Pervasive AI

ECML-PKDD 2023 Tutorial

Davide Bacciu, Antonio Carta, Patrizio Dazzi, Claudio Gallicchio
University of Pisa, Italy

http://pai.di.unipi.it/tutorial-on-continual-learning-in-distributed-and-heterogeneous-systems/

Continual Learning

Davide Bacciu, **Antonio Carta**, Patrizio Dazzi, Claudio Gallicchio
University of Pisa, Italy

Outline

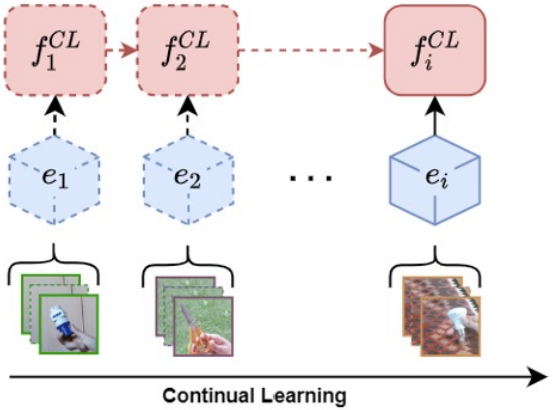
- Continual Learning
- CL for Edge Devices
- Distributed CL

Introduction to Continual Learning

Continual Learning

CL = **Incremental Learning from a non-stationary stream**

+ **environment information**: task labels, task boundaries, ...
+ **constraints**: computational/memory limits, privacy, ...
+ **metrics**: minimize forgetting, maximize transfer, ...

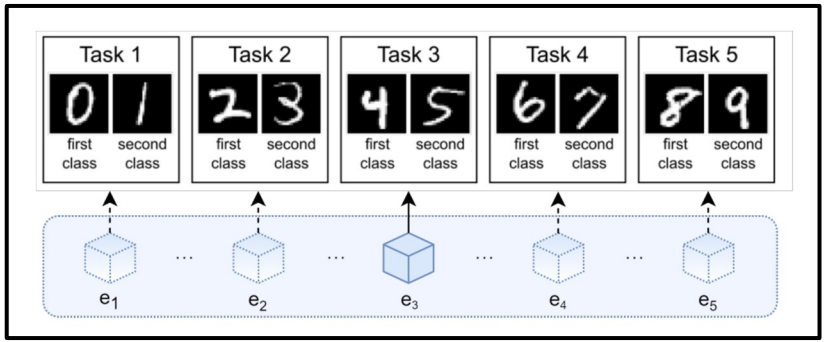


Suggested review:
T. Lesort et al. “Continual Learning for Robotics: Definition, Framework, Learning Strategies, Opportunities and Challenges.” *Information Fusion*. https://doi.org/10.1016/j.inffus.2019.12.004.

Environment Information – Nomenclature

Different streams require different methods

- **Stream**: A list of experiences, each providing a batch of data and some additional information (e.g. task labels)
- **Batch/Online: How much data** do we have in each experience?
- **Class/Domain-incremental**: Do we know the **type of shifts**?
- Do we know **when** the shifts happen?
- Do we have **task labels** at training/inference time?

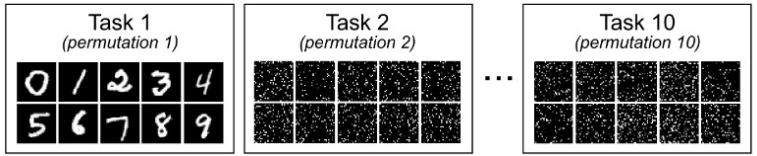


Three Common Scenarios

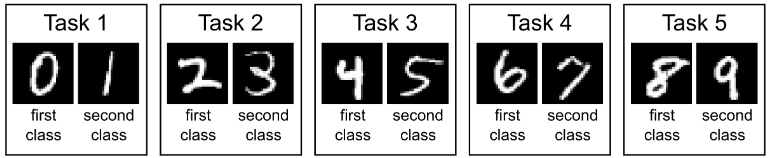
- **Domain-incremental**: each experience provides new instances for old classes. Old instances are never seen again (in the training stream).
- **Class-Incremental**: each experience provides new classes. Old classes are never revisited (in the training stream).
- **Task-Incremental**: each experience provides task labels

- **task-aware/task-agnostic** to highlight presence of task labels

Permuted MNIST



SplitMNIST

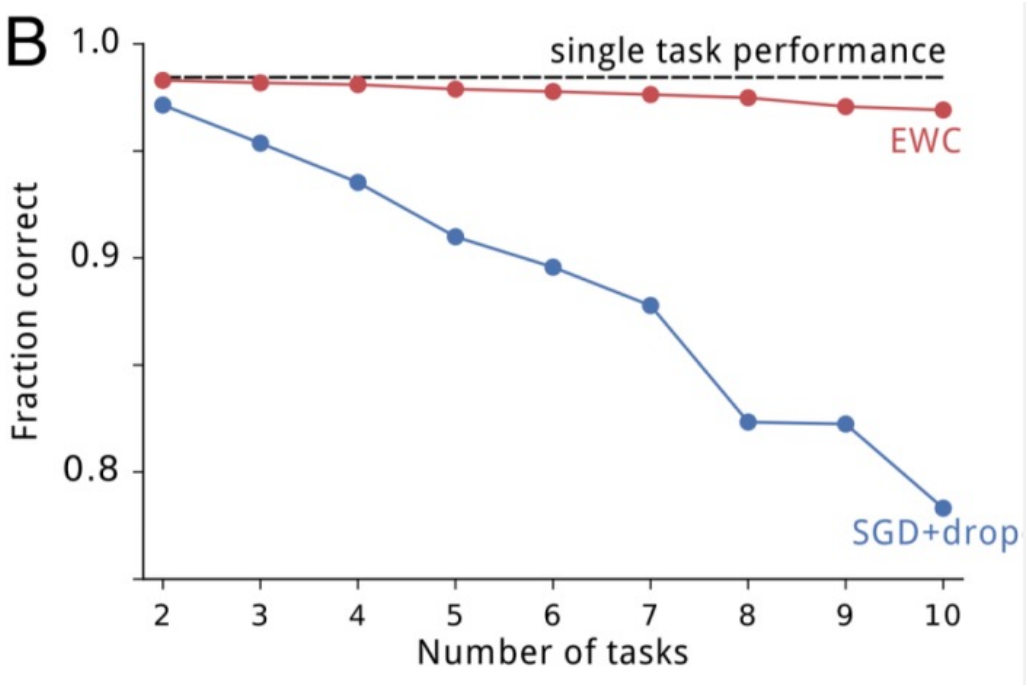


Ven, Gido M. van de, and Andreas S. Tolias. “Three Scenarios for Continual Learning.” *Arxiv Preprint*, April 15, 2019. https://doi.org/10.48550/arXiv.1904.07734.

“Catastrophic Forgetting”

Deep neural networks completely and abruptly forget previously learned information upon learning new information.

Catastrophic Forgetting

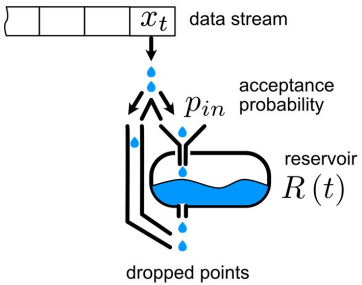


Kirkpatrick, J. et al. “Overcoming Catastrophic Forgetting in Neural Networks.” *PNAS*, 2017, 201611835.

Methodologies

Replay

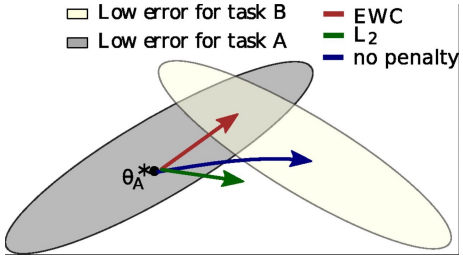
- Keep a buffer of old samples
- Rehearse old samples



Reservoir Sampling

Regularization

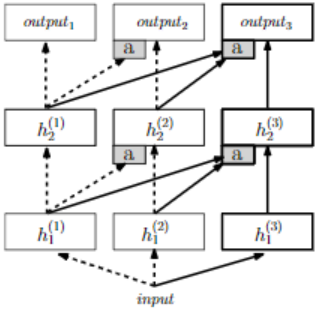
- Regularize the model to balance learning and forgetting



Elastic Weight Consolidation

Architectural

- Expand the model over time with new units/layers



Progressive Neural Networks

Image from https://towardsdatascience.com/reservoir-sampling-for-efficient-stream-processing-97f47f85c11b

CL for Edge Devices

CL for Edge Devices

Applications

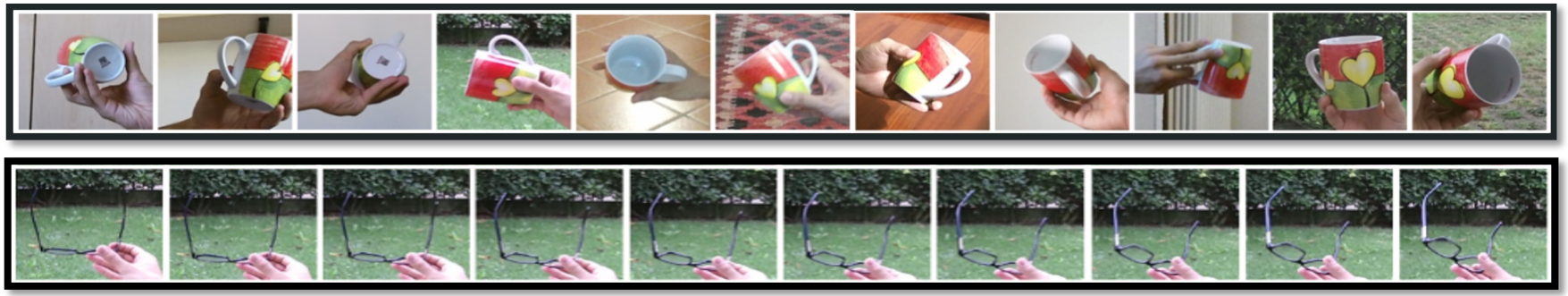
- Model personalization
- Wearable sensors
- Stream of videos

Edge devices bring their own constraints

- **Privacy**: often we cannot save the user's data
- **Efficiency**: energy, memory, and computational power are limited resources
- **Latency**: often require online training and predictions

Continuous Object Recognition: CORe50

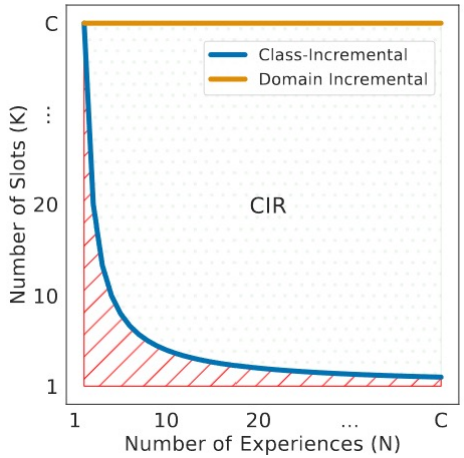
- **Continuous Object Recognition**
  - 50 classes
  - Short videos of object manipulation with different background
  - Temporal coherence from videos
- Many scenarios: batch, online, with repetitions.
- **Advantages**: online, realistic streams



Lomonaco V. and Maltoni D. CORe50: a New Dataset and Benchmark for Continuous Object Recognition. CoRL2017.

CL with Repetitions

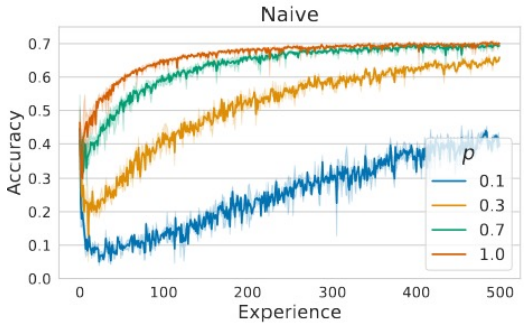
- Real world problems have repetitions of concepts and imbalanced concepts
- A natural form of replay
- Popular benchmark (the three scenarios) may provide a skewed perception of performance of a method
- ***CIR:*** Synthetic generator to simulate streams with repetitions and control the difficulty of the benchmark



H. Hemati et al. “Class-Incremental Learning with Repetition.” CoLLAs 2023. https://doi.org/10.48550/arXiv.2301.11396.

CL with Repetitions – Results

- Repetitions help, even with naive finetuning
- Models change less and less over time
  - *Stability may come for free at scale*
- **Frequency-Aware Replay**: With imbalanced streams correcting the imbalance with the replay buffer helps



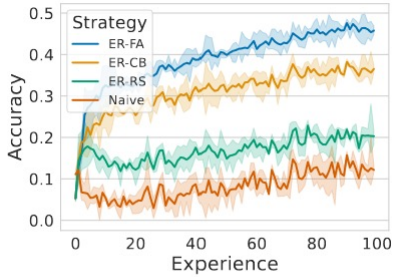


Figure 11: Accuracy of Infrequent Classes.

H. Hemati et al. “Class-Incremental Learning with Repetition.” CoLLAs 2023. https://doi.org/10.48550/arXiv.2301.11396.

CL Methods for Edge Devices

- **Online CL** methods learn from small mini-batches (with replay)
- **Rehersal-free** CL methods avoid storing past data to ensure privacy
- CL methods can exploit **randomized networks** and **pretrained models**

Efficiency – Replay with Class-Balanced Reservoir Sampling

- **Rehersal**: keeps a small buffer with the old data
- **Reservoir Sampling**: random sampling
- **Class-Balanced**: The buffer capacity is class-balanced
- **Advantages**: Efficient, suitable for Online CL

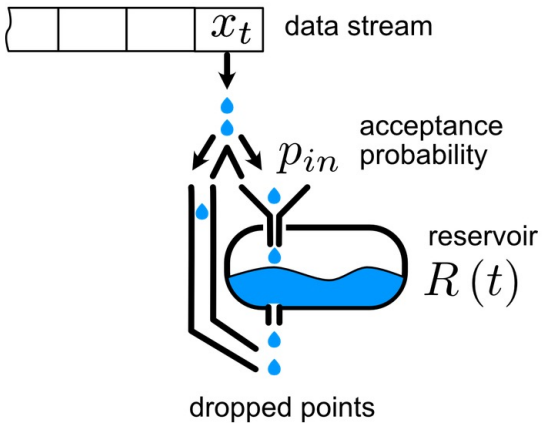
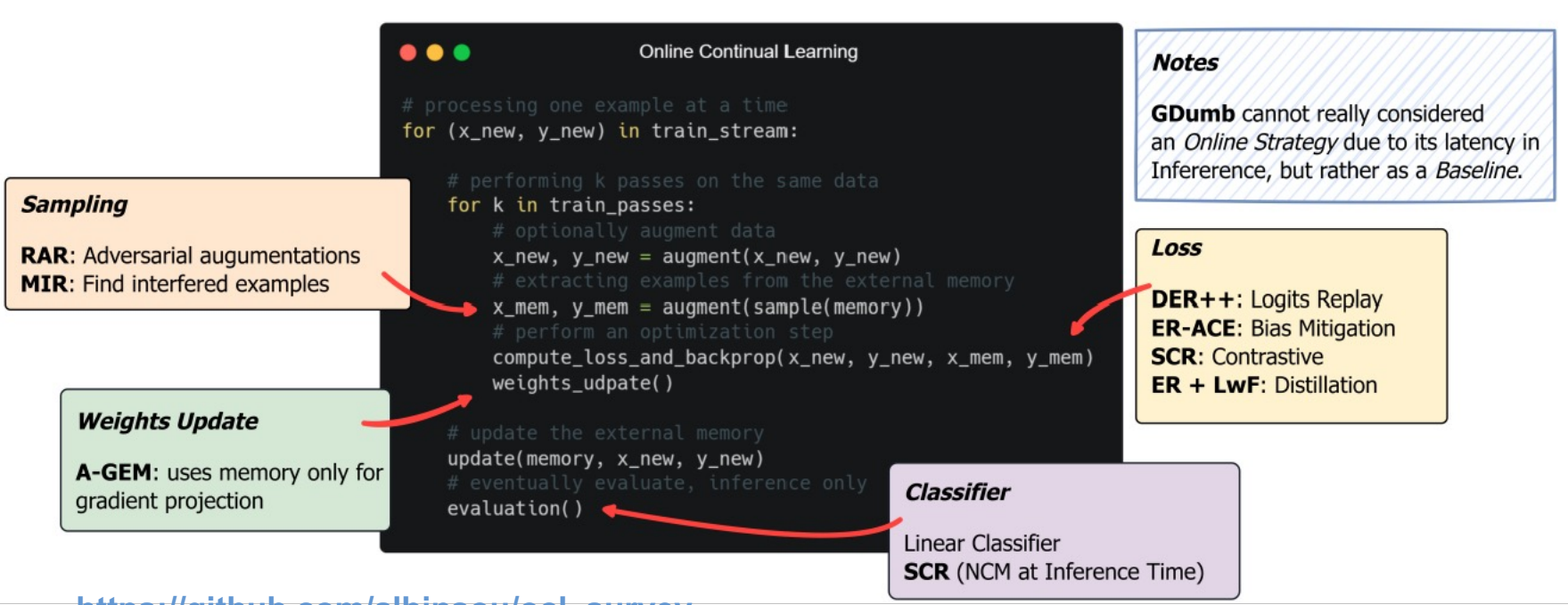


Image from https://towardsdatascience.com/reservoir-sampling-for-efficient-stream-processing-97f47f85c11b

A. Chaudhry et al. “Continual Learning with Tiny Episodic Memories,” 2019. http://arxiv.org/abs/1902.10486.

Online Continual Learning



https://github.com/albinsou/ocl_survey

A. Soutif—Cormerais et al. “A Comprehensive Empirical Evaluation on Online Continual Learning.” VCL@ICCV ‘23

Rehearsal-free CL

- **Hard problem**: if you can, always use replay
- **Pretraining** helps
- **Freezing** lower layers helps
- **Classifier bias**: in class-incremental learning, the classifier will be biased towards new classes

CL with Pretrained Models – Deep SLDA

- Fixed pretrained feature extractor
- Classifier trained via Streaming LDA
- No backpropagation: fast training

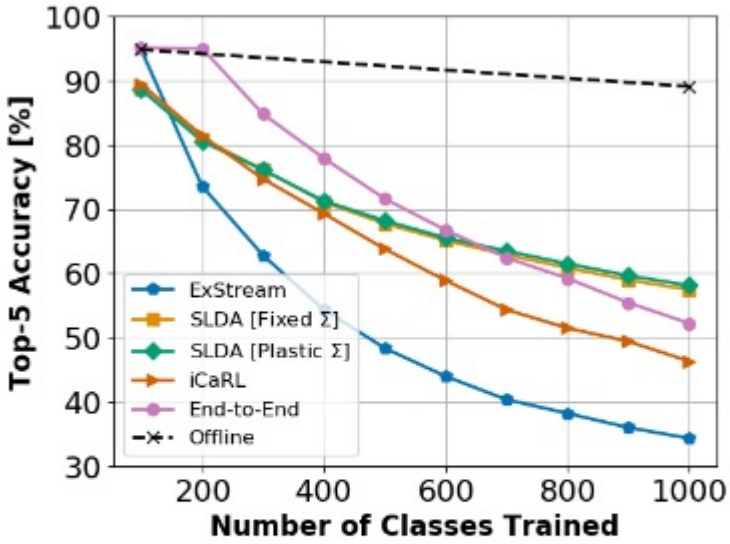


Figure 1: Learning curve for incremental ImageNet. Our Deep SLDA approach achieves the best final top-5 accuracy, while running over 100 times faster and using 1,000 times less memory than the iCaRL and End-to-End models.

Hayes, Tyler L., and Christopher Kanan. “Lifelong Machine Learning with Deep Streaming Linear Discriminant Analysis.” In *CVPRW* 2020. https://doi.org/10.1109/CVPRW50498.2020.00118.

CL with Pretrained Models – AR1

- Pretraining
- Architectural+Regularization Method
- Fix classifier bias with Copy-Weight-Reinit
- Regularization with Synaptic Intelligence

Algorithm 3 AR1

$cw = 0$
init $\bar{\Theta}$ random or from a pre-trained model (e.g. ImageNet)
$\Theta = 0$ (Θ are the optimal shared weights resulting from the last training, see Section 2.3)
$\hat{F} = 0$ ($\hat{F}$ is the weight importance matrix, see Section 2.3).
for each training batch B_i:
    expand output layer with s_i neurons for the new classes in B_i
    $tw = 0$ (for all neurons in the output layer)
    Train the model with SGD on the s_i classes of B_i by simultaneously:
        learn tw with no regularization
        learn $\bar{\Theta}$ subject to SI regularization according to $\hat{F}$ and Θ
    for each class j among the s_i classes in B_i:
        $cw[j] = tw[j] - avg(tw)$
    $\Theta = \bar{\Theta}$
    Update $\hat{F}$ according to trajectories computed on B_i (see eq. 10 and 11)
    Test the model by using $\bar{\theta}$ and cw

Maltoni, Davide, and Vincenzo Lomonaco. “Continuous Learning in Single-Incremental-Task Scenarios.” *Neural Networks* 116 (August 1, 2019): 56–73. https://doi.org/10.1016/j.neunet.2019.03.010.

CL for Time Series – CL with RNNs

- Sequence length increases forgetting
- Replay is still the best method
- Lack of a common benchmark for CL on time series

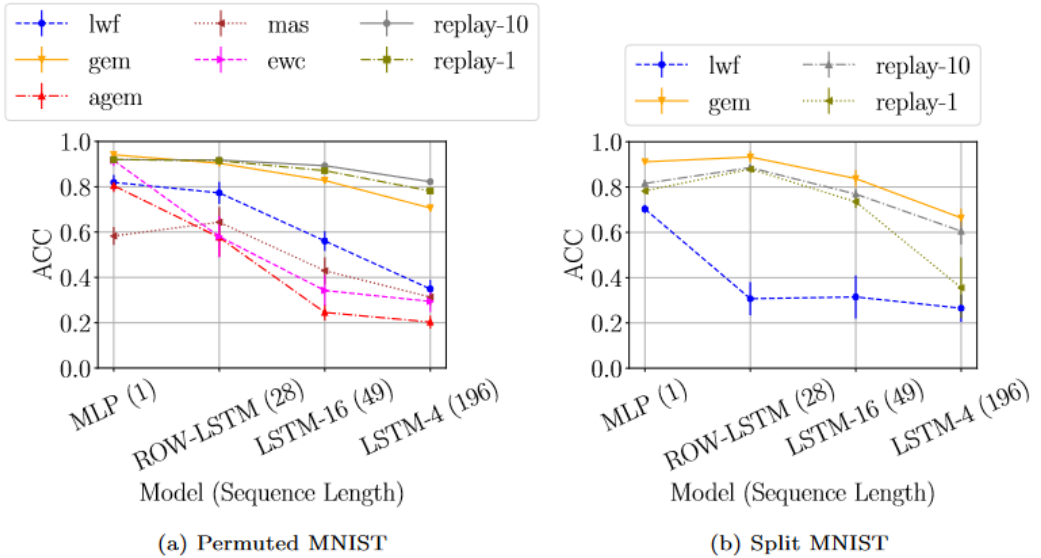


Figure 6: Average ACC on all steps for different sequence lengths and different CL strategies. Sequence length causes a decrease in performances among all strategies. Best viewed in color.

A. Cossu et al. “Continual Learning for Recurrent Neural Networks: An Empirical Evaluation.” *Neural Networks*. http://arxiv.org/abs/2103.07492.

CL for Time Series – CL with ESNs

- Alternative to pretraining for RNNs
- Methods that train only the classifier can be used (SLDA)
- Efficient
- rehersal-free
- Applicable for Online CL

SMNIST	LSTM†	ESN
EWC	$0.21_{\pm 0.02}$	$0.20_{\pm 0.00}$
LWF	$0.31_{\pm 0.07}$	$0.47_{\pm 0.07}$
REPLAY	$\mathbf{0.85_{\pm 0.03}}$	$0.74_{\pm 0.03}$
SLDA	—	$\mathbf{0.88_{\pm 0.01}}$
NAIVE	$0.20_{\pm 0.00}$	$0.20_{\pm 0.00}$
JOINT	$0.97_{\pm 0.00}$	$0.97_{\pm 0.01}$

SSC	LSTM†	ESN
EWC	$0.10_{\pm 0.00}$	$0.09_{\pm 0.02}$
LWF	$0.12_{\pm 0.01}$	$0.12_{\pm 0.02}$
REPLAY	$\mathbf{0.74_{\pm 0.07}}$	$0.36_{\pm 0.07}$
SLDA	—	$\mathbf{0.57_{\pm 0.03}}$
NAIVE	$0.10_{\pm 0.00}$	$0.10_{\pm 0.00}$
JOINT	$0.89_{\pm 0.02}$	$0.91_{\pm 0.02}$

Table 1: Mean ACC and standard deviation over 5 runs on SMNIST and SSC benchmarks. SLDA is applied only to ESN since it assumes a fixed feature extractor. SMNIST contains 5 experiences, while SSC contains 10 experiences. † results are taken from [3], except for replay which has been recomputed to guarantee the use of the same replay policy (200 patterns in memory).

A. Cossu et al. “Continual Learning with Echo State Networks,” ESANN 2021. http://arxiv.org/abs/2105.07674.

Towards Distributed Continual Learning

CL for Pervasive Learning

Personalized Models in a Distributed Setting

- Edge devices:
  - Learn a (single) local task
  - Limited computational resources
  - Small and private datasets
- Server:
  - Learns the multi-task global goal
  - Possibly large resources
  - Enables communication between devices

How do we learn personalized models? Can we achieve forward transfer and knowledge sharing without penalizing the local task accuracy?

Large Pretrained Models

- Properties:
  - Trained on a dedicated server/cluster
  - High computational resources
  - Large and diverse datasets
- Objectives:
  - Learning general knowledge
  - Provide forward transfer for downstream tasks

How do we learn large pretrained models with continual learning? How do we use them in CL?

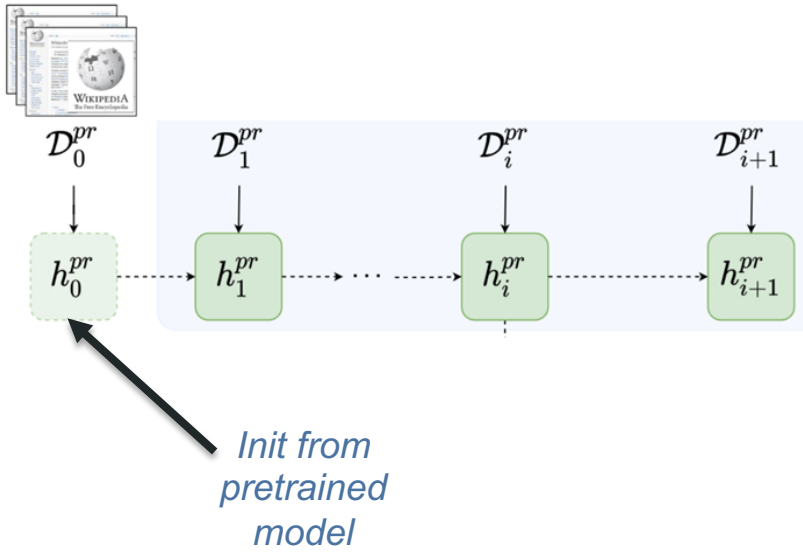
Usage of Pretrained Models in CL

PROBLEMS:

- How do we learn a global model good for all the local tasks
- Do we share a single multi-task model or do we finetune a model for each task, starting from the global model?

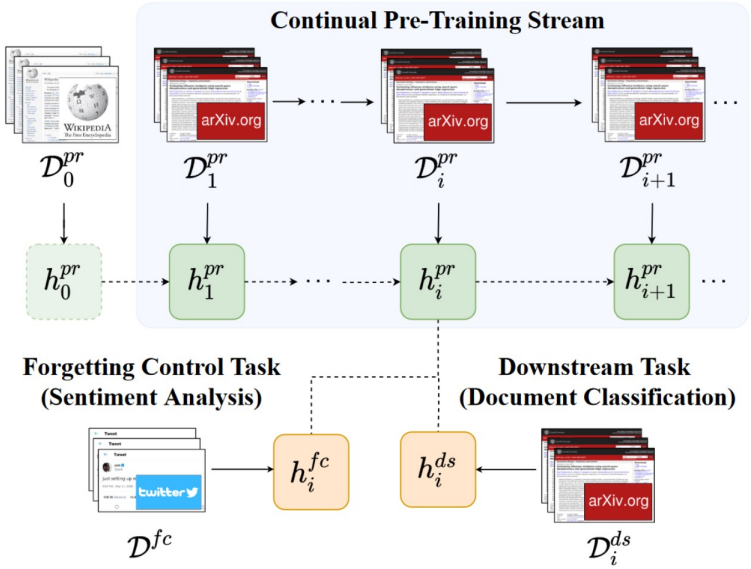
Distributed CL Solutions:

- *Continual Pretraining*: a large continually pretrained model is shared and finetuned separately on each downstream task
- *Federated CL*: centralized approach, a single multi-task model is shared among all the clients
- *Ex-Model/Distributed CL*: Edge devices learn local task and send their parameters to the server. The serve consolidates the models in a single multi-task model



Continual Pretraining

- **Continual Pretraining**: the large pretrained model is periodically updated via CL
- **Continual Finetuning**: whenever a new pretrained model becomes available, finetune the downstream model
  - Can be trained from scratch efficiently with linear probing.



Cossu, A., et al. "Continual Pre-Training Mitigates Forgetting in Language and Vision." arXiv preprint arXiv:2205.09357 (2022).

Continual Pretraining: Results

Evaluation on the Forgetting Control Task

Table 2: Accuracy on the entire dataset of `sentiment analysis` with RoBERTa model. Continual pre-training has been performed sequentially over each experience of `scientific abstracts`. Base refers to the model pre-trained on Wikipedia, while NT refers to the model with vocabulary expansion.

RoBERTa	**Accuracy**					**1-epoch Accuracy**				
Base	93.40					92.40				
Exp.	e1	e2	e3	e4	e5	e1	e2	e3	e4	e5
Pretr	93.40	93.15	93.35	93.20	92.90	92.40	91.80	92.30	91.85	92.20
Pretr. NT	93.75	93.70	93.75	93.60	94.10	91.75	91.15	92.00	92.30	92.45

← fast adaptation

Forgetting is limited even with naive finetuning.
Dynamic vocabulary expansion (NT) slightly improves the performance.

Federated Continual Learning

- Combines Federated and Continual Learning
- Example: **FedWelt**
  - **Client**'s model parameters: base + task-specific + weighted sum of other tasks
  - **Server** keeps all the parameters
  - **Forward transfer**: Clients learn which parameters from the other tasks are useful

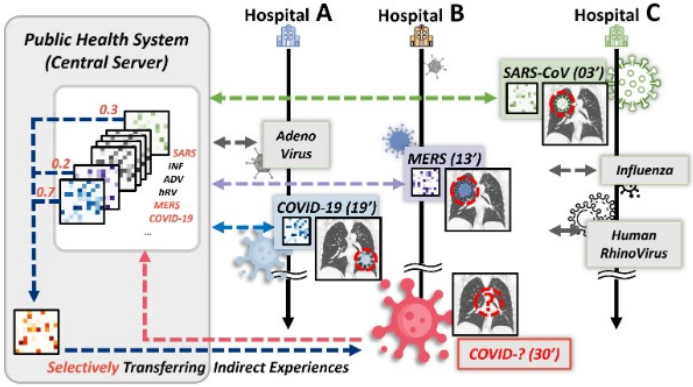
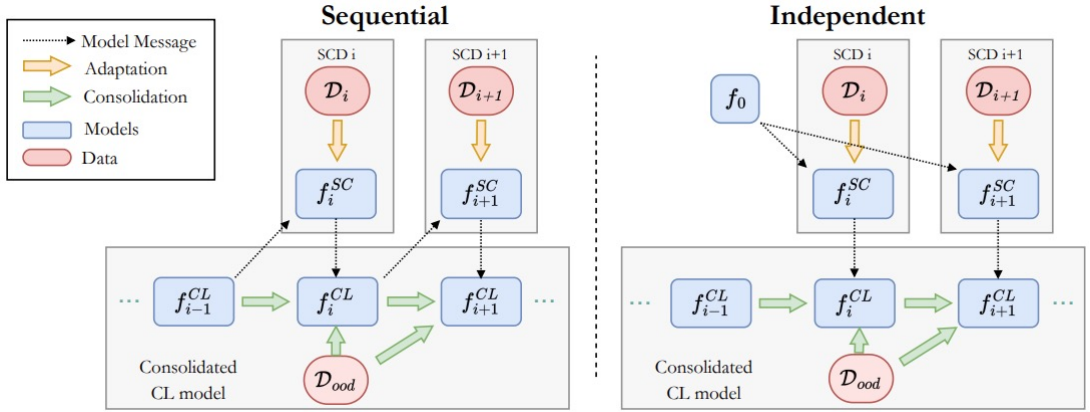


Figure 1. **Concept**. A continual learner at a hospital which learns on sequence of disease prediction tasks may want to utilize relevant task parameters from other hospitals. FCL allows such inter-client knowledge transfer via the communication of task-decomposed parameters.

J. Yoon et al. "Federated Continual Learning with Weighted Inter-Client Transfer." *ArXiv:2003.03196 [Cs, Stat]*, June 14, 2021. http://arxiv.org/abs/2003.03196.

Distributed Continual Learning

- **Distributed CL**:
  - local devices learn a single, starting from the global model as its initialization
  - Server learn a global multi-task model combining the local models sequentially
- **Adaptation**: learn a new local task (on local device)
- **Consolidation**: combine heterogeneous models from different devices (on server)

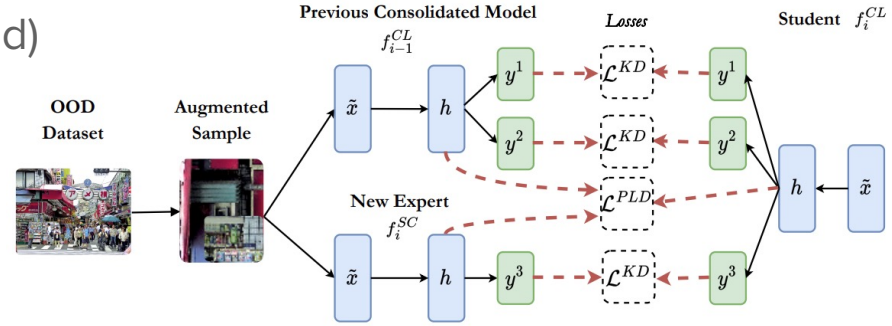


Ex-Model: Continual Learning From a Stream of Trained Models. *A. Carta et al*; CLVISION@CVPRW, 2022

"Projected Latent Distillation for Data-Agnostic Consolidation in Distributed Continual Learning." A. Carta et al. arXiv, March 28, 2023. https://doi.org/10.48550/arXiv.2303.15888.

Data-Agnostic Consolidation

- **Double Knowledge Distillation**
  - Teachers: Previous CL model and New model
  - On the output
  - On the latent activations (Projected)
  - Using o.o.d. data
- Task-incremental method
- **separate adaptation and consolidation is better than sequential finetuning** (i.e. what most CL methods are doing)



Ex-Model: Continual Learning From a Stream of Trained Models. *A. Carta et al*; CLVISION@CVPRW, 2022

"Projected Latent Distillation for Data-Agnostic Consolidation in Distributed Continual Learning." A. Carta et al. arXiv, March 28, 2023.

Avalanche

- CL library built on top of Pytorch
- Currently the most extensive collection of CL benchmarks and algorithms
- Used by the CL community for research, new benchmarks, challenges and courses

Website: avalanche.continualai.org/
CL-baselines: https://github.com/continualAI/continual-learning-baselines/
Avalanche-demo: https://github.com/AntonioCarta/avalanche-demo

powered by


```python
# model
model = SimpleMLP(num_classes=10)

# CL Benchmark Creation
perm_mnist = PermutedMNIST(n_experiences=3)
train_stream = perm_mnist.train_stream
test_stream = perm_mnist.test_stream

# Prepare for training & testing
optimizer = SGD(model.parameters(), lr=0.001, momentum=0.9)
criterion = CrossEntropyLoss()

# Continual learning strategy
cl_strategy = Naive(
    model, optimizer, criterion, train_mb_size=32, train_epochs=2,
    eval_mb_size=32, device=device)

# train and test loop over the stream of experiences
results = []
for train_exp in train_stream:
    cl_strategy.train(train_exp)
    results.append(cl_strategy.eval(test_stream))
```

Conclusion

Continual Learning provides efficient solutions to learn in constrained environments

- **Efficient solutions**: Replay with CBRS
- **Online CL**: fix classifier bias + use of pretrained models + freezing/regularization
- **A key enabler of distributed learning**:
  - Continual pretraining and finetuning
  - Federated CL
  - Ex-Model/Distributed CL