Three types of incremental learning

Gido M. van de Ven, Tinne Tuytelaars & Andreas S. Tolias

Nature Machine Intelligence **4**, 1185–1197 (2022)

BNAIC/BeNeLearn, Delft

8 November 2023

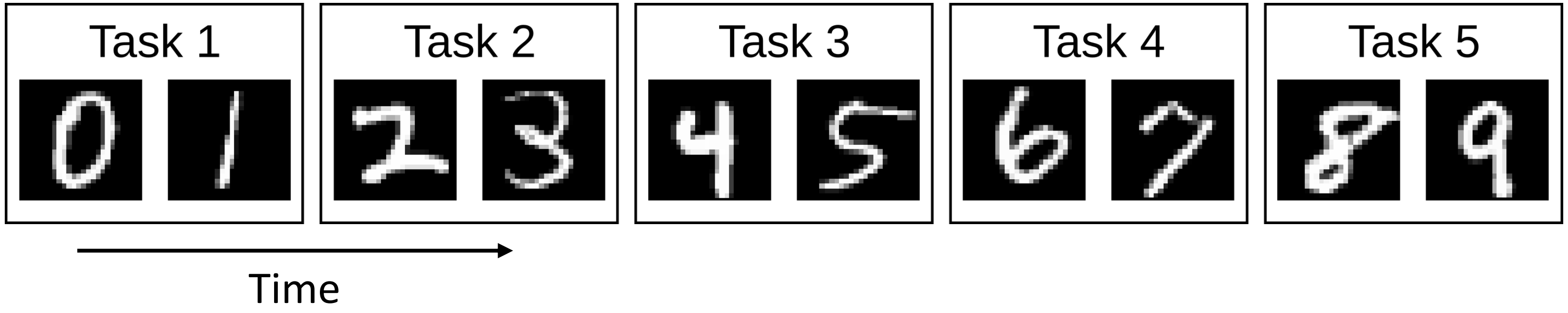
What is continual learning?

- In *classical machine learning*, an algorithm has access to all training data at the same time

- With *continual learning*, two key differences are:
  - the training data arrives incrementally
  - the distribution from which the training data is sampled changes over time

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

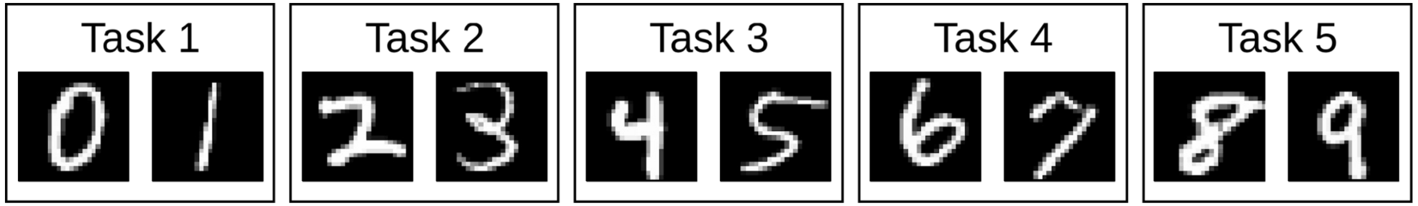


Important problem: ***catastrophic forgetting***

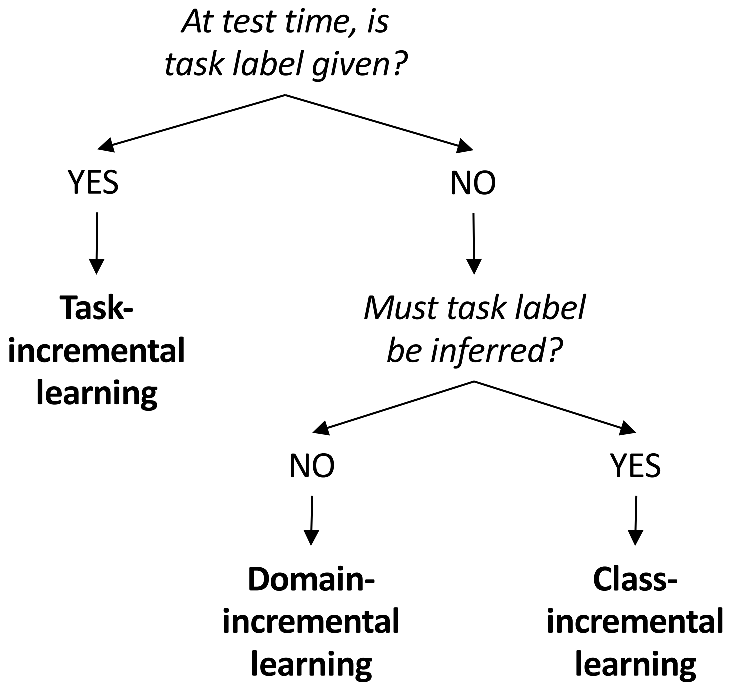
➢ When learning a new task, deep neural networks tend to rapidly forget past tasks

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

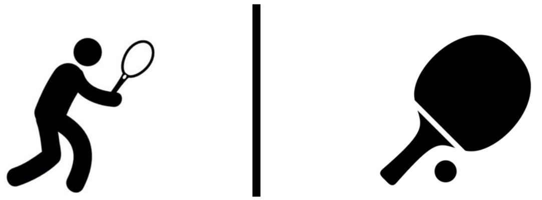


See also the preprint: van de Ven & Tolias (2019) Three scenarios for continual learning. *arXiv preprint,* https://arxiv.org/abs/1904.07734

Three continual learning scenarios: intuitively

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

Main challenge: achieve positive transfer between tasks

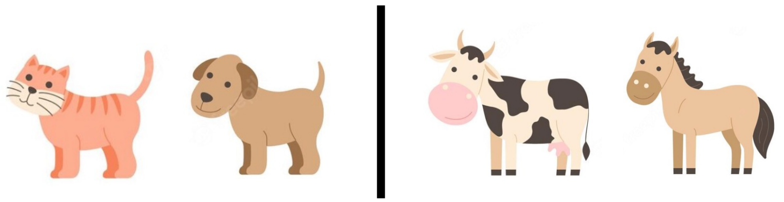


- Domain-incremental learning *(Domain-IL)*
  - Learn the same type of problem in different contexts

Main challenge: alleviate catastrophic forgetting

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

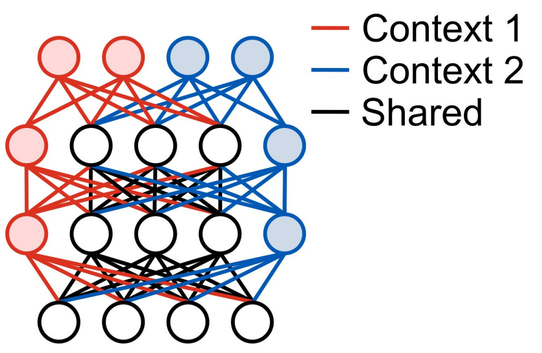
Main challenge: learn to discriminate between objects not observed together



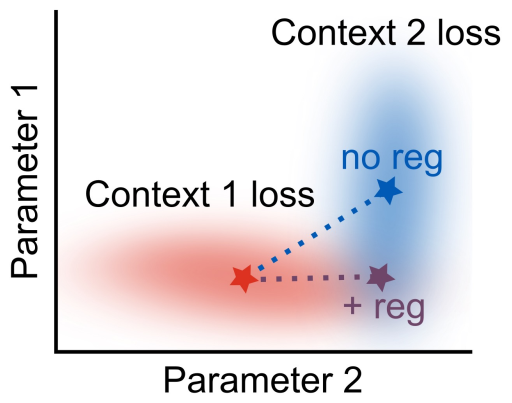
Images designed by Freepik

Strategies for continual learning

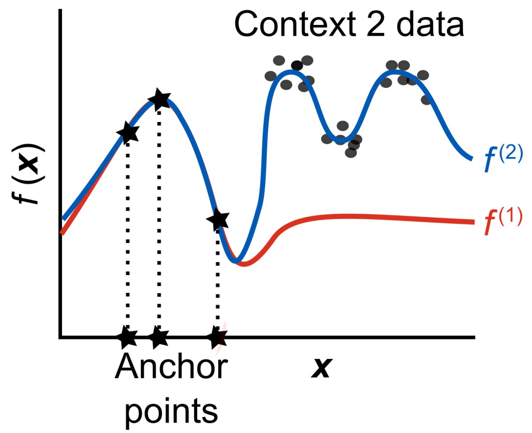
Context-specific components



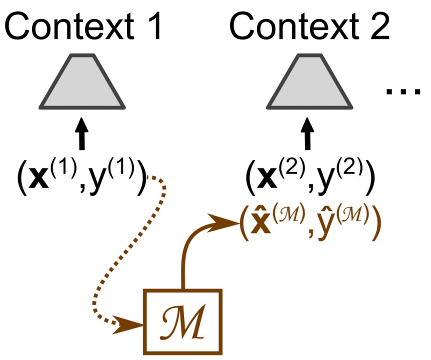
Parameter regularization



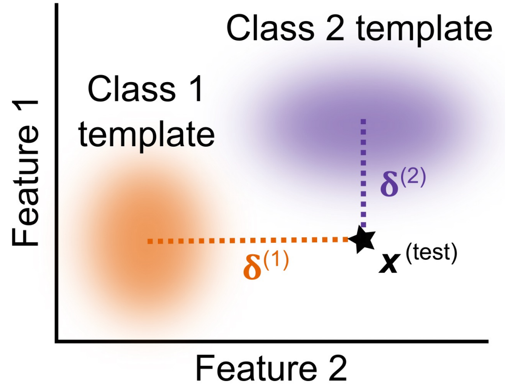
Functional regularization



Replay

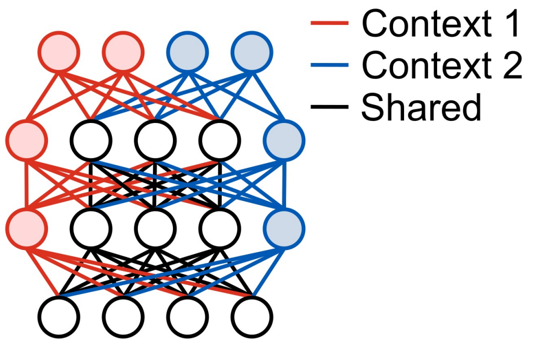


Template-based classification

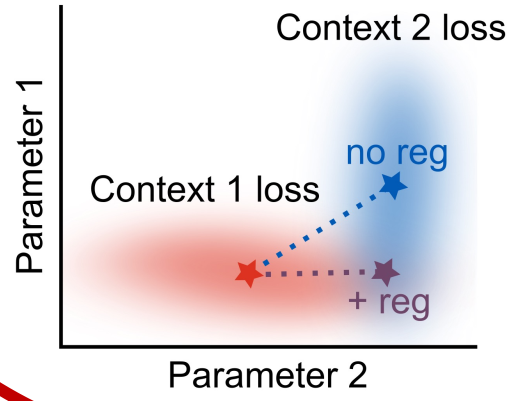


Strategies for continual learning

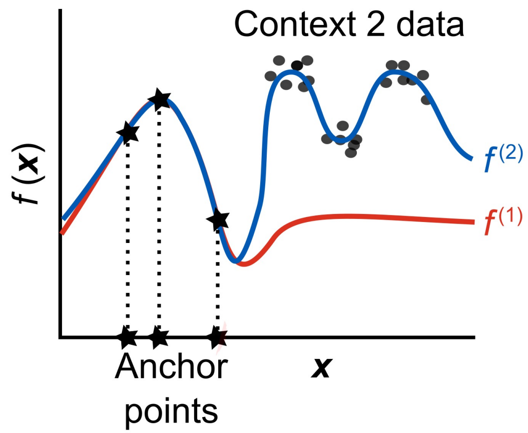
Context-specific components



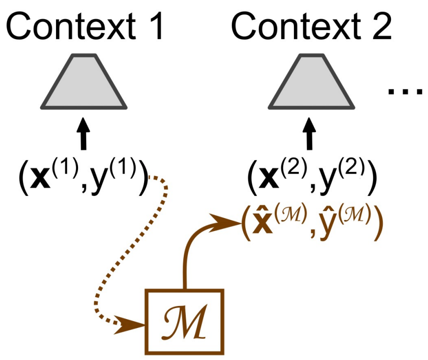
Parameter regularization



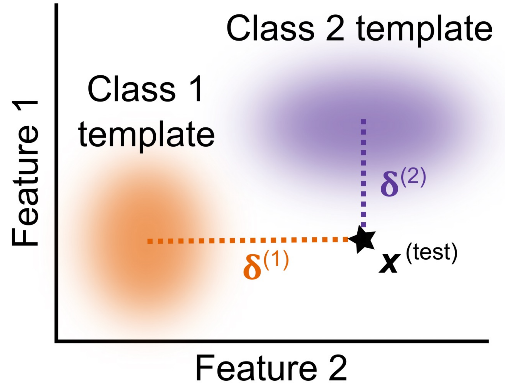
Functional regularization



Replay

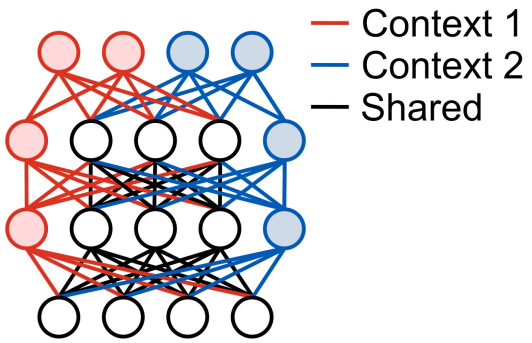


Template-based classification

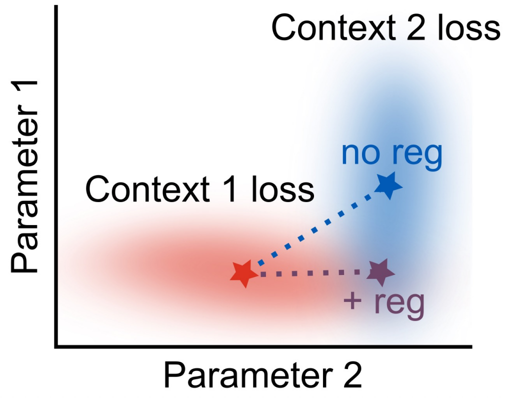


Strategies for continual learning

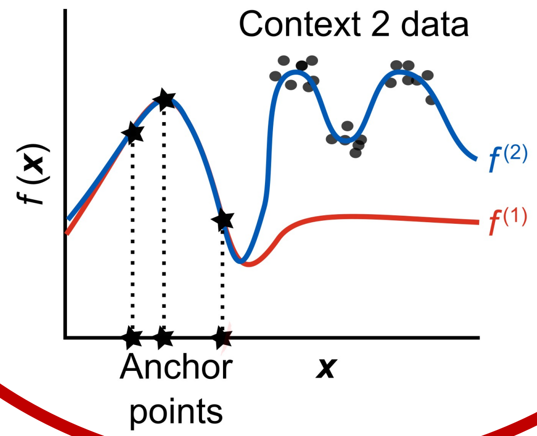
Context-specific components



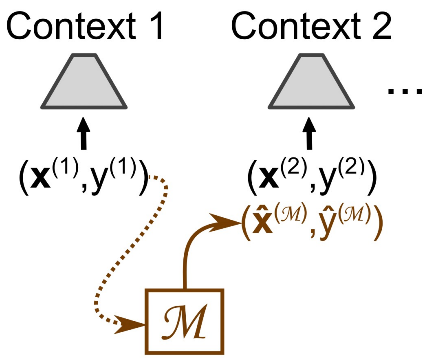
Parameter regularization



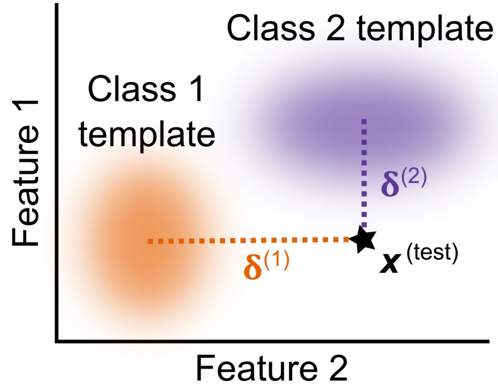
Functional regularization



Replay

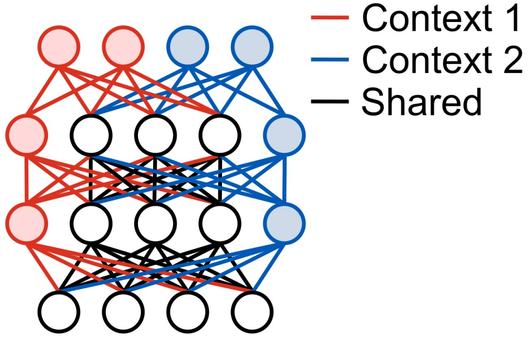


Template-based classification

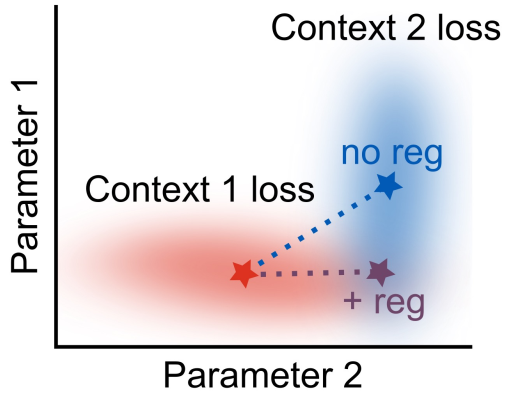


Strategies for continual learning

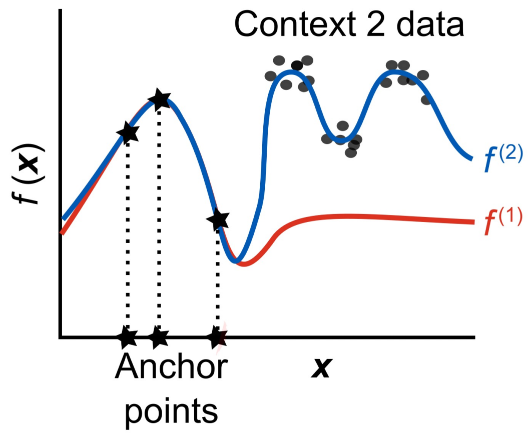
Context-specific components



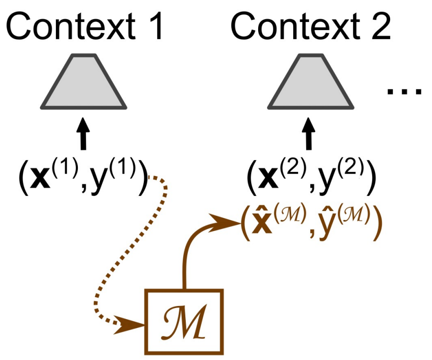
Parameter regularization



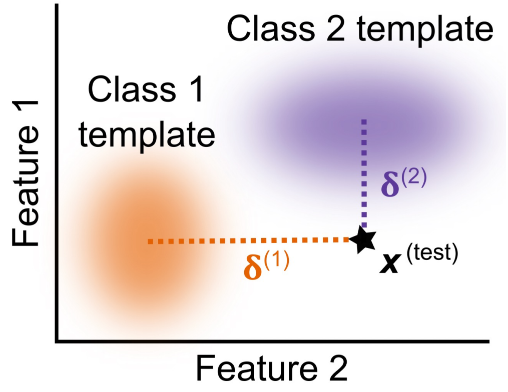
Functional regularization



Replay

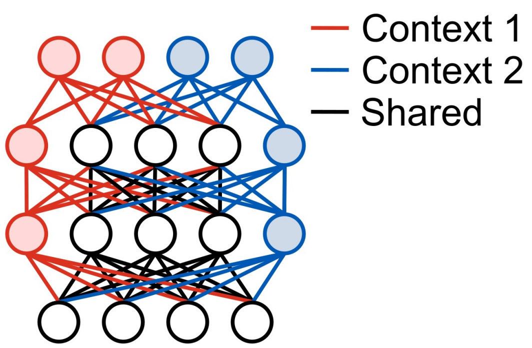


Template-based classification

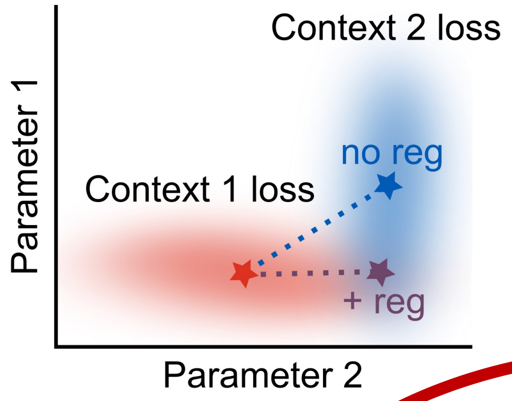


Strategies for continual learning

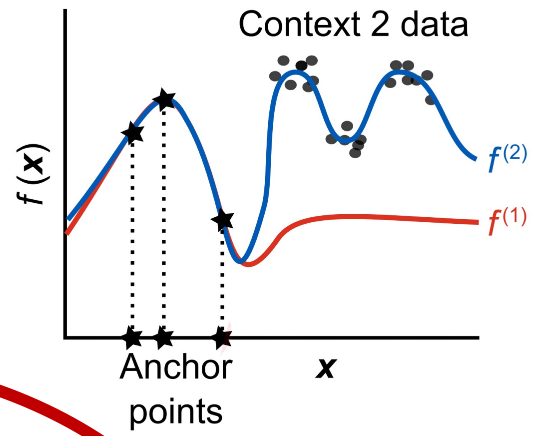
Context-specific components



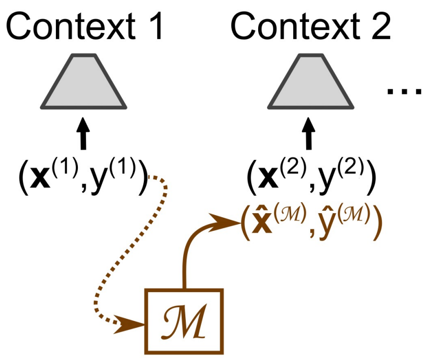
Parameter regularization



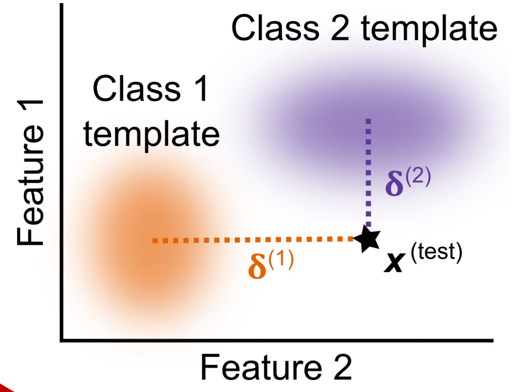
Functional regularization



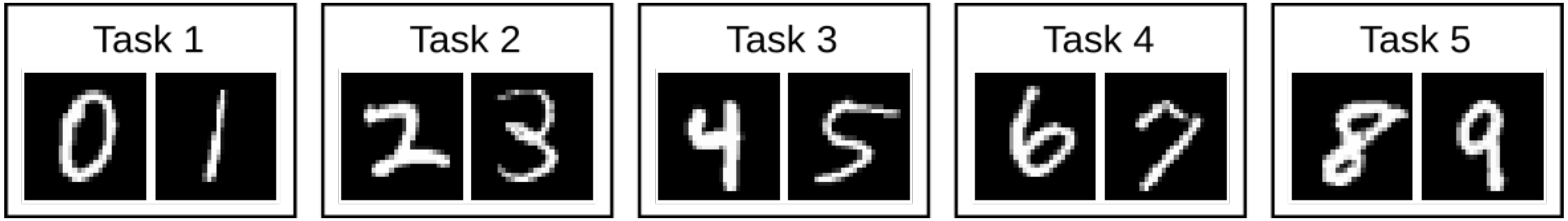
Replay



Template-based classification



Empirical comparison: Split MNIST



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

The same sequence of contexts can be “performed” in three different ways:

→ use for a direct comparison between the three scenarios

Empirical comparison: Split MNIST

Strategy	Method	Budget	GM	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower target*			*84.32 (± 0.99)*	*60.13 (± 1.66)*	*19.89 (± 0.02)*
	Joint – upper target			*99.67 (± 0.03)*	*98.59 (± 0.05)*	*98.17 (± 0.04)*
Context-specific components	Separate Networks	-	-	99.57 (± 0.03)	-	-
	XdG	-	-	99.10 (± 0.10)	-	-
Parameter regularization	EWC	-	-	99.06 (± 0.15)	63.03 (± 1.58)	20.64 (± 0.52)
	SI	-	-	99.20 (± 0.11)	66.94 (± 1.13)	21.20 (± 0.57)
Functional regularization	LwF	-	-	99.60 (± 0.03)	71.18 (± 1.42)	21.89 (± 0.32)
	FROMP	100	-	99.12 (± 0.13)	84.86 (± 1.02)	77.38 (± 0.64)
Replay	DGR	-	yes	99.50 (± 0.03)	95.57 (± 0.30)	90.35 (± 0.24)
	BI-R	-	yes	99.61 (± 0.03)	97.26 (± 0.15)	94.41 (± 0.15)
	ER	100	-	98.98 (± 0.07)	93.75 (± 0.24)	88.79 (± 0.20)
	A-GEM	100	-	98.54 (± 0.10)	87.67 (± 1.33)	65.10 (± 3.64)
Template-based classification	Generative Classifier	-	yes	-	-	93.82 (± 0.06)
	iCaRL	100	-	-	-	92.49 (± 0.12)

Shown is final test accuracy (as %, averaged over all contexts). Academic continual learning setting was used. 'Budget' indicates number of samples per class stored in memory, 'GM' indicates generative model was learned using extra parameters. Experiments were run 20 times, reported is mean (± SEM). ***More comparisons in the paper: Split CIFAR-100 and a 'task-free' version of Split MNIST.***

PyTorch code for all experiments: https://github.com/GMvandeVen/continual-learning

For method abbreviations and references, see extra slide.

Empirical comparison: Split MNIST

Strategy	Method	Budget	GM	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower target*			*84.32 (± 0.99)*	*60.13 (± 1.66)*	*19.89 (± 0.02)*
	Joint – upper target			*99.67 (± 0.03)*	*98.59 (± 0.05)*	*98.17 (± 0.04)*
Context-specific components	Separate Networks	-	-	99.57 (± 0.03)	-	-
	XdG	-	-	99.10 (± 0.10)	-	-
Parameter regularization	EWC	-	-	99.06 (± 0.15)	63.03 (± 1.58)	20.64 (± 0.52)
	SI	-	-	99.20 (± 0.11)	66.94 (± 1.13)	21.20 (± 0.57)
Functional regularization	LwF	-	-	99.60 (± 0.03)	71.18 (± 1.42)	21.89 (± 0.32)
	FROMP	100	-	99.12 (± 0.13)	84.86 (± 1.02)	77.38 (± 0.64)
Replay	DGR	-	yes	99.50 (± 0.03)	95.57 (± 0.30)	90.35 (± 0.24)
	BI-R	-	yes	99.61 (± 0.03)	97.26 (± 0.15)	94.41 (± 0.15)
	ER	100	-	98.98 (± 0.07)	93.75 (± 0.24)	88.79 (± 0.20)
	A-GEM	100	-	98.54 (± 0.10)	87.67 (± 1.33)	65.10 (± 3.64)
Template-based classification	Generative Classifier	-	yes	-	-	93.82 (± 0.06)
	iCaRL	100	-	-	-	92.49 (± 0.12)

Shown is final test accuracy (as %, averaged over all contexts). Academic continual learning setting was used. 'Budget' indicates number of samples per class stored in memory, 'GM' indicates generative model was learned using extra parameters. Experiments were run 20 times, reported is mean (± SEM). ***More comparisons in the paper: Split CIFAR-100 and a 'task-free' version of Split MNIST.***

PyTorch code for all experiments: https://github.com/GMvandeVen/continual-learning

For method abbreviations and references, see extra slide.

Empirical comparison: Split MNIST

Strategy	Method	Budget	GM	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower target*			*84.32 (± 0.99)*	*60.13 (± 1.66)*	*19.89 (± 0.02)*
	Joint – upper target			*99.67 (± 0.03)*	*98.59 (± 0.05)*	*98.17 (± 0.04)*
Context-specific components	Separate Networks	-	-	99.57 (± 0.03)	-	-
	XdG	-	-	99.10 (± 0.10)	-	-
Parameter regularization	EWC	-	-	99.06 (± 0.15)	63.03 (± 1.58)	20.64 (± 0.52)
	SI	-	-	99.20 (± 0.11)	66.94 (± 1.13)	21.20 (± 0.57)
Functional regularization	LwF	-	-	99.60 (± 0.03)	71.18 (± 1.42)	21.89 (± 0.32)
	FROMP	100	-	99.12 (± 0.13)	84.86 (± 1.02)	77.38 (± 0.64)
Replay	DGR	-	yes	99.50 (± 0.03)	95.57 (± 0.30)	90.35 (± 0.24)
	BI-R	-	yes	99.61 (± 0.03)	97.26 (± 0.15)	94.41 (± 0.15)
	ER	100	-	98.98 (± 0.07)	93.75 (± 0.24)	88.79 (± 0.20)
	A-GEM	100	-	98.54 (± 0.10)	87.67 (± 1.33)	65.10 (± 3.64)
Template-based classification	Generative Classifier	-	yes	-	-	93.82 (± 0.06)
	iCaRL	100	-	-	-	92.49 (± 0.12)

Shown is final test accuracy (as %, averaged over all contexts). Academic continual learning setting was used. 'Budget' indicates number of samples per class stored in memory, 'GM' indicates generative model was learned using extra parameters. Experiments were run 20 times, reported is mean (± SEM). ***More comparisons in the paper: Split CIFAR-100 and a 'task-free' version of Split MNIST.***

PyTorch code for all experiments: https://github.com/GMvandeVen/continual-learning

For method abbreviations and references, see extra slide.

Empirical comparison: Split MNIST

Strategy	Method	Budget	GM	Task-IL	Domain-IL	Class-IL
Baselines	*None – lower target*			*84.32 (± 0.99)*	*60.13 (± 1.66)*	*19.89 (± 0.02)*
	Joint – upper target			*99.67 (± 0.03)*	*98.59 (± 0.05)*	*98.17 (± 0.04)*
Context-specific components	Separate Networks	-	-	99.57 (± 0.03)	-	-
	XdG	-	-	99.10 (± 0.10)	-	-
Parameter regularization	EWC	-	-	99.06 (± 0.15)	63.03 (± 1.58)	20.64 (± 0.52)
	SI	-	-	99.20 (± 0.11)	66.94 (± 1.13)	21.20 (± 0.57)
Functional regularization	LwF	-	-	99.60 (± 0.03)	71.18 (± 1.42)	21.89 (± 0.32)
	FROMP	100	-	99.12 (± 0.13)	84.86 (± 1.02)	77.38 (± 0.64)
Replay	DGR	-	yes	99.50 (± 0.03)	95.57 (± 0.30)	90.35 (± 0.24)
	BI-R	-	yes	99.61 (± 0.03)	97.26 (± 0.15)	94.41 (± 0.15)
	ER	100	-	98.98 (± 0.07)	93.75 (± 0.24)	88.79 (± 0.20)
	A-GEM	100	-	98.54 (± 0.10)	87.67 (± 1.33)	65.10 (± 3.64)
Template-based classification	Generative Classifier	-	yes	-	-	93.82 (± 0.06)
	iCaRL	100	-	-	-	92.49 (± 0.12)

Shown is final test accuracy (as %, averaged over all contexts). Academic continual learning setting was used. 'Budget' indicates number of samples per class stored in memory, 'GM' indicates generative model was learned using extra parameters. Experiments were run 20 times, reported is mean (± SEM). ***More comparisons in the paper: Split CIFAR-100 and a 'task-free' version of Split MNIST.***

PyTorch code for all experiments: https://github.com/GMvandeVen/continual-learning

For method abbreviations and references, see extra slide.

Summary

- *Continual learning is not a unitary problem*: there are **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*)
- More details: *van de Ven et al. (2022, Nature Machine Intelligence)*

Funding acknowledgements

This research project has been supported by an IBRO-ISN Research Fellowship, by the ERC-funded project KeepOnLearning (reference number 101021347), by the Lifelong Learning Machines (L2M) program of the Defence Advanced Research Projects Agency (DARPA) via contract number HR0011-18-2-0025 and by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DoI/IBC) contract number D16PC00003. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, IARPA, DoI/IBC, or the U.S. Government.