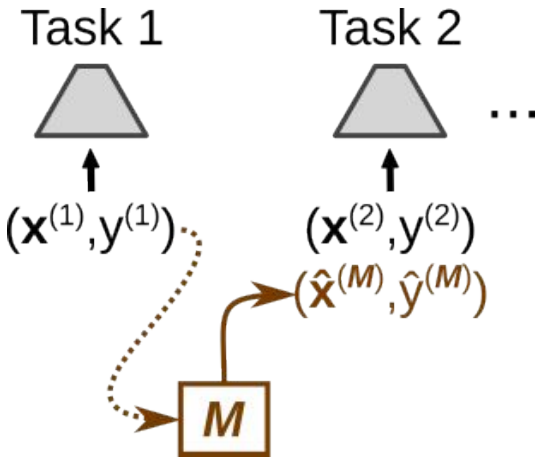


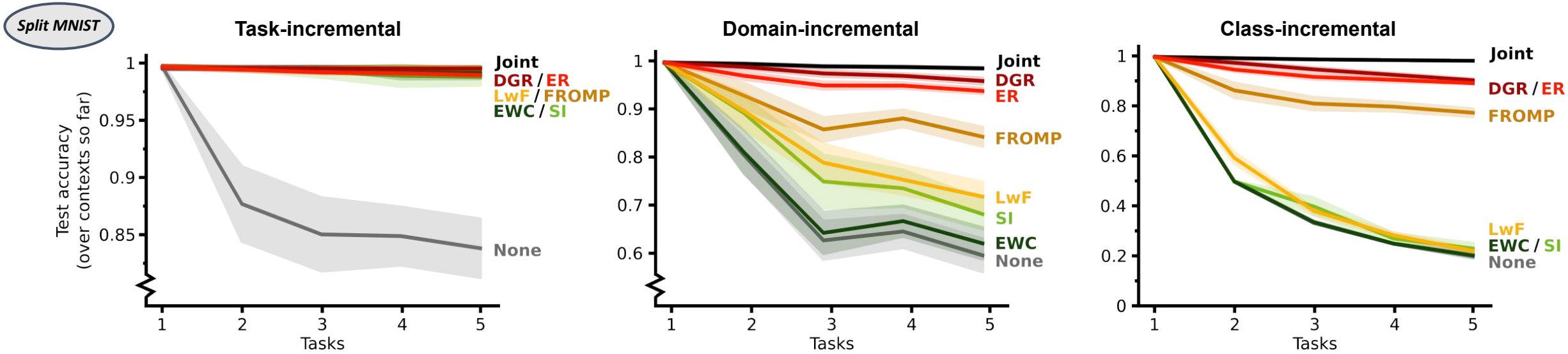
Memory buffer size (**FROMP**): 100 examples per class

Code for these experiments: https://github.com/GMvandeVen/continual-learning

Replay

- Current training data is complemented with data representative of past observations
- The replayed data can be sampled from a memory buffer or a generative model
- Representative methods:
  - Experience Replay [**ER**] (Chaudhry et al., 2019 arXiv)
  - Deep Generative Replay [**DGR**] (Shin et al., 2017 NeurIPS)



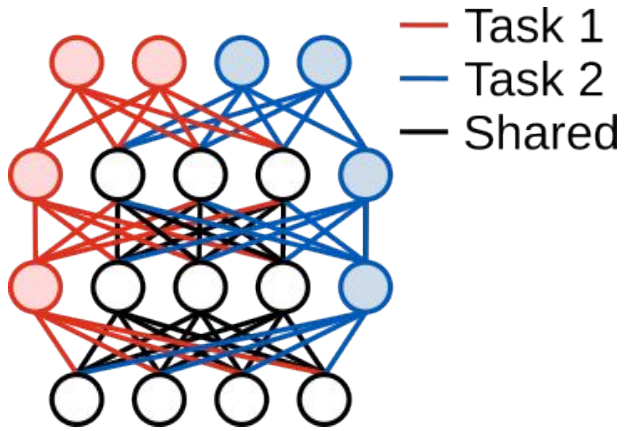


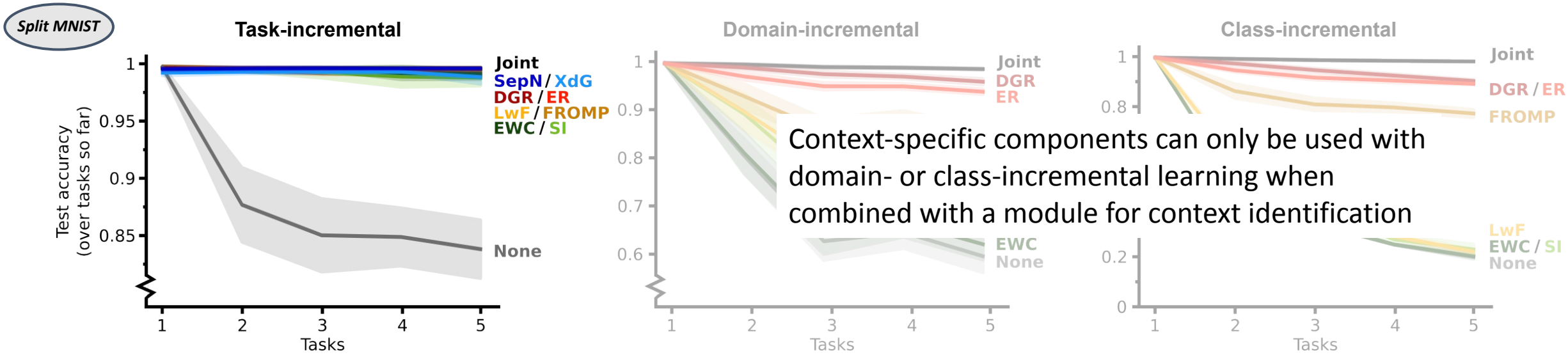
Memory buffer size (**FROMP**, **ER**): 100 examples per class

Code for these experiments: https://github.com/GMvandeVen/continual-learning

Context-specific components

- Parts of the network are only used for specific tasks
- Commonly used example: multi-headed output layer
- Requires knowledge of task identity at test time
- Representative methods:
  - Context-dependent Gating [**XdG**] (Masse et al., 2018 PNAS)
  - Separate Networks [**SepN**]



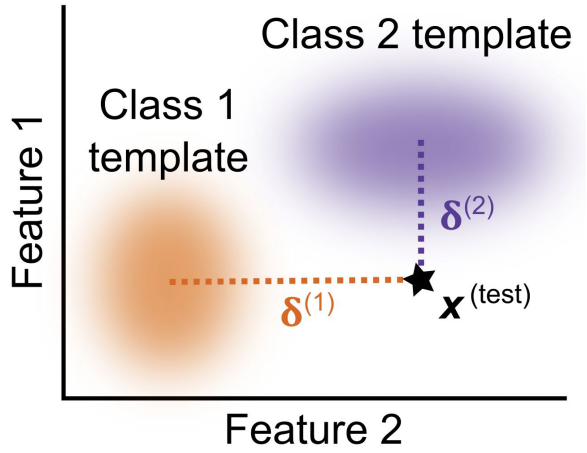


Memory buffer size (**FROMP**, **ER**): 100 examples per class

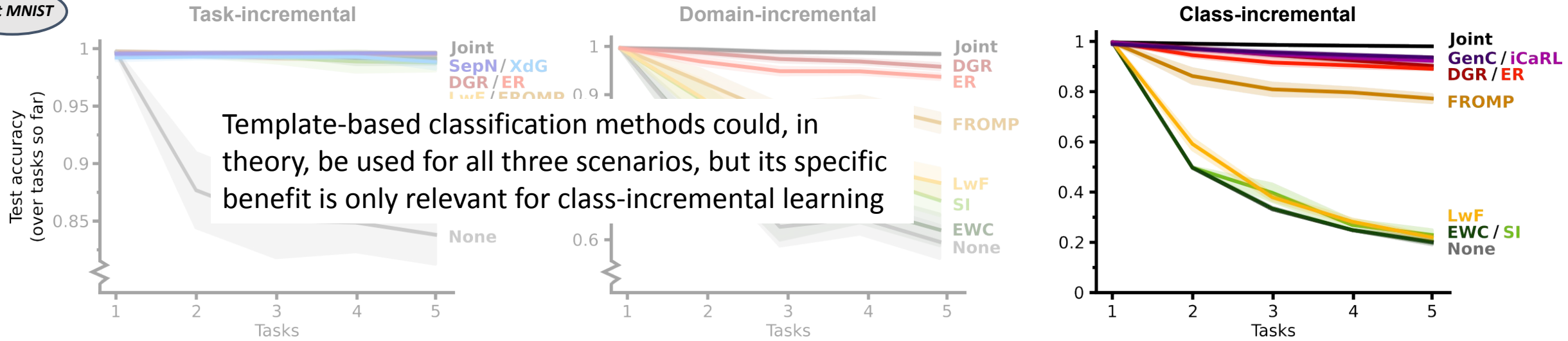
Code for these experiments: https://github.com/GMvandeVen/continual-learning

Template-based classification

- A ‘template’ is learned for each class, and classification is performed based on which template is most suitable for sample to be classified
- Examples of templates are prototypes or generative models
- Allows comparing classes ‘at test time’, rather than during training
- Representative methods:
  - Incremental Classifier and Representation Learning [**iCaRL**] (Rebuffi et al., 2017 CVPR)
  - Generative Classifier [**GenC**] (van de Ven et al., 2021 CVPR-W)



Split MNIST



Template-based classification methods could, in theory, be used for all three scenarios, but its specific benefit is only relevant for class-incremental learning

Memory buffer size (**FROMP**, **ER**, **iCaRL**): 100 examples per class

Code for these experiments: https://github.com/GMvandeVen/continual-learning

Overview: Split CIFAR-100

Strategy	Method	Budget	GM	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower target*			*61.43 (± 0.36)*	*18.42 (± 0.33)*	*7.71 (± 0.18)*
	Joint – upper target			*78.78 (± 0.25)*	*46.85 (± 0.51)*	*49.78 (± 0.21)*
Context-specific components	Separate Networks	-	-	76.83 (± 0.25)	-	-
	XdG	-	-	69.86 (± 0.34)	-	-
Parameter regularization	EWC	-	-	76.34 (± 0.29)	21.65 (± 0.55)	8.24 (± 0.25)
	SI	-	-	74.84 (± 0.39)	22.58 (± 0.42)	8.10 (± 0.24)
Functional regularization	LwF	-	-	78.59 (± 0.24)	29.45 (± 0.39)	25.57 (± 0.27)
	FROMP	100	-	not run	not run	not run
Replay	DGR	-	yes	71.40 (± 0.32)	20.52 (± 0.43)	9.67 (± 0.22)
	ER	100	-	76.43 (± 0.24)	39.00 (± 0.34)	37.57 (± 0.21)
Template-based classification	Generative Classifier	-	yes	-	-	46.83 (± 0.18)
	iCaRL	100	-	-	-	37.83 (± 0.21)

Shown is final test accuracy (as %, averaged over all tasks) on Split CIFAR-100. 'Budget' indicates number of samples per class stored in memory, 'GM' indicates generative model was learned using extra parameters. Experiments were run 10 times, reported is the mean (± SEM). Source: van de Ven et al. (2022, Nature Machine Intelligence)

Overview: Split CIFAR-100

Strategy	Method	Budget	GM	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower target*			*61.43 (± 0.36)*	*18.42 (± 0.33)*	*7.71 (± 0.18)*
	Joint – upper target			*78.78 (± 0.25)*	*46.85 (± 0.51)*	*49.78 (± 0.21)*
Context-specific components	Separate Networks	-	-	76.83 (± 0.25)	-	-
	XdG	-	-	69.86 (± 0.34)	-	-
Parameter regularization	EWC	-	-	76.34 (± 0.29)	21.65 (± 0.55)	8.24 (± 0.25)
	SI	-	-	74.84 (± 0.39)	22.58 (± 0.42)	8.10 (± 0.24)
Functional regularization	LwF	-	-	78.59 (± 0.24)	29.45 (± 0.39)	25.57 (± 0.27)
	FROMP	100	-	not run	not run	not run
Replay	DGR	-	yes	71.40 (± 0.32)	20.52 (± 0.43)	9.67 (± 0.22)
	ER	100	-	76.43 (± 0.24)	39.00 (± 0.34)	37.57 (± 0.21)
Template-based classification	Generative Classifier	-	yes	-	-	46.83 (± 0.18)
	iCaRL	100	-	-	-	37.83 (± 0.21)

Shown is final test accuracy (as %, averaged over all tasks) on Split CIFAR-100. 'Budget' indicates number of samples per class stored in memory, 'GM' indicates generative model was learned using extra parameters. Experiments were run 10 times, reported is the mean (± SEM). Source: van de Ven et al. (2022, Nature Machine Intelligence)

Overview: Split CIFAR-100

Strategy	Method	Budget	GM	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower target*			*61.43 (± 0.36)*	*18.42 (± 0.33)*	*7.71 (± 0.18)*
	Joint – upper target			*78.78 (± 0.25)*	*46.85 (± 0.51)*	*49.78 (± 0.21)*
Context-specific components	Separate Networks	-	-	76.83 (± 0.25)	-	-
	XdG	-	-	69.86 (± 0.34)	-	-
Parameter regularization	EWC	-	-	76.34 (± 0.29)	21.65 (± 0.55)	8.24 (± 0.25)
	SI	-	-	74.84 (± 0.39)	22.58 (± 0.42)	8.10 (± 0.24)
Functional regularization	LwF	-	-	78.59 (± 0.24)	29.45 (± 0.39)	25.57 (± 0.27)
	FROMP	100	-	not run	not run	not run
Replay	DGR	-	yes	71.40 (± 0.32)	20.52 (± 0.43)	9.67 (± 0.22)
	ER	100		76.43 (± 0.24)	39.00 (± 0.34)	37.57 (± 0.21)
Template-based classification	Generative Classifier	-	yes	-	-	46.83 (± 0.18)
	iCaRL	100	-	-	-	37.83 (± 0.21)

Shown is final test accuracy (as %, averaged over all tasks) on Split CIFAR-100. 'Budget' indicates number of samples per class stored in memory, 'GM' indicates generative model was learned using extra parameters. Experiments were run 10 times, reported is the mean (± SEM). Source: van de Ven et al. (2022, Nature Machine Intelligence)

Overview: Split CIFAR-100

Strategy	Method	Budget	GM	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower target*			*61.43 (± 0.36)*	*18.42 (± 0.33)*	*7.71 (± 0.18)*
	Joint – upper target			*78.78 (± 0.25)*	*46.85 (± 0.51)*	*49.78 (± 0.21)*
Context-specific components	Separate Networks	-	-	76.83 (± 0.25)	-	-
	XdG	-	-	69.86 (± 0.34)	-	-
Parameter regularization	EWC	-	-	76.34 (± 0.29)	21.65 (± 0.55)	8.24 (± 0.25)
	SI	-	-	74.84 (± 0.39)	22.58 (± 0.42)	8.10 (± 0.24)
Functional regularization	LwF	-	-	78.59 (± 0.24)	29.45 (± 0.39)	25.57 (± 0.27)
	FROMP	100	-	not run	not run	not run
Replay	DGR	-	yes	71.40 (± 0.32)	20.52 (± 0.43)	9.67 (± 0.22)
	ER	100	-	76.43 (± 0.24)	39.00 (± 0.34)	37.57 (± 0.21)
Template-based classification	Generative Classifier	-	yes	-	-	46.83 (± 0.18)
	iCaRL	100	-	-	-	37.83 (± 0.21)

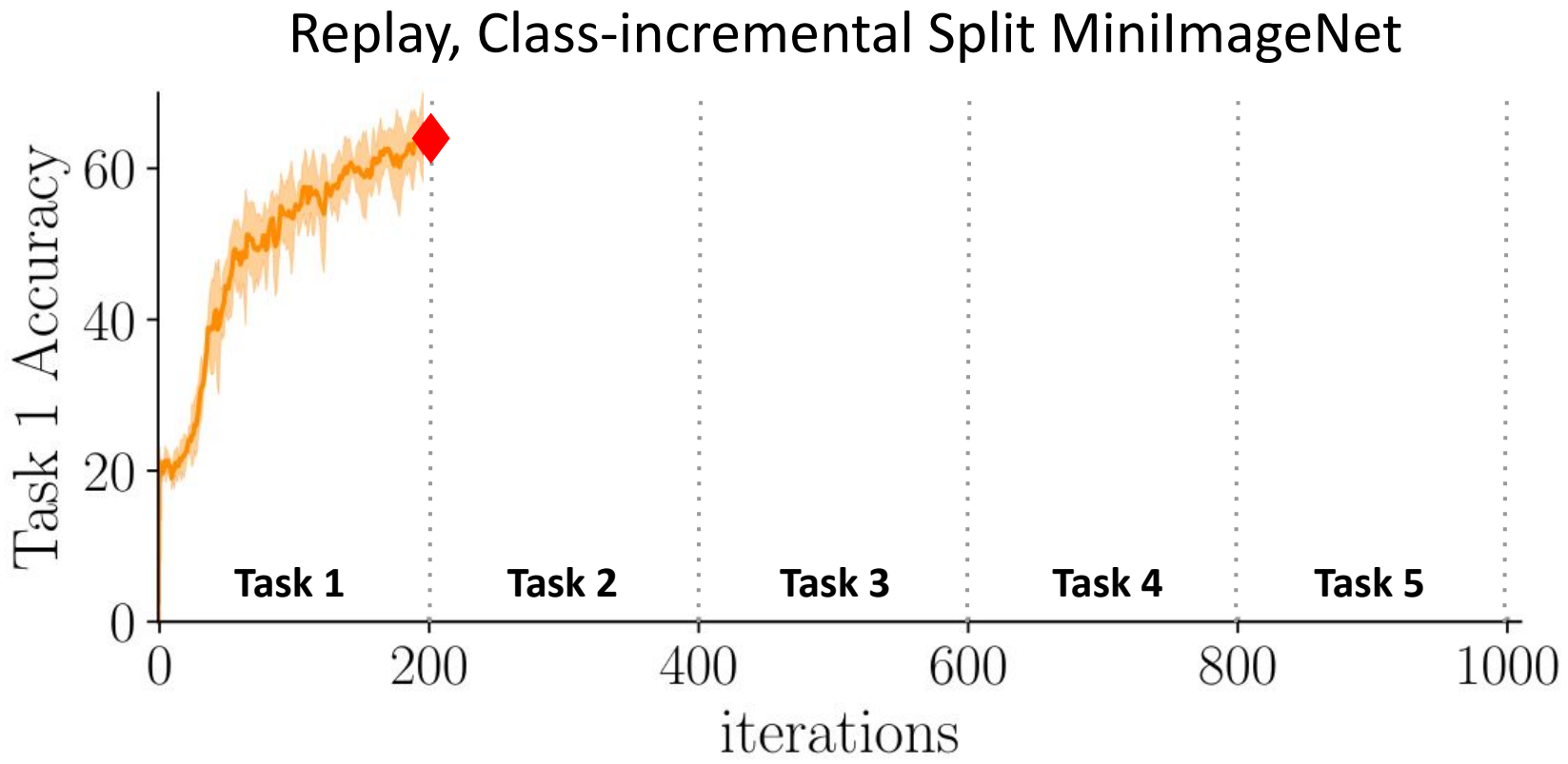
Shown is final test accuracy (as %, averaged over all tasks) on Split CIFAR-100. 'Budget' indicates number of samples per class stored in memory, 'GM' indicates generative model was learned using extra parameters. Experiments were run 10 times, reported is the mean (± SEM). Source: van de Ven et al. (2022, Nature Machine Intelligence)

Summary

- Continual learning, the skill of incrementally learning from a non-stationary stream of data, is a *key aspect of intelligence*.
- Deep neural networks suffer from catastrophic forgetting.
- Successful continual learning could lead to substantial reductions in required resources, allow for efficiently correcting mistakes/biases in trained models and on-device personalization.
- Continual learning is not a unitary problem: two often made distinctions are between task-, domain- and class-incremental learning, and between task-based and task-free continual learning.
- Five main computational approaches for continual learning are: (i) parameter regularization, (ii) functional regularization, (iii) replay, (iv) context-dependent processing and (v) template-based classification.

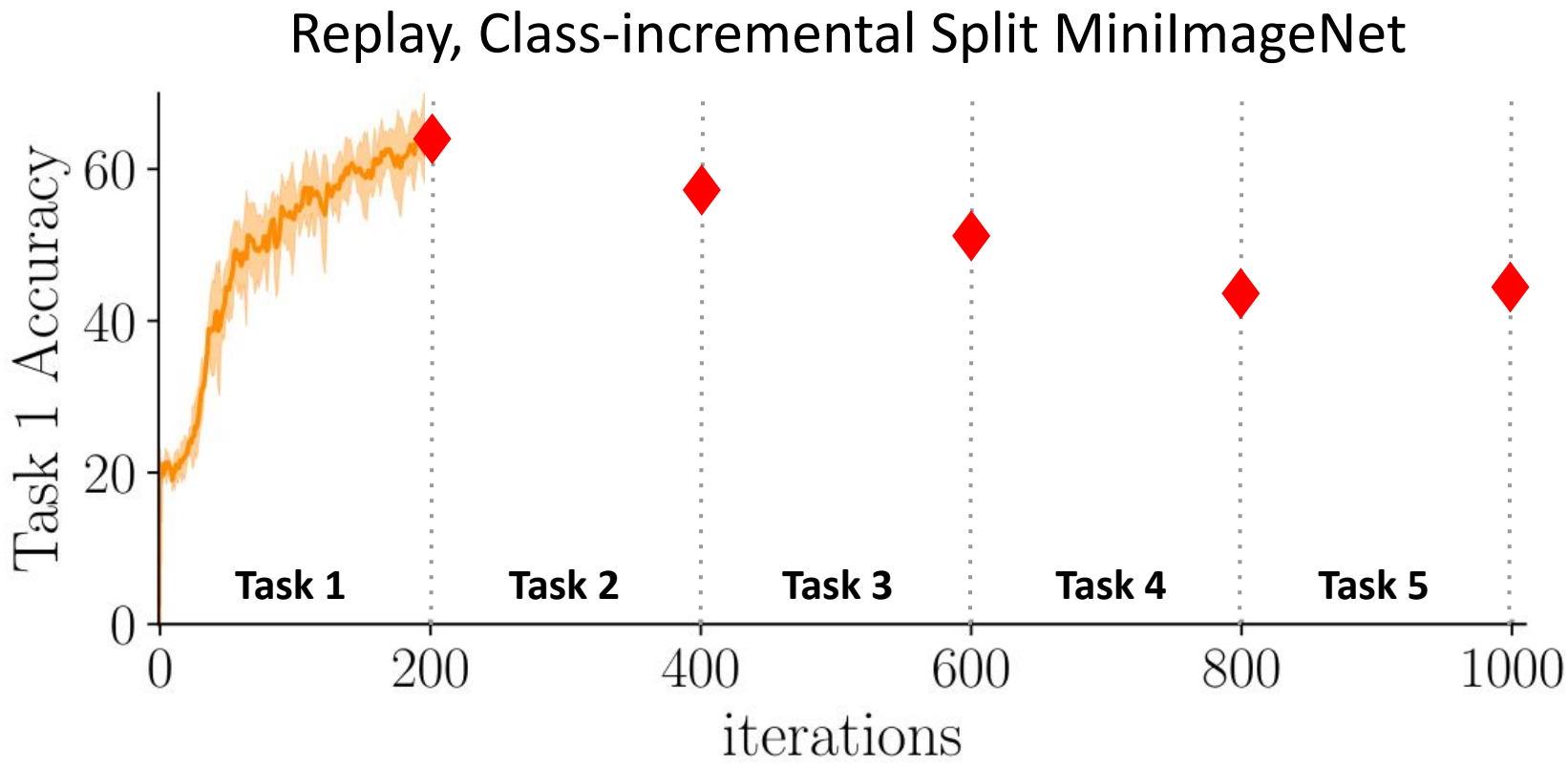
The road forward:
Open topics in continual learning

Does replay prevent forgetting?



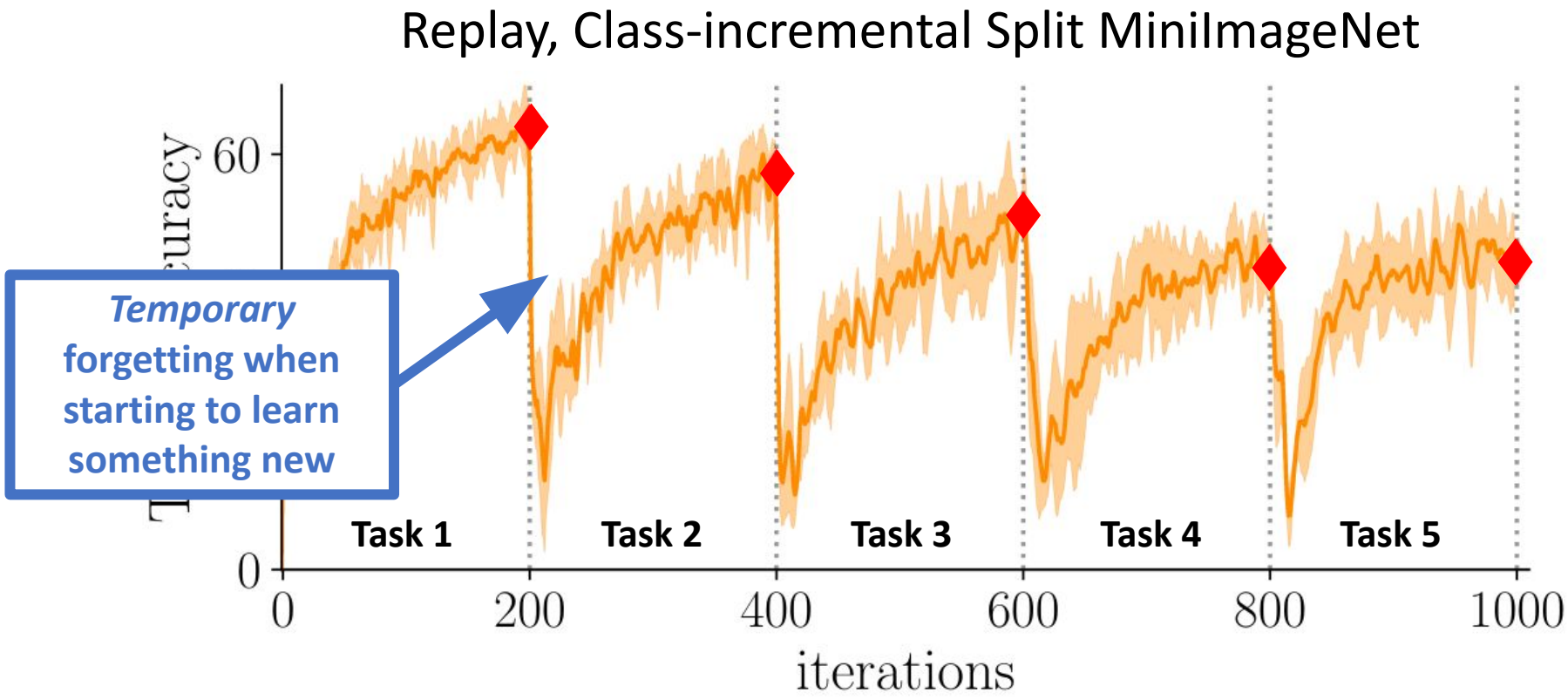
De Lange, **van de Ven** & Tuytelaars (2023) "Continual evaluation for lifelong learning: Identifying the stability gap" *ICLR, spotlight*

Does replay prevent forgetting?



De Lange, **van de Ven** & Tuytelaars (2023) "Continual evaluation for lifelong learning: Identifying the stability gap" *ICLR, spotlight*

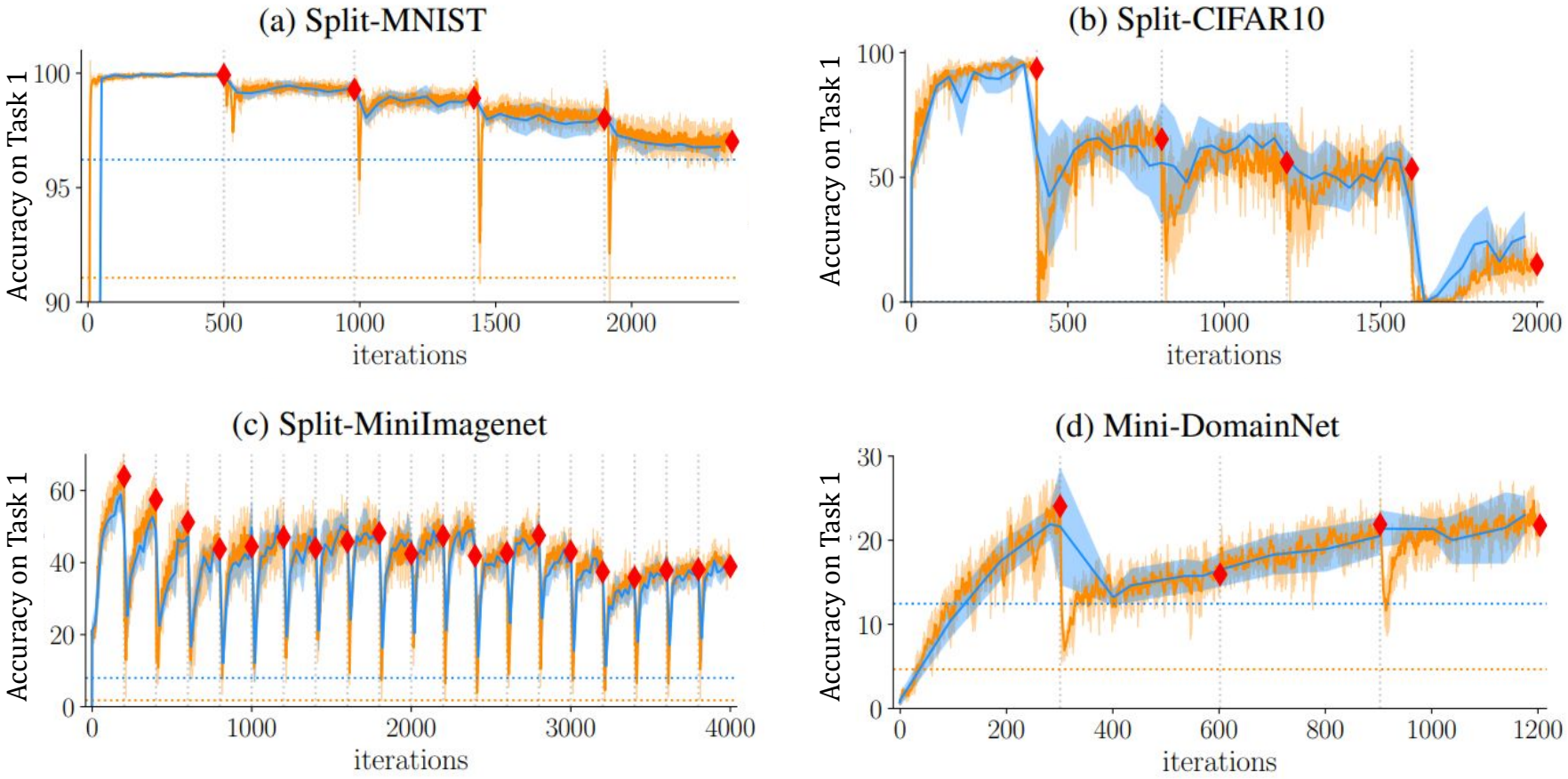
Does replay prevent forgetting?



De Lange, **van de Ven** & Tuytelaars (2023) "Continual evaluation for lifelong learning: Identifying the stability gap" *ICLR, spotlight*

The stability gap is consistently observed

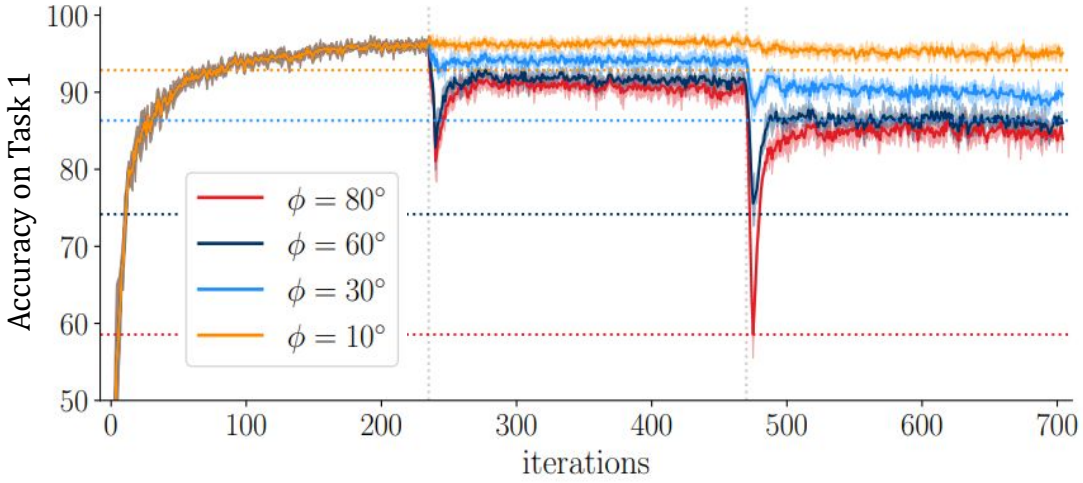
Replay, Class-incremental on …



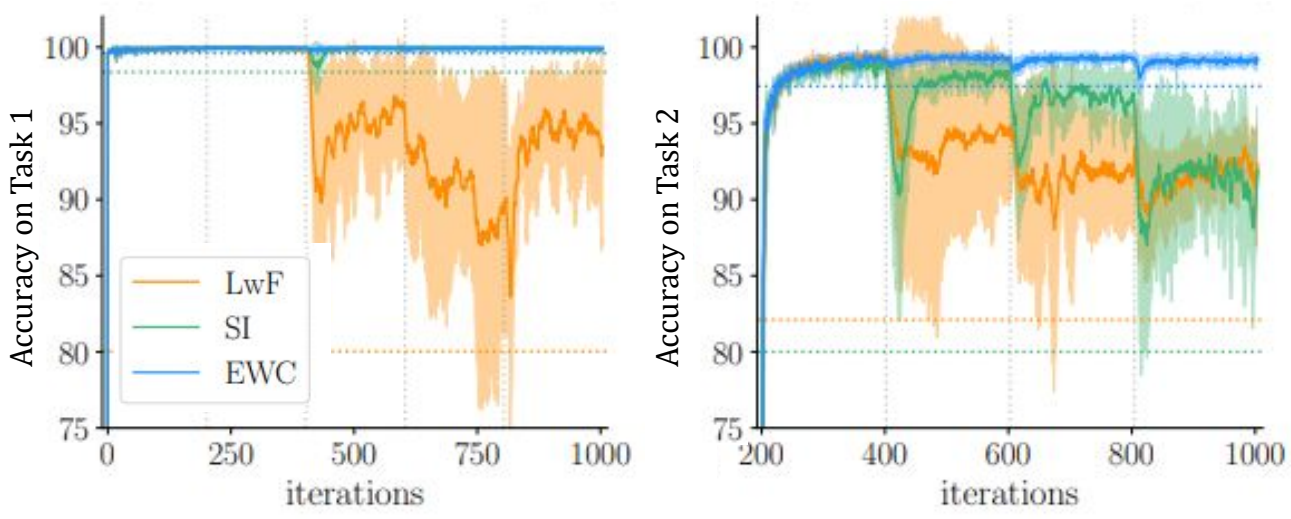
De Lange, **van de Ven** & Tuytelaars (2023) "Continual evaluation for lifelong learning: Identifying the stability gap" *ICLR, spotlight*

… also in other settings or with other methods

Replay, Domain-incremental Rotated MNIST



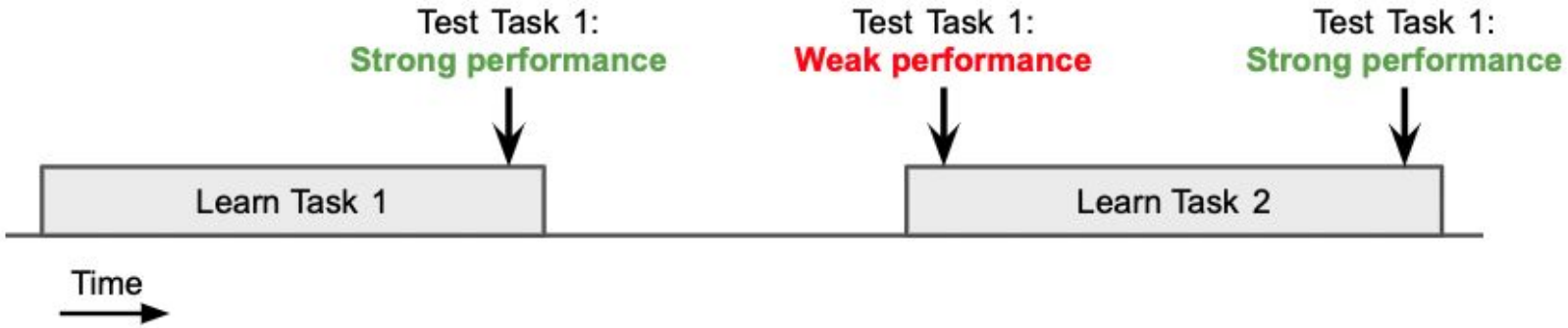
Regularization, Task-incremental Split MNIST



De Lange, **van de Ven** & Tuytelaars (2023) "Continual evaluation for lifelong learning: Identifying the stability gap" *ICLR, spotlight*

Why should we care?

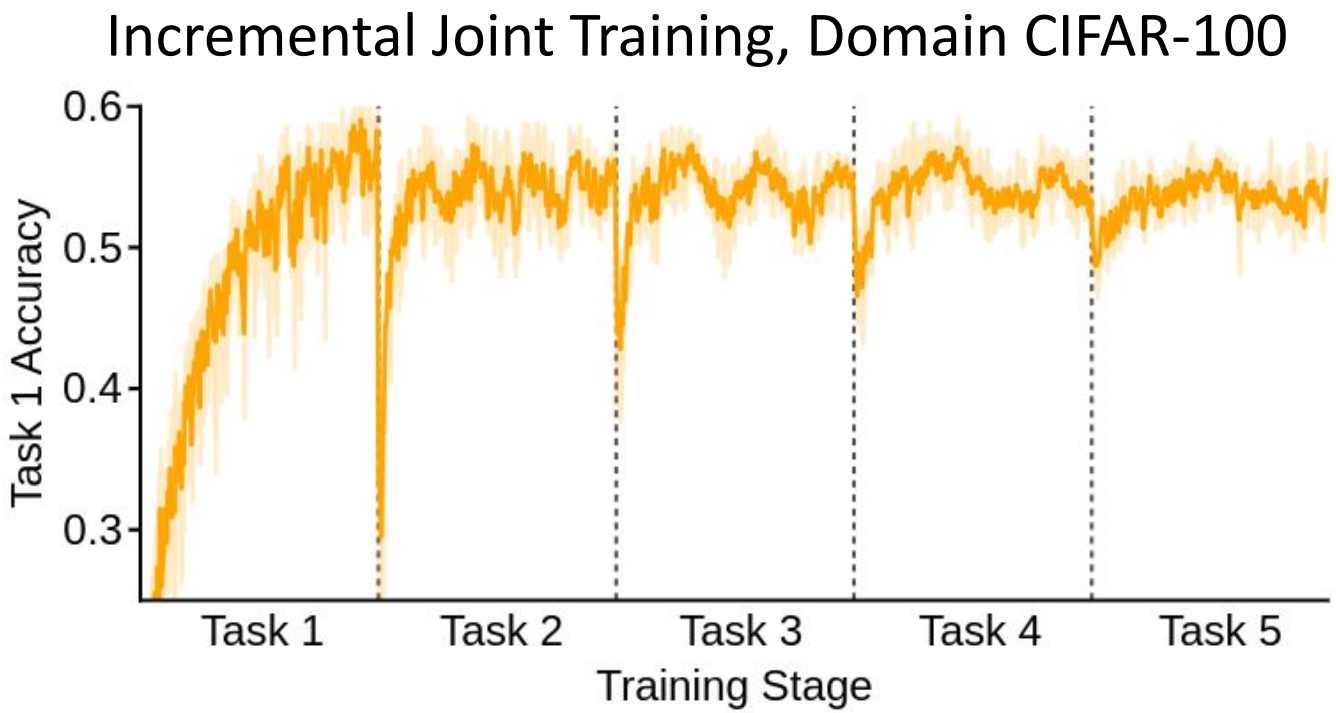
- Problematic for safety-critical applications
  - Worst-case performance might be important
  - Could be exploitable by adversarial agent with control over the training stream
- Could avoiding the stability gap lead to better *final performance*?
  - Preventing forgetting seems more efficient than having to re-learn
- Scientifically interesting
  - Do humans suffer from transient forgetting upon learning something new?



How to avoid the stability gap?

- Continue to improve the quality of replay?

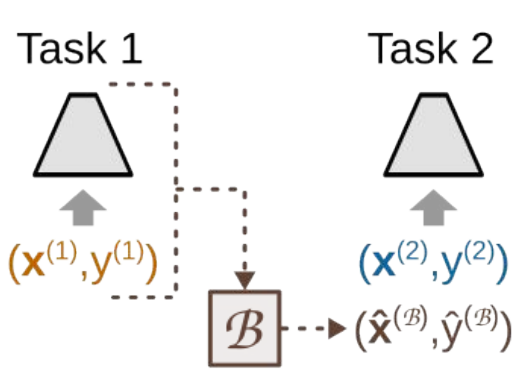
The stability gap occurs even with “perfect” replay!



Hess, Tuytelaars & **van de Ven** (2023) “Two Complementary Perspectives to Continual Learning” *Pre-registered Report, 1st ContinualAI Unconference*

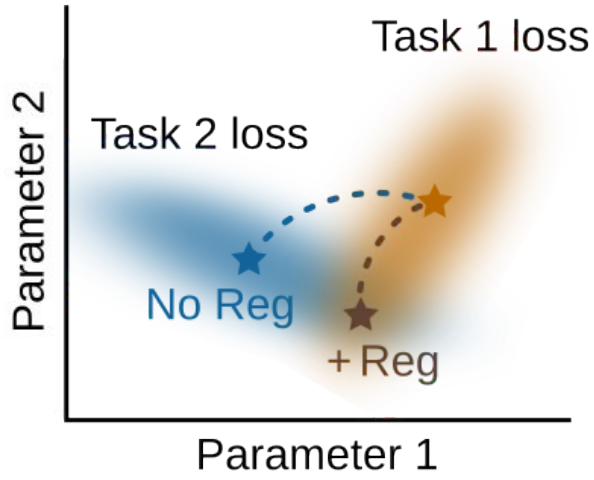
Current approach to continual learning: make changes to the loss function (i.e., focus on *where* to go)

Replay



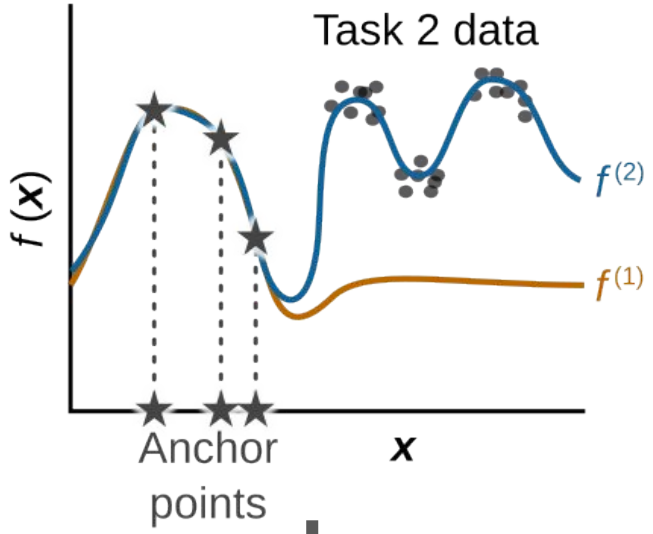
$$\ell_{\text{total}} = \ell_{\text{current}} + \ell_{\text{replay}}$$

Parameter regularization



$$\ell_{\text{total}} = \ell_{\text{current}} + \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\Sigma}$$

Functional regularization



$$\ell_{\text{total}} = \ell_{\text{current}} + \langle f_{\boldsymbol{\theta}}, f_{\boldsymbol{\theta}^*} \rangle_{\mathcal{A}}$$

Even with a perfect approximation to the joint loss, the stability gap persists!

Continual learning needs a new direction

- To overcome the stability gap, changes must be made to *how* the loss function is optimized

- Existing optimization-based methods for continual learning treat modifying the optimization routine as alternative to modifying the loss
  - *"optimize the loss on the new task under the constraint that the loss on old tasks stays low"*

- My proposal: continual learning must do both
  - *"optimize an approximation to the joint loss under the constraint that the loss on old tasks stays low"*

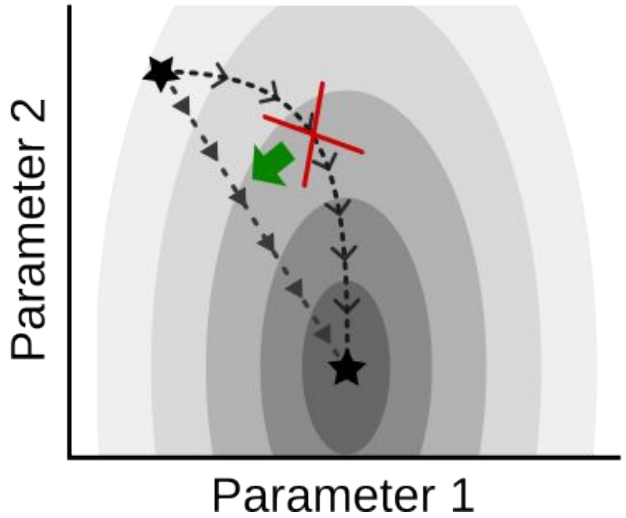
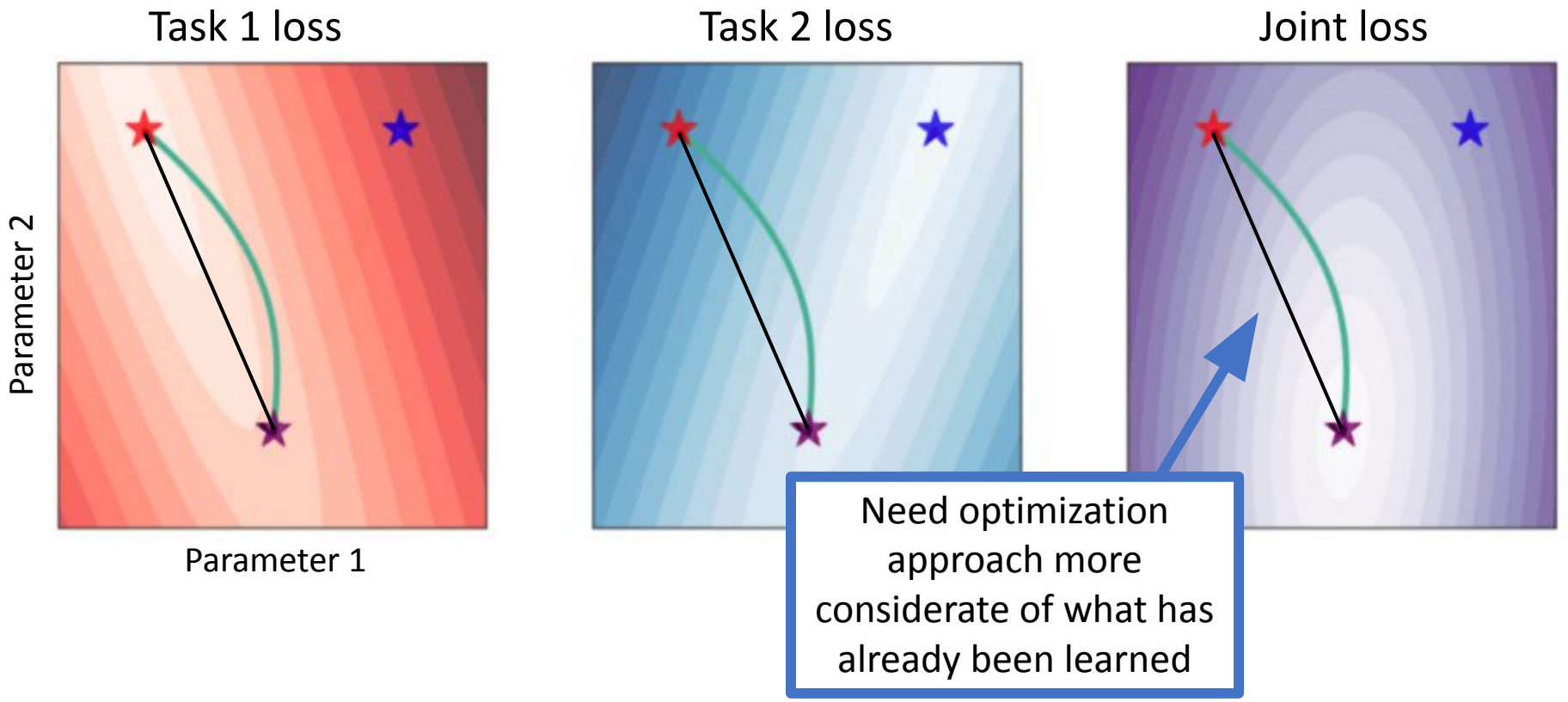


Illustration with a toy example



Kao, Jensen, **van de Ven**, Bernacchia & Hennequin (2021) "Natural continual learning: success is a journey, not (just) a destination" *NeurIPS*

Funding acknowledgements

The work and research towards preparing this presentation has been supported by an IBRO-ISN Research Fellowship, by the *Lifelong Learning Machines* (L2M) program of the Defence Advanced Research Projects Agency (DARPA) via contract number HR0011-18-2-0025 and by funding from the European Union under the Horizon 2020 research and innovation program (ERC project *KeepOnLearning*, grant agreement No. 101021347) and under Horizon Europe (Marie Skłodowska-Curie fellowship, grant agreement No. 101067759).