Tutorial:
“Deep Continual Learning”

Gido van de Ven

E-mail: gido.vandeven@kuleuven.be

Website: https://gmvandeven.github.io

Dagstuhl Seminar

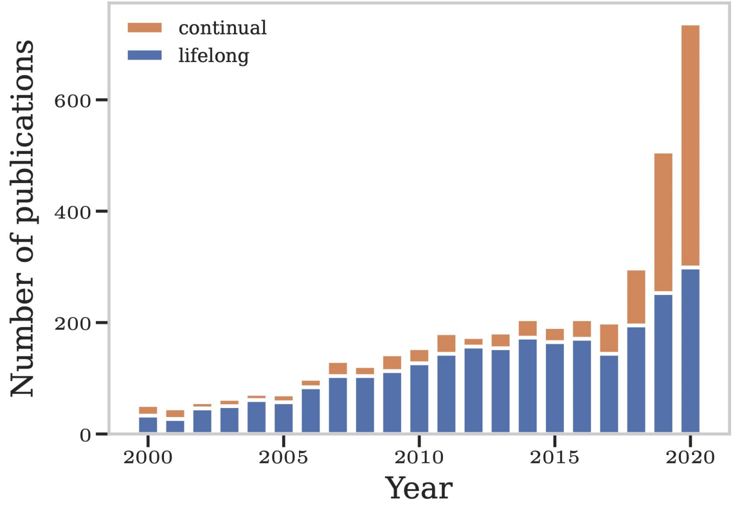
20 March 2023

Part 1:
The Continual Learning *Problem*

The term ‘continual learning’

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

- 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 arrives incrementally
    - there is non-stationarity

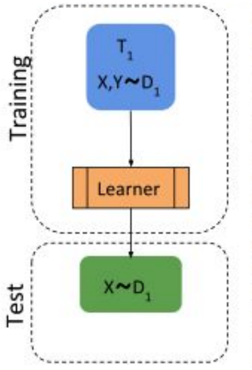


Number of machine learning publications per year, based on keyword occurrence in abstract.
Source: Mundt et al. (2022, ICLR)

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 agent learning throughout its lifetime

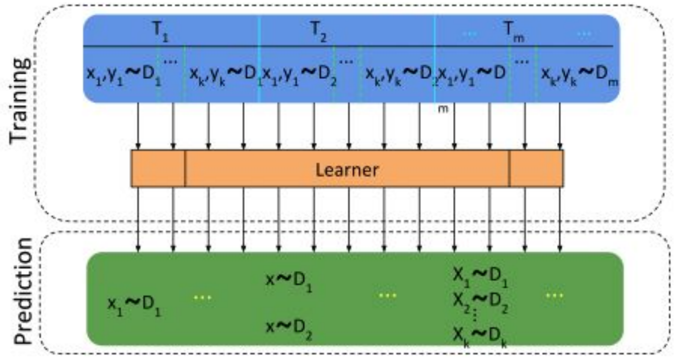
Continual learning in relation to other fields

Standard ML



- One task
- Data available at same time

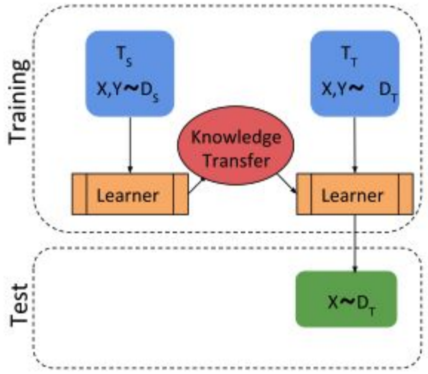
Continual Learning



- Multiple tasks
- Data arrive incrementally
- Goal: all tasks

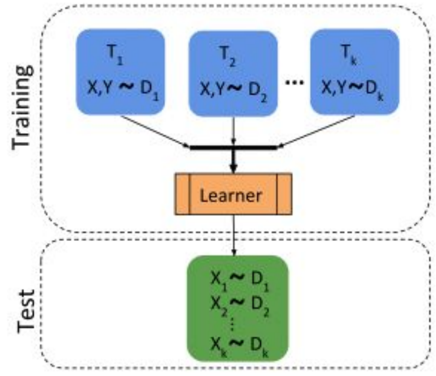
~~non-~~stationarity

Transfer Learning



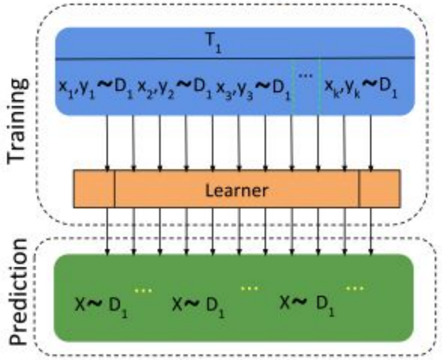
- Multiple tasks
- Data arrive incrementally
- Goal: last task

Multitask Learning



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

Online Learning

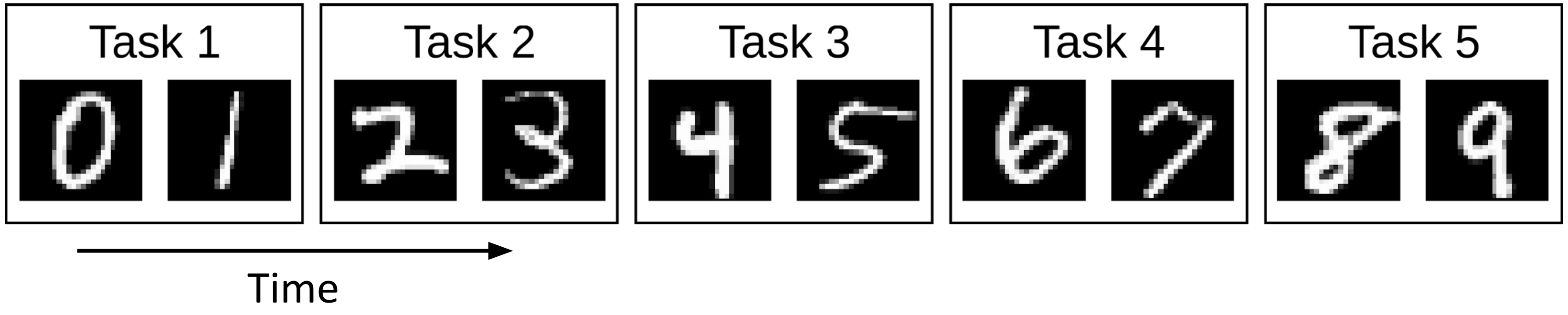


- One task
- Data arrive incrementally

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

The canonical continual learning example: Split MNIST

- MNIST dataset is split in multiple parts/episodes/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



Important problem: ***catastrophic forgetting***

→ When learning a new task, deep neural networks tend to rapidly forget past tasks

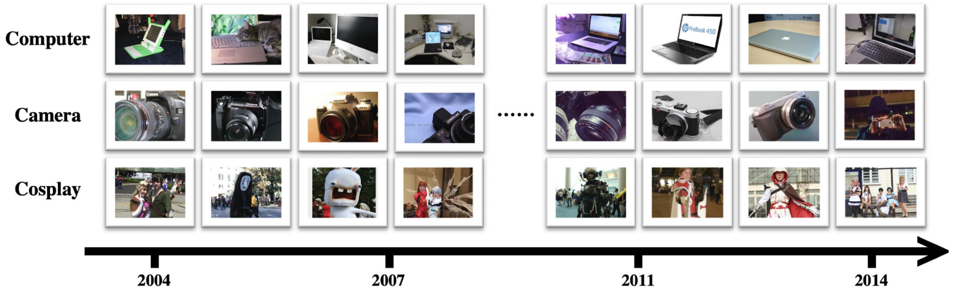
(*) Often the term “task” is used for this. Although this has some issues, given the widespread use, in this tutorial we mostly use this term.

Going beyond Split MNIST

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

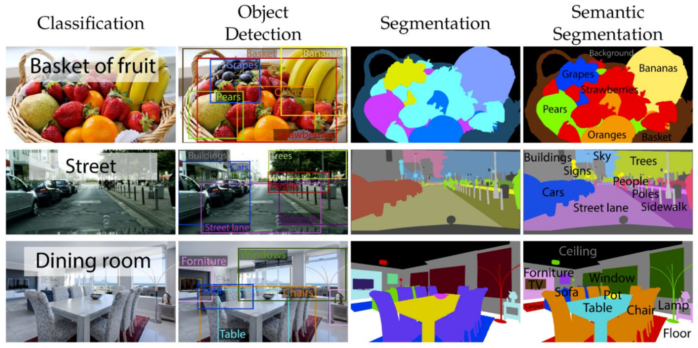
Source: van de Ven et al. (2020, Nature Communications)

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



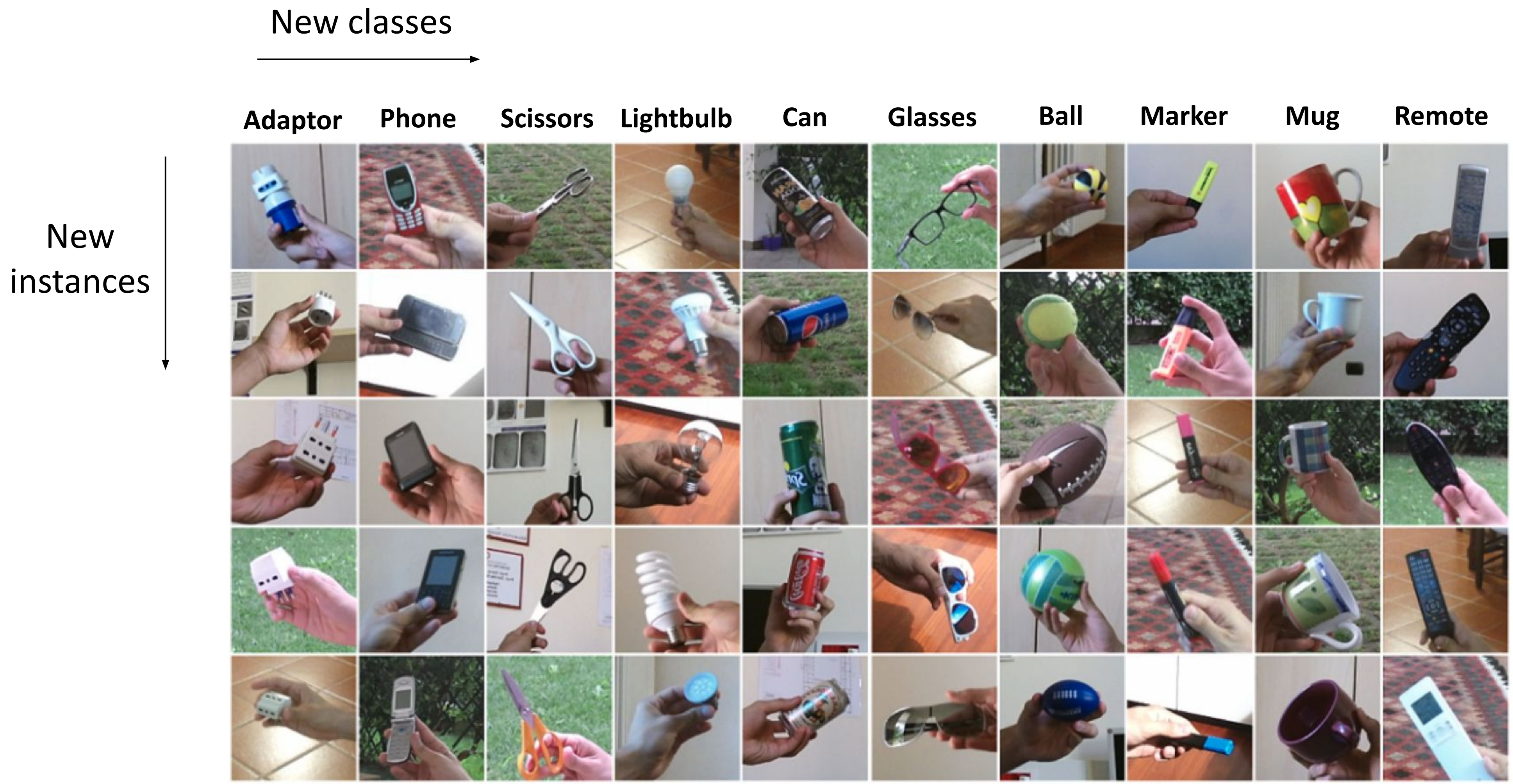
Source: Lin et al. (2021, NeurIPS Datasets and Benchmarks Track)

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



Source: Toldo et al. (2020, Technologies)

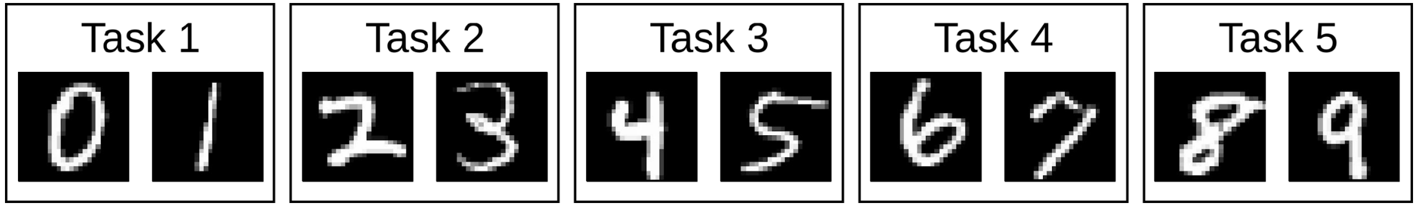
CORe50: different types of continual learning



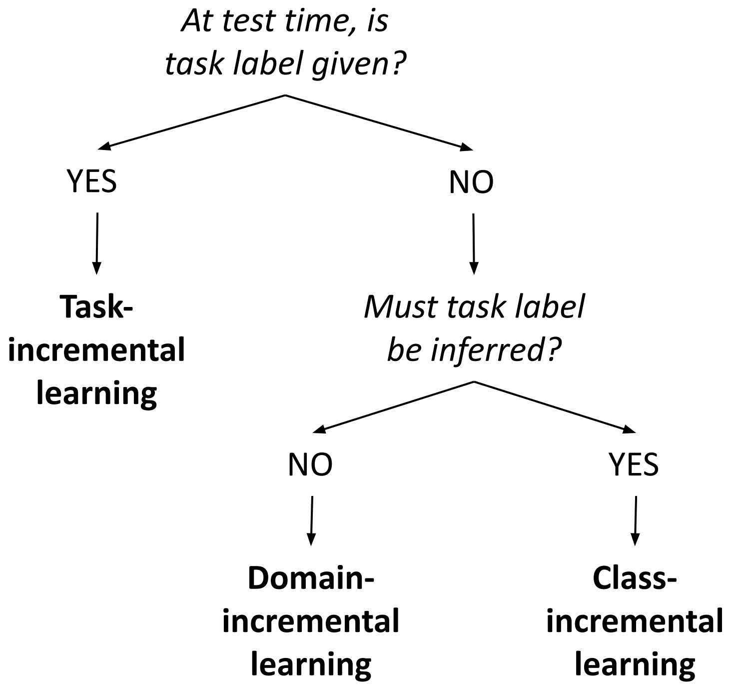
Source: Lomonaco & Maltoni (2017, CoRL)

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

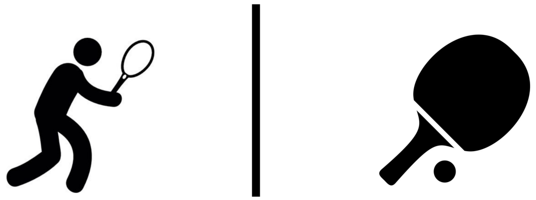


Source: van de Ven & Tolias (2018, NeurIPS workshop)

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

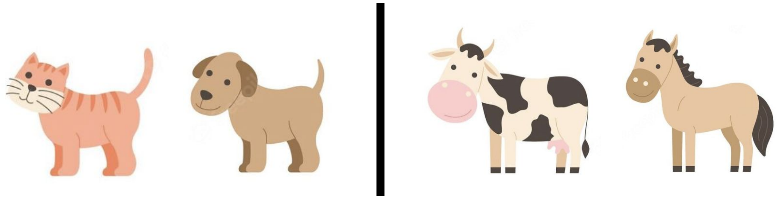


- Domain-incremental learning *(Domain-IL)*
  - Learn the same type of problem in different contexts

Important challenge: alleviate catastrophic forgetting

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

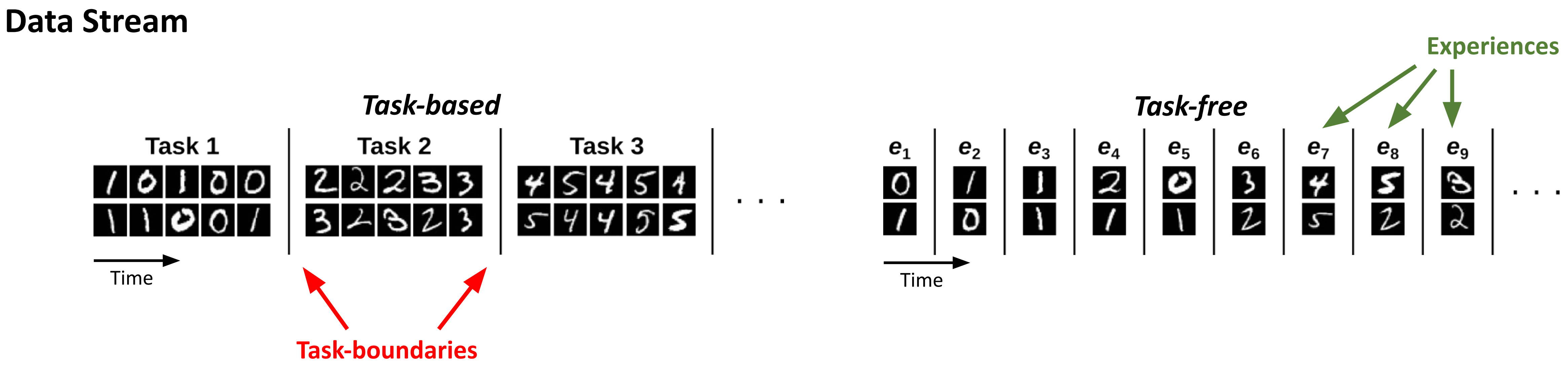
Important challenge: learn to discriminate between objects not observed together



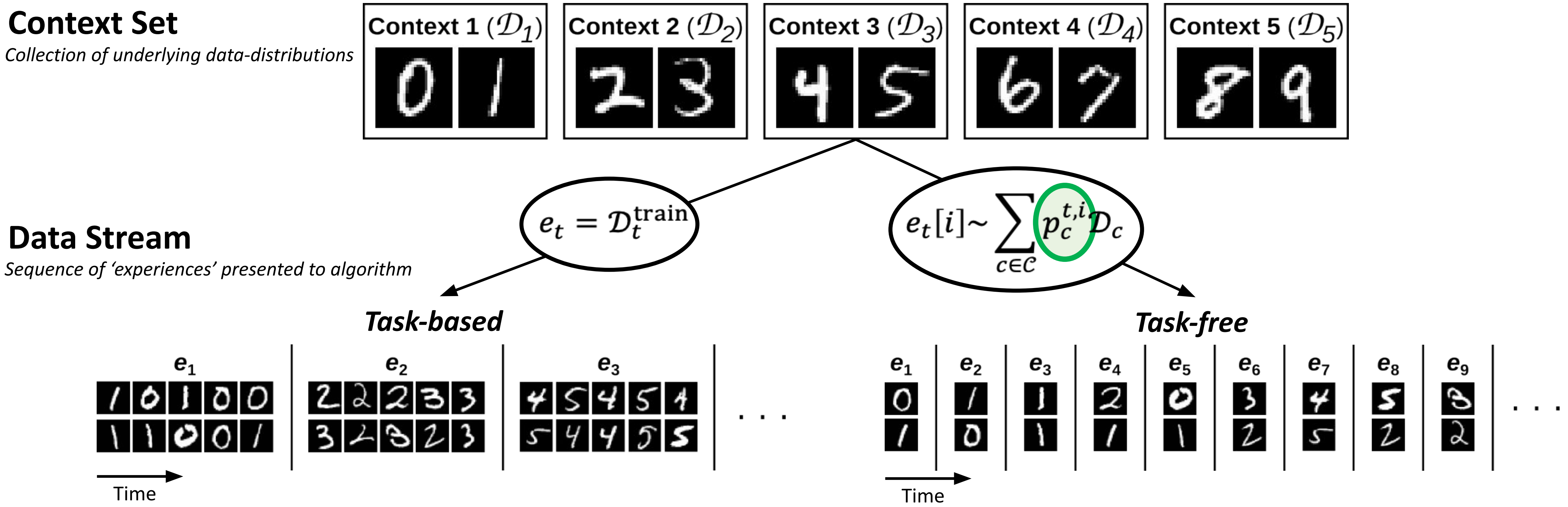
Images designed by Freepik

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

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



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



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

Context Set

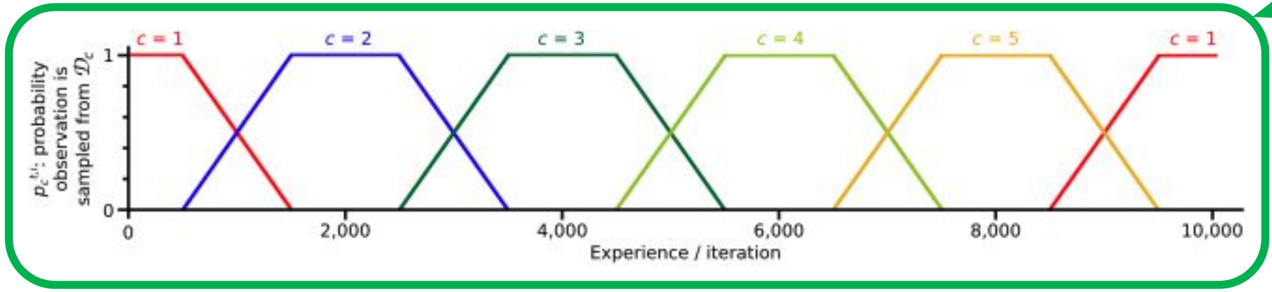
Collection of underlying data-distributions

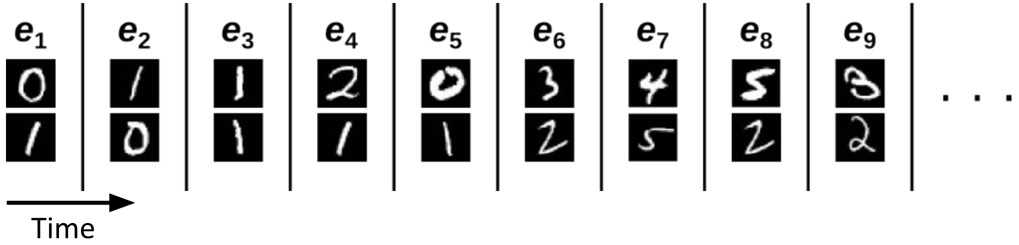
$$e_t[i] \sim \sum_{c\in\mathcal{C}} p_c^{t,i} \mathcal{D}_c$$

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?”

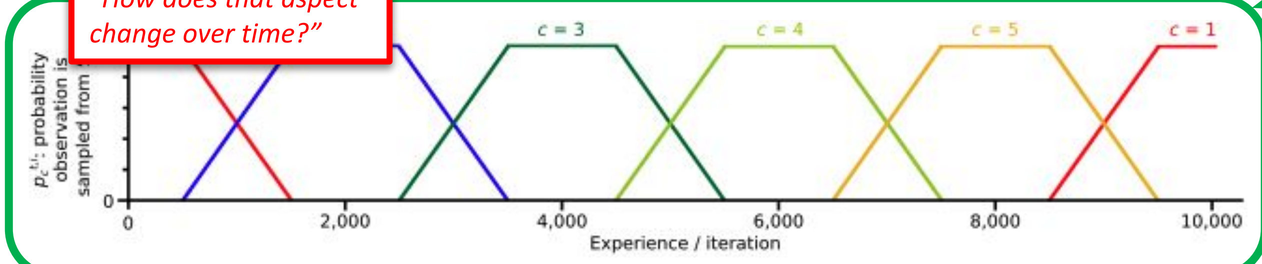
$$e_t[i] \sim \sum_{c \in \mathcal{C}} p_c^{t,i} \mathcal{D}_c$$

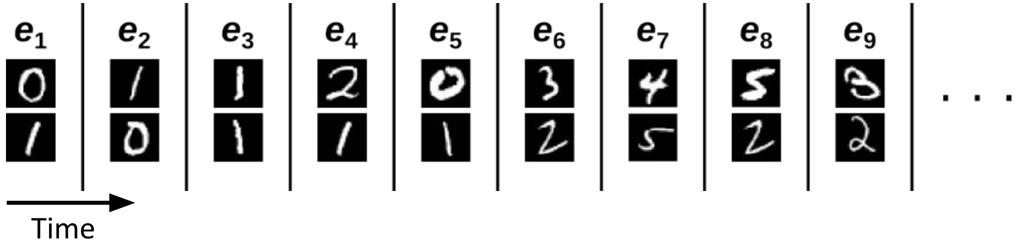
[2] Data Stream

Sequence of ‘experiences’ presented to algorithm

“How does that aspect change over time?”

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	$f: \mathcal{X} \rightarrow \mathcal{C} \times \mathcal{Y}$

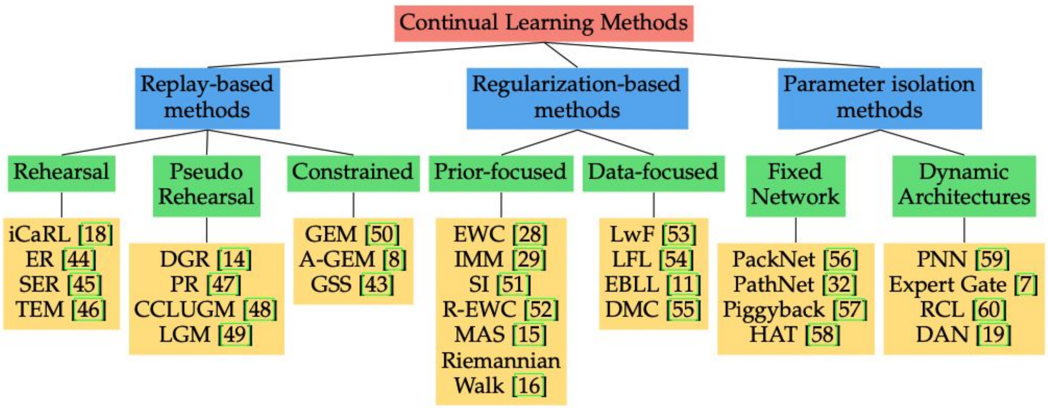
$\mathcal{X}$ = image pixel space

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

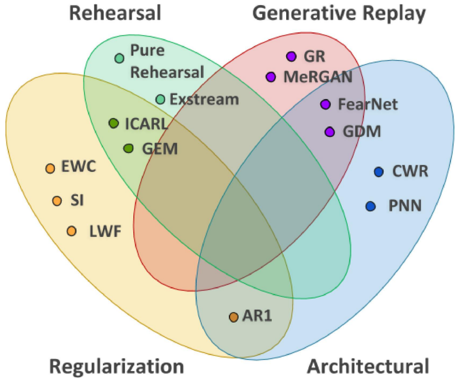
$\mathcal{Y}$ = within-context label space = {0,1}

Part 2:
Continual Learning *Strategies*

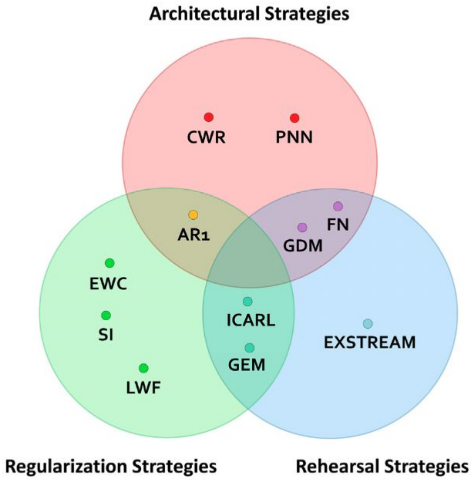
Categorizations of continual learning strategies



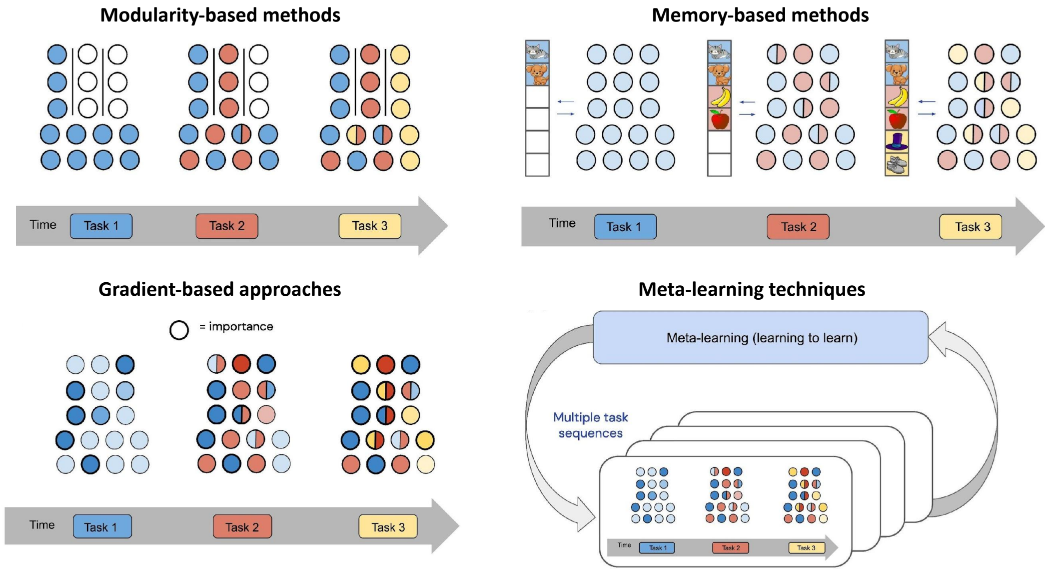
Source: De Lange et al. (2021, TPAMI)



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



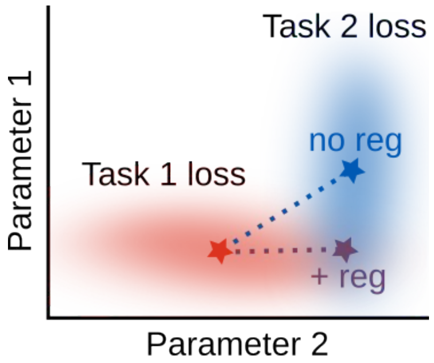
Source: Maltoni & Lomonaco (2019, Neural Networks)



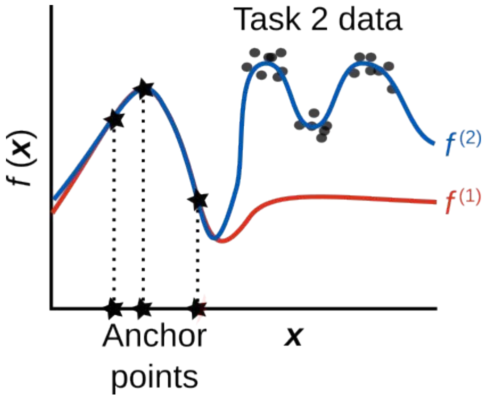
Source: Hadsell et al. (2020, Trends in Cognitive Sciences)

Categorizations of continual learning strategies

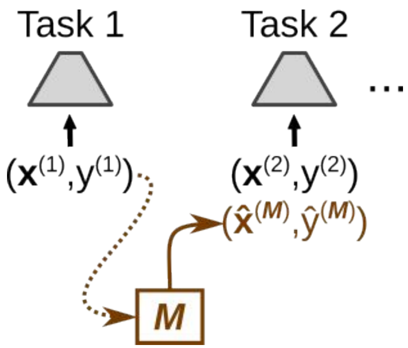
Parameter regularization



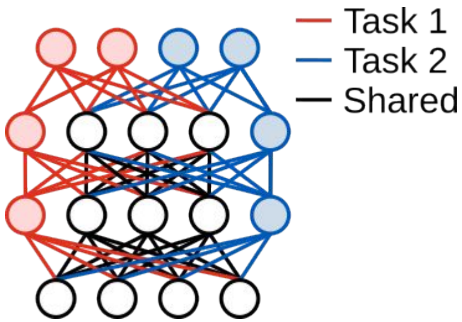
Functional regularization



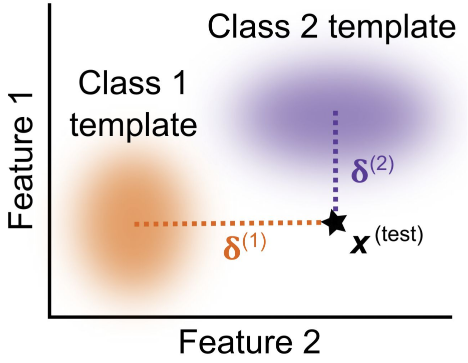
Replay



Context-specific components



Template-based classification



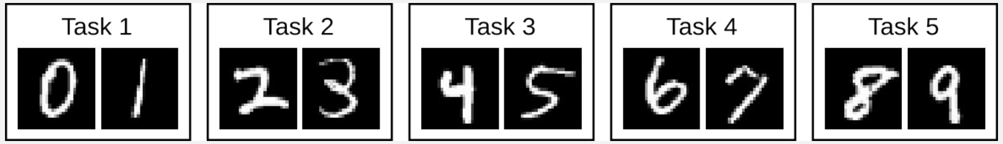
Source: van de Ven et al. (2022, Nature Machine Intelligence)

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

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

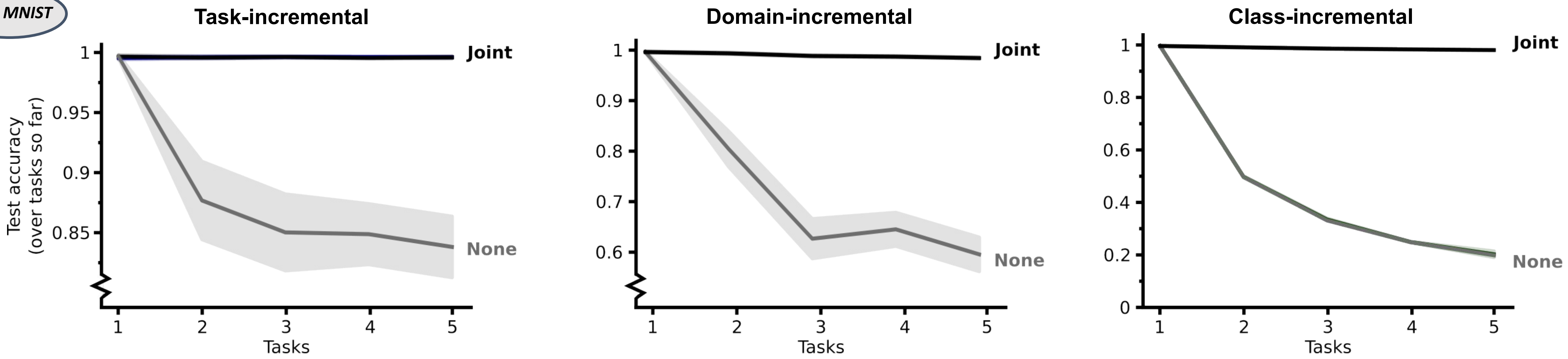
Joint: Network trained on all tasks at the same time (*upper target*)

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

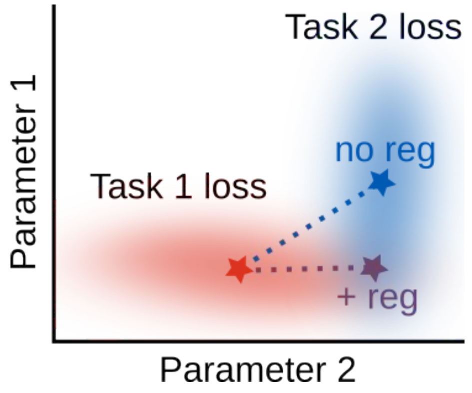


Code for these experiments: https://github.com/GMvandeVen/continual-learning

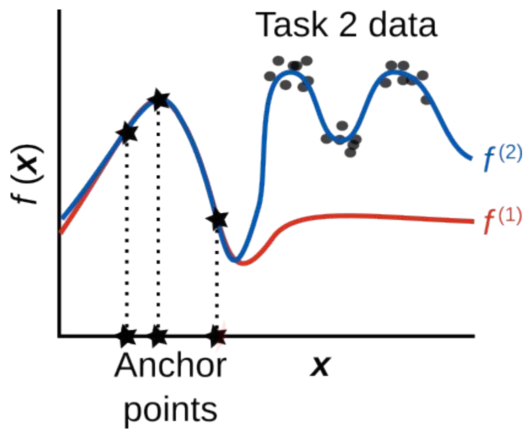
Regularization

- 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:

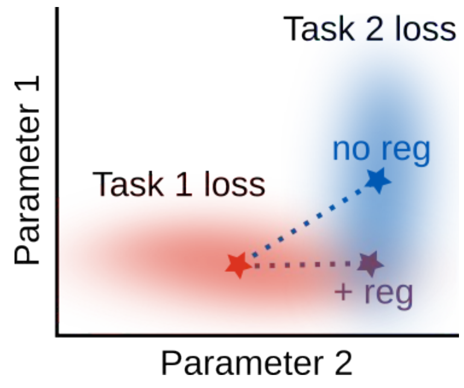
Parameter regularization



Functional regularization



Parameter regularization



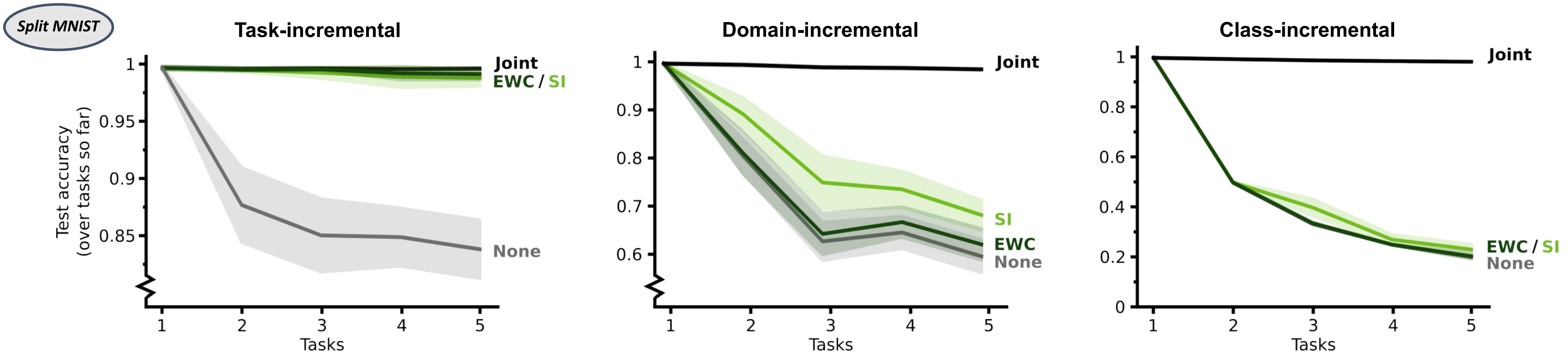
- Parameters important for past tasks are encouraged not to change too much when learning a new task

- 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)

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

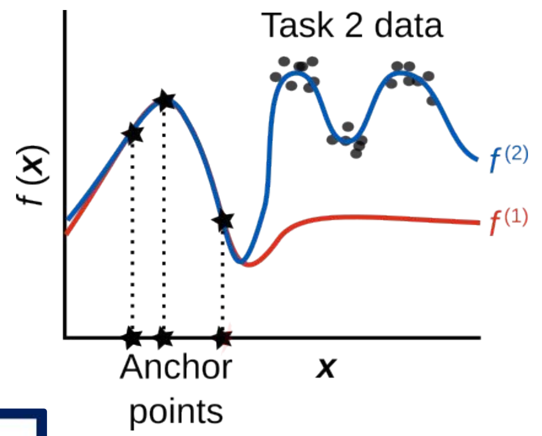
θ^* : parameters relative to which changes are penalized
Σ : estimate of how important parameters are
$||.||_\Sigma$: weighted norm



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

Functional regularization

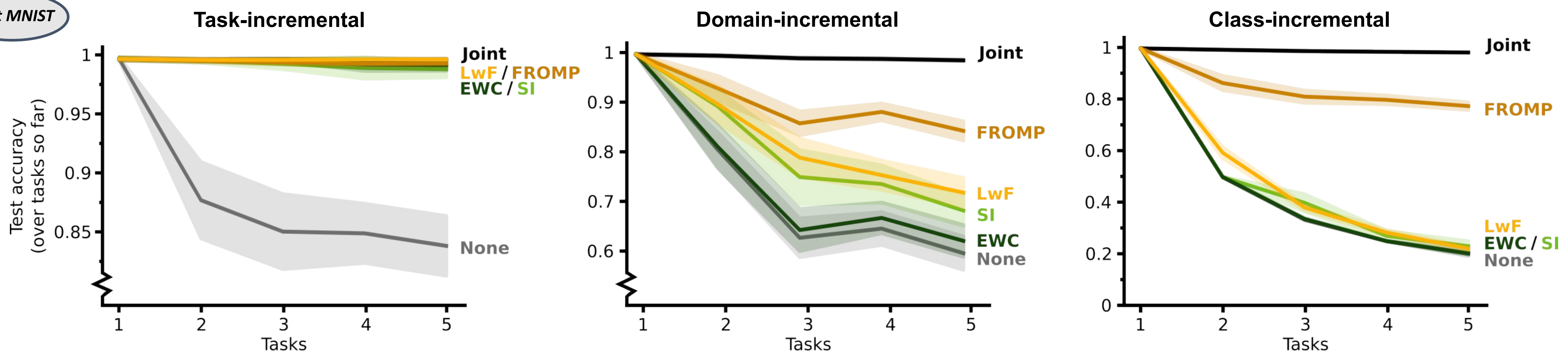
- The input-output mapping learned previously is encouraged not to change too much at a particular set of inputs (the 'anchor points')
- Also referred to as knowledge distillation
- Representative methods:
  - Learning without Forgetting [LwF] (Li & Hoiem, 2017 TPAMI)
  - Functional Regularization Of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

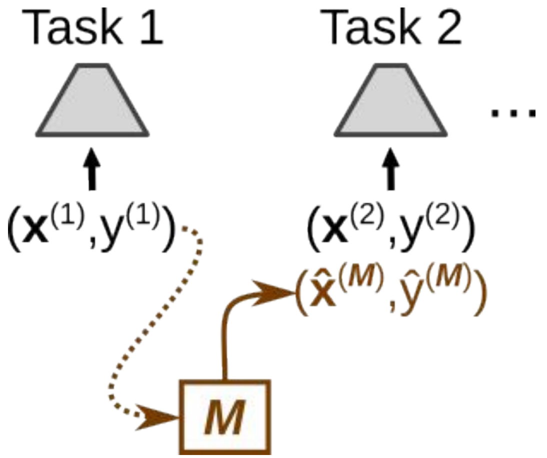


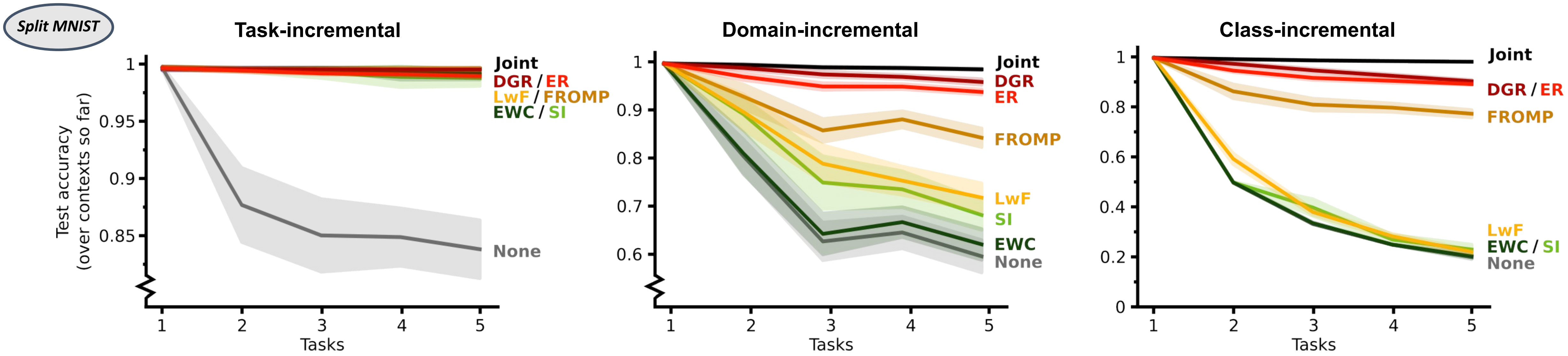
Memory buffer size (FROMP): 100 examples per class

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)



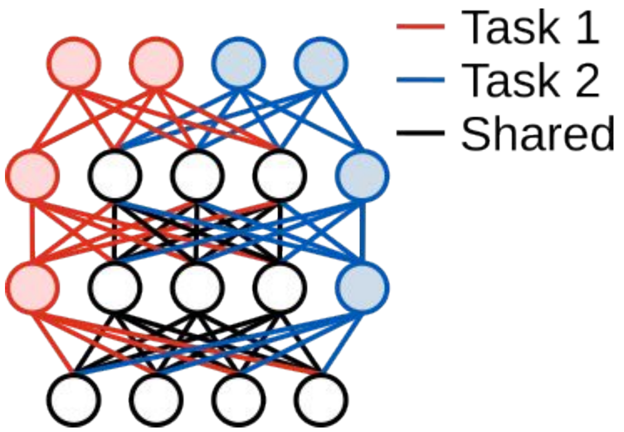


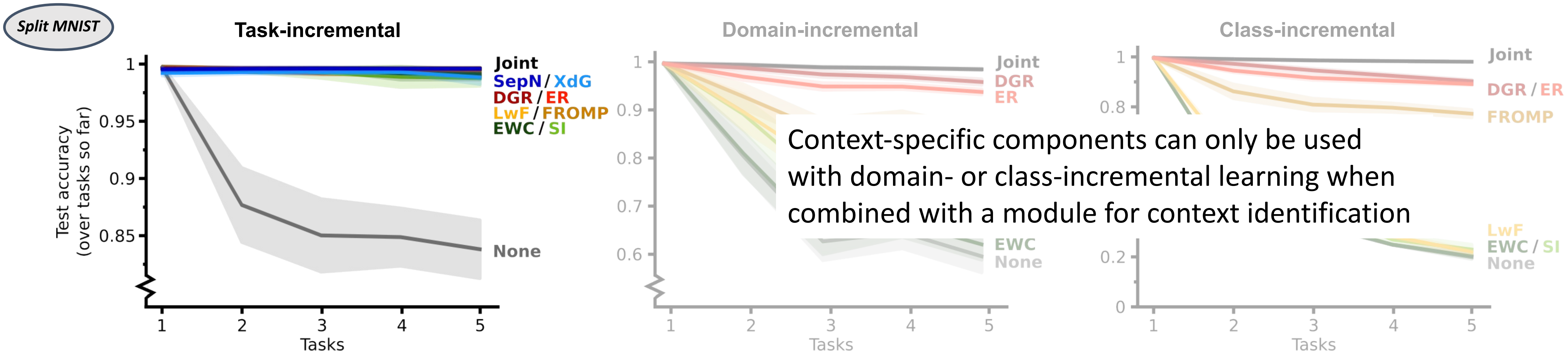
Memory buffer size (FROMP, ER): 100 examples per class

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]





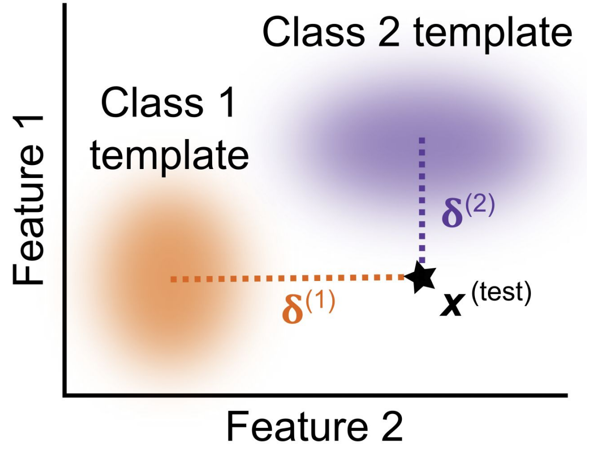
Context-specific components can only be used with domain- or class-incremental learning when combined with a module for context identification

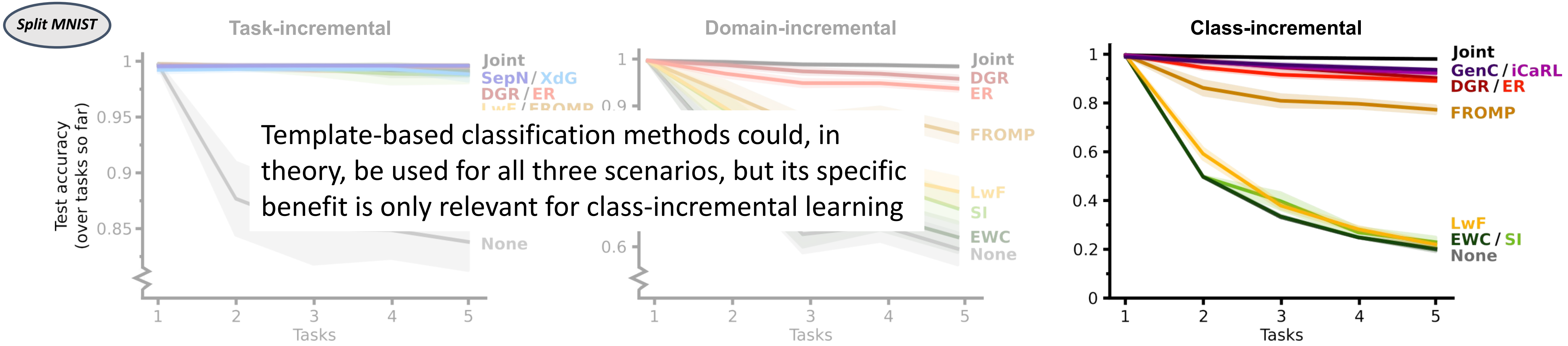
Memory buffer size (FROMP, ER): 100 examples per class

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)





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 is not a unitary problem*: we discussed **three scenarios** that differ substantially in terms of difficulty and in terms of the effectiveness of different computational strategies
- **Regularization-based methods** often have relatively low memory and computational costs, but they struggle in certain settings
- **Replay** can work well in all three scenarios, but has relatively high memory and computational costs
- **Class-incremental learning** seems to require either replay (*to allow comparing classes during training*) or template-based classification (*to allow comparing classes during inference*)