Ex-Model: Continual Learning from a Stream of Trained Models

Antonio Carta
University of Pisa
antonio.carta@di.unipi.it

Andrea Cossu
Scuola Normale Superiore
andrea.cossu@sns.it

Vincenzo Lomonaco
University of Pisa
vincenzo.lomonaco@unipi.it

Davide Bacciu
University of Pisa
davide.bacciu@unipi.it

Abstract

Learning continually from non-stationary data streams is a challenging research topic of growing popularity in the last few years. Being able to learn, adapt, and generalize continually in an efficient, effective, and scalable way is fundamental for a sustainable development of Artificial Intelligent systems. However, an agent-centric view of continual learning requires learning directly from raw data, which limits the interaction between independent agents, the efficiency, and the privacy of current approaches. Instead, we argue that continual learning systems should exploit the availability of compressed information in the form of trained models. In this paper, we introduce and formalize a new paradigm named "Ex-Model Continual Learning" *(ExML), where an agent learns from a sequence of previously trained models instead of raw data. We further contribute with three ex-model continual learning algorithms and an empirical setting comprising three datasets (MNIST, CIFAR-10 and CORe50), and eight scenarios, where the proposed algorithms are extensively tested. Finally, we highlight the peculiarities of the ex-model paradigm and we point out interesting future research directions.*

1. Introduction

Continual learning (CL) studies learning in dynamic, non-stationary environments [22, 24]. Recently, there has been significant progress in the development of continual learning algorithms able to efficiently learn deep hierarchical representations from a sequence of experiences or tasks with increasingly robust and effective solutions, even for challenging scenarios with high degrees of non-stationarity [16, 22].

Most of these solutions follow an agent-centric view of Artificial Intelligence, which tends to mimic the same operative constraints of biological learning systems [7, 16, 25]. Under this view, a continual learning agent directly interacts with the environment and learns from *raw data*. This framework is closer to neuroscience-grounded theories of learning and intelligence [7, 12], but it ignores the opportunities and challenges provided by the pervasive and distributed nature of the modern computing infrastructure:

1. *expert models*: continual learning should reuse knowledge from expert models, such as local personalized models or large pretrained models.
2. *distributed learning*: agents in a distributed environment should be able to learn independently and to share knowledge efficiently at the same time.
3. *sample efficiency*: learning from raw data may be inefficient due to noise and redundancy inherent to high-dimensional perceptual data.
4. *privacy*: sharing knowledge between agents must be limited by privacy constraints, and each agent should be allowed to set its privacy constraints.

Currently, (1) is partially addressed by initializing continual learning models using pretrained models [6, 21]. However, it is not possible to use multiple pretrained models or to exploit a pretrained model after the initialization phase. Recently, some works have partially addressed (2) by studying federated continual learning [9, 27, 31]. Unfortunately, this approach requires a tight integration between the devices, intensive communication and strong assumptions about the model's architecture and learning procedure. Point (3) is often ignored in the continual learning literature. Pretrained models can partially address (3) by providing a compressed form of knowledge, i.e. the model's parameters, that can be used to learn more efficiently. Lastly, (4) is explored in CL with settings such as data-free class-incremental scenarios [3, 26], where access to the previous data is forbidden. Again, this scenario assumes a single agent and access to the current data, making it difficult to share knowledge between multiple agents.

In this paper, we propose a novel framework based on an alternative and integrative approach of the four points

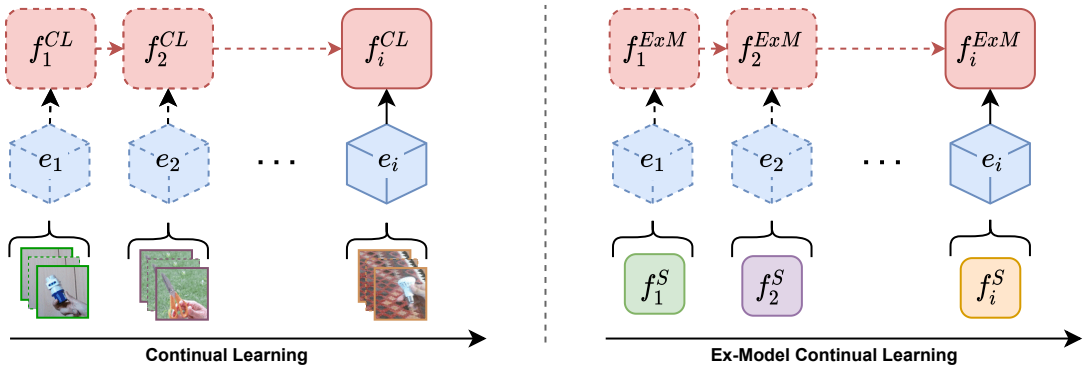

Figure 1. Classic Continual Learning scenario (left) compared to an Ex-model Continual Learning scenario (right). The CL model f_i^{ExM} is trained using the stream of expert models f_i^S, without access to the original data.

above, envisioning a more pervasive and distributed form of continual learning. Compressed knowledge and skills in the form of trained neural models (*"neural skills"*, for short) are generated and made available every day. So our motivating question is rather: *why not to learn directly from them instead?* Learning directly from trained models allows to efficiently share knowledge between domain experts (1), to train each expert independently (2) and efficiently (3). Additionally, each expert can determine its privacy level (4) by not sharing the model or training with differentially private algorithms [1].

Intuitively, learning from models resembles other forms of learning from compressed knowledge, such as when we learn from books or use the Internet instead of learning by trial and error. We argue that learning from compressed knowledge will become more and more important for the same reasons. Towards this vision, the original contributions of this paper can be summarized as follows:

1. We propose and formalize *Ex-Model Continual Learning*[1] (ExML) as a new CL paradigm designed to allow efficient and private sharing of compressed knowledge between independent agents (Section 2).

2. We propose a family of continual learning strategies, *Ex-Model Distillation (ED)*, based on data-free knowledge distillation (Section 3). In particular, we compare three possible instances of Ex-Model Distillation: two of them perform distillation by generating synthetic data, while the other relies on out-of-distribution data unrelated to the task solved by the expert (Section 4).

3. We assess the performance of Ex-Model Distillation strategies against five different baselines, three popular continual learning benchmarks (MNIST, CIFAR-10 and CORe50) and scenarios (Task-Incremental, Domain-Incremental and Class-Incremental) in order to highlight the general applicability of our solutions (Section 5). We release the code to easily instantiate the ExML scenario and to reproduce our experiments[2].

We believe that the introduction of the ExML paradigm, together with the design of the Ex-Model Distillation strategies and the experimental setup, will provide a robust starting point to study continual learning from models and its impacts on many downstream applications (Section 7).

2. Ex-Model Continual Learning

We begin by formalizing a *classic* continual learning scenario (Figure 1), where data arrives in a streaming fashion as a (possibly infinite) sequence of learning experiences $S = e^1, \dots, e^n$. We assume a supervised classification problem, where each experience e_i consists of a batch of samples $\mathcal{D}^i$, where each sample is a tuple $\langle x_k^i, y_k^i \rangle$ of input and target, respectively, and the labels y_k^i are from the set $\mathcal{Y}^i$, which is a subset of the entire universe of classes $\mathcal{Y}$. Notice that it is very easy to generalize the scenario to different CL problems. Usually $\mathcal{D}^i$ is split into a separate train set $\mathcal{D}_{train}^i$ and test set $\mathcal{D}_{test}^i$. A continual learning algorithm $\mathcal{A}^{CL}$ is a function with the following signature [16]:

$$\mathcal{A}^{CL} : \langle f_{i-1}^{CL}, \mathcal{D}_{train}^i, \mathcal{M}_{i-1}, t_i \rangle \rightarrow \langle f_i^{CL}, \mathcal{M}_i \rangle \quad (1)$$

where f_i^{CL} is the model learned after training on experience e^i, $\mathcal{M}_i$ a buffer of past knowledge, such as previous samples or activations, stored from the previous experiences and usually of fixed size. The term t_i is a task label which may be used to identify the correct data distribution. Most of the experiments in this paper assume the most challenging scenario of t_i being unavailable. Usually, CL algorithms are limited in the amount of resources that they can use, and

[1] The meaning of *ex* comes from latin, which can be roughly translated as "out of, from".

[2] https://github.com/AntonioCarta/ex_model_cl

they are designed to scale up to a large number of training experiences without increasing their computational cost over time. The objective of a CL algorithm is to minimize the loss $\mathcal{L}_S$ over the entire stream of data S:

$$\mathcal{L}_S(f_n^{CL}, n) = \frac{1}{\sum_{i=1}^{n} |\mathcal{D}_{test}^i|} \sum_{i=1}^{n} \mathcal{L}_{exp}(f_n^{CL}, \mathcal{D}_{test}^i) \quad (2)$$

$$\mathcal{L}_{exp}(f_n^{CL}, \mathcal{D}_{test}^i) = \sum_{j=1}^{|\mathcal{D}_{test}^i|} \mathcal{L}(f_n^{CL}(\boldsymbol{x}_j^i), y_j^i), \quad (3)$$

where the loss $\mathcal{L}(f_n^{CL}(\boldsymbol{x}), y)$ is computed on a single sample $\langle \boldsymbol{x}, y \rangle$, such as cross-entropy in classification problems.

Stream of Experts In an ExML scenario, there is *no direct access to a data stream* $\mathcal{D}_1, \dots, \mathcal{D}_n$. Instead, the stream consists of expert models $f_1^S, \dots, f_n^S$, where each expert is trained on some specific domain (Figure 1). As a consequence, an ex-model algorithm must learn only by extracting information from each expert. To keep the scenario as general as possible, we do not make any assumption about the models, such as their architecture or the specific hyperparameters used during training. We assume that the ExML algorithms have no control over how the experts have been trained. Each model f_i^S has been trained on a corresponding learning experience e^i to minimize $\mathcal{L}_{exp}(f_i^S, \mathcal{D}_{train}^i)$. We denote by f_i^S the learned function, θ_i^S its parameters, $f_i^S(\boldsymbol{x})$ the model's output, i.e. the logits, for a particular input sample $\boldsymbol{x}$, and $p_i^S(\boldsymbol{x})$ the output probabilities computed by applying the softmax function to the model's output.

ExML Scenario The objective of the ExML scenario is to continuously update a model f_i^{ExM} whenever a new expert f_i^S becomes available. Notice that the loss $\mathcal{L}_{exp}(f_i^{ExM}, \mathcal{D}_{train}^i)$ cannot be evaluated since we do not have access to the original data. Since the stream of models may be unbounded, training strategies must be scalable up to a large number of experts. Therefore, ex-model algorithms cannot keep in memory all the previous experts. As a result, there are two constraints in an ExML scenario: *lack of access to the original data and limited computational resources*.

Overall, an ExML algorithm $\mathcal{A}^{ExM}$ is a function with the following signature:

$$\mathcal{A}^{ExM} : \langle f_{i-1}^{ExM}, f_i^S, \mathcal{M}_{i-1}^{ex}, t_i \rangle \rightarrow \langle f_i^{ExM}, \mathcal{M}_i^{ex} \rangle, \quad (4)$$

where f_i^{ExM} is the current model, f_i^S the current expert from the stream, $\mathcal{M}_{i-1}^{ex}$ is a set of samples from out-of-distribution data or synthetically generated and currently available to the model (Section 4), and t_i the task label information. Again, notice that task labels are optional and they may not be available in many scenarios. The objective of ex-model algorithms is to minimize Eq. 2, the loss over the original (and unavailable) data stream.

3. Ex-Model Distillation

In this paper, we propose a family of algorithms, called Ex-model Distillation (ED) algorithms, to solve ex-model continual learning. The core idea behind our strategy is to exploit a cumulative buffer of synthetic or auxiliary data, generated from the expert model, to train the ex-model using knowledge distillation [10]. In this section, we describe how to perform the distillation and defer the data generation process to Section 4. The algorithm consists of two steps: buffer update and knowledge distillation. An overview of the algorithm is provided in Algorithm 1.

Buffer Update Let us assume to have access to a set of samples $\mathcal{M}_{i-1}^{ex}$ of fixed size N, where samples $\langle \boldsymbol{x}^{syn}, y^{syn} \rangle$ are obtained from the previous steps of the algorithm and they act as surrogate data in place of the original data from $e_1, \dots, e_{i-1}$. We use a data generating procedure $\mathcal{A}^{gen}$ to generate a new set of samples

$$\mathcal{D}_i^{ex} = \mathcal{A}^{gen}(f_i^S, \frac{N}{i}), \quad (5)$$

where $|\mathcal{D}_i^{ex}| = \lfloor \frac{N}{i} \rfloor$. $\mathcal{A}^{gen}$ generates synthetic data using the expert f_i^S. To obtain a new buffer $\mathcal{M}_i^{ex}$ of size N, we subsample $\tilde{\mathcal{M}}_{i-1}^{ex} = subsample(\mathcal{M}_{i-1}^{ex})$, such that $|\tilde{\mathcal{M}}_{i-1}^{ex}| = N - \frac{N}{i}$ and combine it with the new data to obtain the updated buffer $\mathcal{M}_i^{ex} = \tilde{\mathcal{M}}_{i-1}^{ex} \cup \mathcal{D}_i^{ex}$.

Knowledge Distillation Once we have updated the synthetic buffer, we can start the distillation process. Differently from knowledge distillation, we need to distill knowledge from two different models, the previous ex-model f_{i-1}^{ExM}, and the current expert from the stream f_i^S. Ex-model algorithms use $\mathcal{M}_i^{ex}$ to distill the knowledge from the current expert without forgetting previous knowledge. Each of these models is trained on a different (possibly overlapping) set of classes: $\mathcal{Y}^{prev} = \bigcup_{k=0}^{i-1} \mathcal{Y}^k$ for the ex-model, and $\mathcal{Y}^i$ for the expert. Given a sample $\langle \boldsymbol{x}^{syn}, y^{syn} \rangle$ and the output $\boldsymbol{y}^{curr} = f_i^{ExM}(\boldsymbol{x}^{syn})$ from the current ex-model, the target logits $\tilde{\boldsymbol{y}}$ are computed by combining the normalized logits of the previous ex-model and current expert:

$$\boldsymbol{y}^{ExM} = normalize(f_{i-1}^{ExM}(\boldsymbol{x}^{syn})) \quad (6)$$

$$\boldsymbol{y}^{S} = normalize(f_i^{S}(\boldsymbol{x}^{syn})) \quad (7)$$

$$(8)$$

$$\tilde{\boldsymbol{y}} = \begin{cases} \boldsymbol{y}^{ExM} & if\ y^{syn} \in \mathcal{Y}^{prev} \\ \boldsymbol{y}^{S} & if\ y^{syn} \in \mathcal{Y}^{i} \\ \frac{1}{2}(\boldsymbol{y}^{S} + \boldsymbol{y}^{ExM}) & if\ y^{syn} \in \mathcal{Y}^{prev} \wedge y^{syn} \in \mathcal{Y}^{i}. \end{cases} \quad (9)$$

Output normalization allows to combine the outputs independently from the difference in scale between the two models, which would create a bias if not removed. The resulting vector $\tilde{\boldsymbol{y}}$ is used as a target for the distillation by minimizing the Mean Squared Error (MSE) loss

$$\mathcal{L}_{MSE}(\boldsymbol{y}^{curr}, \tilde{\boldsymbol{y}}) = \|\boldsymbol{y}^{curr} - \tilde{\boldsymbol{y}}\|_2^2 . \tag{10}$$

Eq. 10 by itself is not sufficient to train a good model. The main limitation of the loss is that it is unable to distinguish whether a sample $\boldsymbol{x}^{syn}$ should be classified by the previous ex-model (i.e., it belongs to one of the previous experiences) or by the current expert (i.e. it belongs to the current experience) since the units of each model are treated separately. Therefore, the ex-model distillation loss $\mathcal{L}_{ED}$ combines the MSE with the crossentropy loss $\mathcal{L}_{CE}$:

$$\begin{aligned}\mathcal{L}_{ED}(\boldsymbol{y}^{curr}, \tilde{\boldsymbol{y}}, y^{syn}) =& \mathcal{L}_{MSE}(\boldsymbol{y}^{curr}, \tilde{\boldsymbol{y}}) \\ &+\lambda_{CE}\mathcal{L}_{CE}(\boldsymbol{y}^{curr}, y^{syn}).\end{aligned} \tag{11}$$

The loss in Eq. 11 is optimized by stochastic gradient descent for a fixed number of iterations, sampling randomly the buffer at each step to create a mini-batch.

Algorithm 1 Ex-Model Distillation

Require: Stream of pretrained experts S and a continually learned model f^{ExM}.

$\mathcal{M}_0^{ex} \leftarrow \{\}$ ▷ empty buffer
for f_i^S in S **do**
  $\mathcal{D}_i^{ex} \leftarrow \mathcal{A}^{gen}(f_i^S, \frac{N}{i})$
  $\tilde{\mathcal{M}}_{i-1}^{ex} \leftarrow subsample(\mathcal{M}_{i-1}^{ex})$
  $\mathcal{M}_i^{ex} \leftarrow \tilde{\mathcal{M}}_{i-1}^{ex} \cup \mathcal{D}_i^{ex}$
  for k in $1, \dots, n_{iter}$ **do** ▷ Knowledge Distillation
    $\langle \boldsymbol{x}^k, \boldsymbol{y}^k \rangle \leftarrow sample(\mathcal{M}_i^{ex})$
    $\boldsymbol{y}^{curr} \leftarrow f^{ExM}(\boldsymbol{x}^k)$
    $\tilde{y} \leftarrow get\_target(\boldsymbol{x}^k)$ ▷ Eq. 9
    $L \leftarrow \mathcal{L}_{ED}(\boldsymbol{y}^k, \tilde{\boldsymbol{y}}^k, y^k)$
    do SGD step on L
  end for
end for

4. Distillation Data

As discussed in Section 2, ex-model distillation needs an alternative source of data $\mathcal{M}_i^{ex}$ to distill the knowledge from the expert. Since the original data $\mathcal{D}_i$ is not available, we need an alternative source of samples. In this section, we show three methods that can be used to generate a synthetic dataset. Notice that in order to obtain a good performance it is not necessary that the synthetic data resembles the original data. Even highly distorted images or images from different domains may be useful to distill the knowledge from f_i^{ExM}. In fact, we will see in the experimental results that the synthetic data that we will use may be widely different from the original data.

Model Inversion. Model inversion [5] extracts samples using f_i^S by maximizing the output probabilities of the chosen class by gradient descent. Given a randomly initialized sample $\boldsymbol{x}^{syn}$, a target class y^{syn}, and an expert f_i^S, with model inversion we optimize $\boldsymbol{x}^{syn}$ by stochastic gradient descent by minimizing the crossentropy $\mathcal{L}_{CE}\left(f_i^S(\boldsymbol{x}^{syn})/\tau, y^{syn}\right)$, where τ is the softmax temperature. We can generate a batch of images for each class by using different random initializations. Since the computations for each sample are independent, they can be optimized in parallel using large mini-batches.

Data Impression. Data impression is a data extraction method proposed in [23] for the data-free offline training scenario. Data Impression exploits the classifier's weights of the expert $\boldsymbol{W} \in \mathbb{R}^{N_c \times N_h}$, where N_h is the number of hidden units and N_c the number of classes, to define a Dirichlet distribution used to sample probability targets. Data impression treats each row $\boldsymbol{w}_k$ of $\boldsymbol{W}$ as a template for class k, computing the matrix of pairwise similarities $C(k,j) = \frac{\boldsymbol{w}_k^\top \boldsymbol{w}_j}{\|\boldsymbol{w}_k\|, \|\boldsymbol{w}_j\|}$. The similarity coefficients are used to define a Dirichlet distribution $Dir(N_c, \boldsymbol{\alpha}^k)$ for each class such that $\boldsymbol{\alpha}^k = \beta \boldsymbol{c}_k$, $\boldsymbol{y}^{di} \sim Dir(N_c, \boldsymbol{\alpha}^k)$, where β is a temperature parameter and $\boldsymbol{c}_k$ the kth row of the similarity matrix. Targets $\boldsymbol{y}^{di}$ sampled from the resulting Dirichlet distribution are used to optimize a randomly initialized $\boldsymbol{x}^{syn}$ using a knowledge distillation loss $\mathcal{L}_{KD}(p_{f_i^{ExM}}(\boldsymbol{x}^{syn}), \boldsymbol{y}^{di}, \tau)$. We generate a different target for each sample using the Dirichlet distribution corresponding to the desired class. Since data impression provides a target for the entire output distribution instead of a single target class, which is needed for the ex-model distillation loss (Eq. 11), we set $y^{syn} = \arg\max \boldsymbol{y}^{di}$. The advantage of Data Impression compared to Model Inversion is that the Dirichlet distribution of the soft targets models the class similarities instead of ignoring them.

Auxiliary Data. The usage of auxiliary data is an alternative solution that does not require additional computation to generate synthetic samples. For example, for image classification tasks we may use large open datasets such as ImageNet [4] as a substitute for the original data. While the images may represent different classes, a large dataset of diverse images may be sufficient to distill knowledge from the expert models. This technique is also more efficient since it does not require a separate data generation phase. However, it is possible to use it only if a large open dataset is available, which may be true for image classification problems with natural images but more difficult in other domains, such as the medical domain, where data is scarcer. Since

data comes from a different domain than the original one, we do not have a target class corresponding to the original domain. Therefore, we set $\boldsymbol{y}^{syn} = \arg\max f_i^S(\boldsymbol{x}^{syn})$ for each sample $\boldsymbol{x}^{syn}$ in the auxiliary dataset.

4.1. Natural Image Priors

Synthetic data generation methods tend to generate images with unrealistic artifacts. The end-to-end optimization over the raw pixels generates images with high output probabilities in the target classes that do not resemble the original images. Natural image priors are regularization terms that encourage natural looking images.

Augmentations All the synthetic images are augmented with common image augmentations, both during the generation and during the ex-model distillation. Depending on the dataset, we use small displacements, rotations, and horizontal flip.

L^2 norm penalization We penalize the L^2 norm of each image $\mathcal{L}_{norm}(\boldsymbol{x}^{syn}) = \|\boldsymbol{x}^{syn}\|_2^2$. This regularization term is used to penalize high activations.

Blur In natural images, neighboring pixels are similar. To encourage this property, we penalize the term $\mathcal{L}_{blur}(\boldsymbol{x}^{syn}) = \|\boldsymbol{x}^{syn} - blur(\boldsymbol{x}^{syn})\|_2^2$, where $blur(\boldsymbol{x}^{syn})$ is the result of applying a gaussian blur with a 3×3 kernel to the raw image $\boldsymbol{x}^{syn}$.

Matching of batch normalization statistics Batch normalization layers provide useful information about the activations' statistic of the original images [29]. Ideally, synthetic images should have the same statistics. Given a model with k batch normalization layers with mean and variance μ_i, σ_i, and minibatch statistic for the synthetic images $\mu_i^{syn}, \sigma_i^{syn}$, we penalize the term $\mathcal{L}_{bns} = \sum_{i=0}^{k}(\|\mu_i - \mu_i^{syn}\|_2^2 + \|\sigma_i - \sigma_i^{syn}\|_2^2)$.

5. Experiments

The objective of the experimental evaluation is twofold: first, we evaluate ex-model scenarios under different conditions by proposing novel ExML benchmarks with varying levels of complexity. Second, we assess the performance of different ex-model distillation strategies by comparing different sources of synthetic data as defined in Section 4, along with a set of baselines.

For each scenario, we trained a separate expert model for each learning experience using the original data. For each configuration, we repeated the training phase 5 times to obtain 5 independent streams of experts that we used to compute the mean and standard deviation. For simplicity, we use the same architecture during the ex-model continual learning phase. All the experiments are implemented using Avalanche [19]. Source code for the proposed strategies, along with the experiments configuration, code to reproduce the experiments, and the pretrained experts is available online. Please refer to the repository and the additional material for extended details about the hyperparameters of the experiments.

Table 1. Summary of datasets and scenarios.

scenario	stream length	total classes	classes per step	model
Split MNIST (NC)	5	10	2	LeNet
Split CIFAR10 (NC)	10	10	2	ResNet18
CIFAR10-MT	10	10	2	ResNet18
CORe50-NC	9	50	10/5	MobileNet
CORe50-NI	9	50	50	MobileNet

Datasets and continual learning scenarios In each experiment, the stream of experts is trained on popular CL benchmarks. Most benchmarks come from the class-incremental literature [22], where each experience provides data for New Classes (NC), never seen before. We also ran experiments on the New Instances (NI) scenario [18], where each experience has the same classes with different instances (e.g. different backgrounds). Therefore, in the NI scenario expert models are trained on the same set of classes. Additionally, we show the results for joint training, i.e. the offline training where the data is seen all at once. In this setting, we do not have a stream of pretrained models. We used this scenario to evaluate the performance of the data extraction methods in the absence of continual learning.

We evaluated the proposed strategies on MNIST [15] with a stream of LeNet [14] models, using the Split MNIST (NC) scenario, with 5 experiences and 2 classes for each experience. We also provide the results for joint training to evaluate the degradation in performance from a simple data-free knowledge distillation to a more challenging Ex-Model CL scenario. For CIFAR10 [13], we used a ResNet18 [8] and we evaluated both the popular joint training scenario and the Split-CIFAR10 (NC) scenario [22]. The joint scenario uses all 10 classes at once, while the class incremental scenario uses 2 classes per experience. Furthermore, we evaluate Split-CIFAR10 in a multitask setting with a multi-head classifier (CIFAR10-MT).

Finally, we used CORe50 [18], a dataset specifically designed for continual learning. In the joint scenario we used all the 50 classes of the dataset, while in the class-incremental (NC) scenario we used 10 classes for the first experience and 5 for the subsequent ones. We also experimented with CORe50 in the NIC scenario. For all the configurations, we used a MobileNet [11] pretrained on ImageNet.

Ex-model strategies are evaluated in the single incremental task (SIT) setting with a single head: this means that the model does not have a task label to distinguish between the different experiences. This is the most challenging setting. A summary of the configuration for each scenario is shown

Table 2. Stream accuracy computed on the test set for MNIST and CIFAR10 continual learning scenarios. Ensemble methods' results are not shown for joint scenarios because ensembling is not necessary when there is a single model.

	Ex-model scenario	MNIST		CIFAR10		
		Joint	NC	Joint	NC	MT
Oracle	✗	93.71±0.28	99.42±0.19	87.37±1.11	96.58±0.86	96.58±0.86
Ensemble Avg.	✗	–	33.40±4.74	–	51.85±2.37	–
Min Entropy	✗	–	39.41±5.27	–	52.03±2.67	–
Param. Avg.	✓	–	20.11±0.97	–	10.00±0.00	51.85±2.37
Model Inversion ED	✓	93.09±1.43	43.23±3.00	64.55±3.25	17.40±3.96	61.71±7.52
Data Impression ED	✓	92.12±0.88	36.05±6.74	52.64±5.82	24.70±6.85	61.15±3.92
Aux. Data ED	✓	89.35±0.18	35.48±6.35	76.94±2.68	41.35±5.83	60.72±3.70

in Table 1.

Ex-model strategies We evaluated three Ex-model Distillation (ED) strategies: Model Inversion ED, Data Impression ED, Aux. Data ED. Each one uses a different source for synthetic data. All data extraction strategies use a memory buffer with a fixed size (5000 samples) to maintain the data extracted from the previous experiences while keeping the memory occupation reasonable for continual learning on a large stream of experts. We use Fashion MNIST [28] as auxiliary data for MNIST scenarios, and ImageNet for CIFAR10 and CORe50. Additionally, we show the results of four baselines. Except Param. Avg., all the baselines do not satisfy the constraints of the ex-model scenario since they store the full stream of models or require the original data.

Oracle This is an ensemble of the stream of experts. The ensemble uses a task label to determine the correct model to use. Notice that this ensemble achieves higher results than offline training since the task label makes the classification easier by excluding all the classes corresponding to different tasks.

Ensemble Avg. This is an ensemble of experts which computes the output as the average of the experts' outputs.

Min. Entropy This is a strategy that computes the output for each expert in the stream and uses as final output only the prediction with the minimum entropy (i.e. the one with less uncertainty). Given the output probabilities $\boldsymbol{p}^j$ computed by the j-th expert, the ensemble selects the outputs from expert j^*, where $j^* = \arg\min_j -\sum_i p_i^j \log p_i^j$. This ensemble computes all the output in parallel, similarly to the Oracle ensemble. However, the entropy is used to select the appropriate expert instead of the task label.

Param. Avg. This ensemble is a single model obtained by averaging the expert's parameters. This is the only baseline which respects the ex-model scenario constraints since it keeps a single model and it does not use the original data. Unlike ED, this strategy assumes that the experts' architectures are all equal.

Replay ED This is a strategy where the ex-model distillation is applied using the original data. Notice that this is different from a simple rehearsal strategy since it uses the loss of Eq. 11 instead of the crossentropy.

Ensemble baselines show the performance that can be obtained by combining the expert models without any training. The memory requirements of an ensemble grows linearly in the number of models, making these strategies not admissible for an ex-model scenario. Instead, Replay ED shows the performance of the ex-model distillation in the ideal setting where we have access to the original data.

5.1. Results

Table 2 shows the stream accuracy on the test set for MNIST and CIFAR10 scenarios. The average accuracy over time for the entire stream and single experiences is also available in the supplementary material. When we have access to the entire stream of expert models and task labels (Oracle) we obtain the upper bound performance. Notice that the accuracy of this strategy in the class incremental scenarios is even higher than the joint scenarios since the task labels provide additional information that restricts the number of possible classes for a given sample. Ensemble methods have a large drop in performance in the class incremental scenarios. Notice that we do not consider them to be proper ex-model strategies since they keep the entire stream of models. The only proper ex-model ensemble strategy is the Param. Avg., which obtains a random performance on CIFAR10. This is due to the large number of models that need to be averaged together, each one trained on different data. Ex-model distillation strategies have a very high performance in joint scenarios, showing that synthetic and auxiliary data is sufficient to perform the knowledge distillation. However, there is a large performance drop in the class incremental scenarios, except for CIFAR10-MT, which

Table 3. Stream accuracy computed on the test set for CORe50 continual learning scenarios. Ensemble methods' results are not shown for joint scenarios because ensembling is not necessary when there is a single model.

	Ex-model scenario	CORe50		
		Joint	NC	NI
Oracle	✗	85.73±0.29	96.04±1.08	–
Ensemble Avg.	✗	–	26.30±1.38	69.92±0.70
Min. Entropy	✗	–	42.41±0.96	61.36±1.86
Param. Avg.	✓	–	2.00±0.00	2.00±0.00
Model Inversion ED	✓	50.06±2.76	33.1±1.93	44.38±4.93
Data Impression ED	✓	52.91±2.09	17.57±3.57	43.26±2.36
Aux. Data ED	✓	81.82±0.29	34.87±1.16	44.51±2.91

(a) Original data. (b) Joint MNIST, Model Inversion (c) Joint MNIST, Data Impression (d) Split MNIST, Model Inversion (e) Split MNIST, Data Impression

Figure 2. Original data and generated samples for Joint MNIST and Split MNIST.

is multi-task and therefore easier to learn. Table 3 shows the stream accuracy on the test set for CORe50 scenarios. CORe50 is more challenging than MNIST and CIFAR10 due to the higher resolution images (we use 128x128 images in our experiments). However, since we used a pretrained MobileNet, all the expert models start from a common initialization with a rich feature extractor. The initialization helps to learn and mitigate the interference between the experts in the NC and NI scenarios. Notice that in the NI scenario it is not possible to evaluate the Oracle baseline since CORe50-NI provides a single test set that cannot be split, unlike the NC scenario where we split by classes.

Buffer samples Figure 2 shows the samples generated by model inversion and data impression on joint MNIST and Split MNIST. In both settings, the images have been trained until the desired class was predicted with probability > 0.99. Both methods generate visually plausible digits in the joint scenario, wich resemble multiple digits superimposed over each other. Instead, in the class incremental scenarios, despite the high confidence of the model, the images are far from being realistic. This result may partially explain the different results of joint and continual learning scenarios. In joint scenarios, the single expert must be able to distinguish between all the possible classes. Instead, in continual scenarios, the experts will overfit their small subset of data, and will incorrectly classify out-of-domain classes with high confidence. As a result, the generated images will not be realistic because the model did not learn to extract the features that would help to classify the images in the joint domain.

For more samples please check the additional material. While the difference between the joint and NC scenarios is less striking in some CIFAR10 and CORe50 configurations, the general conclusions are the same and it can be easily noticed that images generated in the NC scenario are qualitatively worse.

Buffer size Figure 3 shows the test stream accuracy on CIFAR10-MT for increasing buffer sizes. The minimum size is 10, corresponding to a single sample per class. Notice that the blue line showing ex-model distillation using a subset of the original data increases with larger buffers. Instead, other methods show negligible differences between the minimum and maximum buffer size. This results hints that the major limitation of current data extraction techniques is the scaling to larger buffers. There may be several reasons for the lack of scaling in accuracy. For example, the diversity between generated images of the same class may be insufficient, which renders large buffers useless (see the additional material for some samples).

Buffer strategy Figure 3 shows the performance of ex-model distillation techniques against ex-model distillation on the real data (Replay ED). We notice that there is a large gap between Replay ED and proper ex-model distillation strategies. Overall, we did not find a large performance difference between Model Inversion and Data Impression. Instead, we see techniques based on data generation perform better on MNIST, while auxiliary data is better on some CIFAR10 and CORe50 scenarios. We argue that this is a

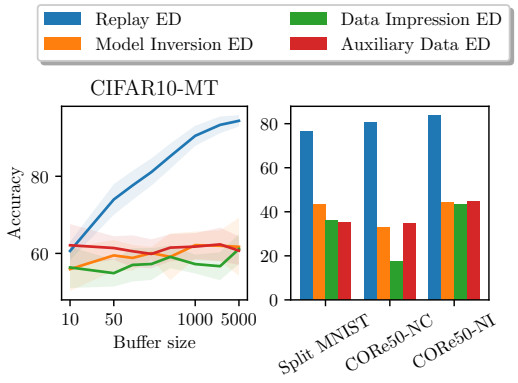

Figure 3. Test stream accuracy on CIFAR10-MT for increasing buffer sizes (left). Results are shown on a semi-logarithmic scale. Comparison between different strategies, including ex-model distillation using a small buffer of the original data (Replay ED, on the right).

consequence of the similarity between the original data and the auxiliary data. MNIST and Fashion MNIST are very different, except for the fact that they both use greyscale images, since the domains are completely separate. Instead, CIFAR10 and CORe50 are much closer to ImageNet since they both contains natural images, albeit with different resolutions and classes. Furthermore, it appears synthetic data techniques perform much better on multitask scenarios.

6. Related Works

The objective of continual learning is to continuously adapt the model without forgetting the previous knowledge [16]. Recently, there has been an increasing interest towards data-free settings, where the previous data is not available. Under this constraint, the most popular scenario is the data-free class-incremental learning (DF-CIL), where new classes appear over time. Notice that, despite the apparent similarity, DF-CIL is a very different scenario from ex-model. First, in DF-CIL the current data is available, making it possible to exploit a subset of the real data. Furthermore, in DF-CIL the model is trained sequentially, while in ex-model scenarios the experts are trained independently.

Learning without forgetting (LwF) [17] is a continual learning strategy that mitigates catastrophic forgetting via knowledge distillation of the old model's logits computed on the new data. More recently, several proposals have adapted knowledge distillation to DF-CIL scenarios. [32] exploits publicly available training data. [3, 26] train generative models to extract synthetic samples.

Data-free knowledge distillation methods apply several techniques to generate representative and diverse samples. Image priors help to guide the optimization process towards natural looking images during the model inversion [5]. [20] proposes the use of norm penalization to penalize high activations and total variation to penalize differences between small shifts. [29] proposes to match batch normalization statistics between the real and generated data. [23] generates targets logits according to a Dirichlet distribution instead of hard targets in order to capture inter-class similarities, while [30] trains generative networks instead of the samples directly. Notice that realistic images are not strictly necessary to perform knowledge distillation. [2] shows that knowedge distillation can be modeled as function matching. They show that aggressive augmentations combined with long training regimes help the knowledge distillation.

7. Discussion and Conclusion

In this paper we introduced Ex-Model Continual Learning, a novel scenario to continuously train a model from a stream of pretrained experts, without assuming any access to training data. We proposed a family of continual learning strategies, called Ex-Model Distillation, able to transfer knowledge from the experts to the Ex-Model, trained continuously. We validated the ability of three ED strategies to learn in our novel scenario against three different continual learning benchmarks. ExML exploits the growing number of pretrained models currently available for many different applications (object detection, language modelling...), without making any assumptions on the model architecture or the training modalities. ExML would benefit from an organized categorization of the existing pretrained models and of the type of knowledge they acquired during training. Such *neural skills catalogue* would make it easier to decide, when possible, which expert to select in order to best incorporate the required knowledge.

ExML is related to modern distributed learning paradigms, where different models are trained independently and their knowledge is then aggregated into a centralized architecture. Unlike federated learning, where each agent is constantly communicating with a centralized server, in ExML each agent is independent and the communication between agents is limited. Moreover, as privacy-aware settings are gaining relevance within the machine learning community, the need to learn in data-free environment will become mandatory for many applications. Medical environments, for example, are often subjected to strong privacy constraints, where it might not be possible to transfer data collected from patients to other devices. Ultimately, ExML constitutes a novel paradigm which does not supersede the available continual learning scenarios, but instead it stands as promising alternative to deliver continual learning capabilities to otherwise inaccessible real-world environments.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 308–318, New York, NY, USA, Oct. 2016. Association for Computing Machinery. 2

[2] Lucas Beyer, Xiaohua Zhai, Amélie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. Knowledge distillation: A good teacher is patient and consistent. *arXiv:2106.05237 [cs]*, June 2021. 8

[3] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Dual-Teacher Class-Incremental Learning With Data-Free Generative Replay. *arXiv:2106.09835 [cs]*, June 2021. 1, 8

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009. 4

[5] Alexey Dosovitskiy and Thomas Brox. Inverting Visual Representations with Convolutional Networks. *arXiv:1506.02753 [cs]*, Apr. 2016. 4, 8

[6] Tyler L. Hayes and Christopher Kanan. Lifelong Machine Learning with Deep Streaming Linear Discriminant Analysis. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 887–896, Seattle, WA, USA, June 2020. IEEE. 1

[7] Tyler L. Hayes, Giri P. Krishnan, Maxim Bazhenov, Hava T. Siegelmann, Terrence J. Sejnowski, and Christopher Kanan. Replay in Deep Learning: Current Approaches and Missing Biological Elements. *arXiv:2104.04132 [cs, q-bio]*, Apr. 2021. 1

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 5

[9] Sean M. Hendryx, Dharma Raj KC, Bradley Walls, and Clayton T. Morrison. Federated Reconnaissance: Efficient, Distributed, Class-Incremental Learning. *arXiv:2109.00150 [cs]*, Aug. 2021. 1

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network. pages 1–9, 2015. 3

[11] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv:1704.04861 [cs]*, Apr. 2017. 5

[12] Ronald Kemker and Christopher Kanan. FearNet: Brain-Inspired Model for Incremental Learning. In *International Conference on Learning Representations*, Feb. 2018. 1

[13] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. page 60. 5

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. 5

[15] Yann LeCun. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. 5

[16] Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual Learning for Robotics: Definition, Framework, Learning Strategies, Opportunities and Challenges. *arXiv:1907.00182 [cs]*, Nov. 2019. 1, 2, 8

[17] Zhizhong Li and Derek Hoiem. Learning without Forgetting. *arXiv:1606.09282 [cs, stat]*, Feb. 2017. 8

[18] Vincenzo Lomonaco and Davide Maltoni. CORe50: A new dataset and benchmark for continuous object recognition. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 17–26. PMLR, Nov. 2017. 5

[19] Vincenzo Lomonaco, Lorenzo Pellegrini, Andrea Cossu, Antonio Carta, Gabriele Graffieti, Tyler L. Hayes, Matthias De Lange, Marc Masana, Jary Pomponi, Gido M. van de Ven, Martin Mundt, Qi She, Keiland Cooper, Jeremy Forest, Eden Belouadah, Simone Calderara, German I. Parisi, Fabio Cuzzolin, Andreas S. Tolias, Simone Scardapane, Luca Antiga, Subutai Ahmad, Adrian Popescu, Christopher Kanan, Joost van de Weijer, Tinne Tuytelaars, Davide Bacciu, and Davide Maltoni. Avalanche: An End-to-End Library for Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3600–3610, 2021. 5

[20] Aravindh Mahendran and Andrea Vedaldi. Understanding Deep Image Representations by Inverting Them. *arXiv:1412.0035 [cs]*, Nov. 2014. 8

[21] Davide Maltoni and Vincenzo Lomonaco. Continuous learning in single-incremental-task scenarios. *Neural Networks*, 116:56–73, Aug. 2019. 1

[22] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost van de Weijer. Class-incremental learning: Survey and performance evaluation. *arXiv:2010.15277 [cs]*, Oct. 2020. 1, 5

[23] Gaurav Kumar Nayak, Konda Reddy Mopuri, Vaisakh Shaj, R. Venkatesh Babu, and Anirban Chakraborty. Zero-Shot Knowledge Distillation in Deep Networks. *arXiv:1905.08114 [cs, stat]*, May 2019. 4, 8

[24] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, Feb. 2019. 1

[25] Mark B. Ring. CHILD: A First Step Towards Continual Learning. *Machine Learning*, 28(1):77–104, July 1997. 1

[26] James Smith, Yen-Chang Hsu, Jonathan Balloch, Yilin Shen, Hongxia Jin, and Zsolt Kira. Always Be Dreaming: A New Approach for Data-Free Class-Incremental Learning. *arXiv:2106.09701 [cs]*, June 2021. 1, 8

[27] Anastasiia Usmanova, François Portet, Philippe Lalanda, and German Vega. A distillation-based approach integrating continual learning and federated learning for pervasive services. *arXiv:2109.04197 [cs]*, Sept. 2021. 1

[28] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv:1708.07747 [cs, stat]*, Sept. 2017. 6

[29] Hongxu Yin, Pavlo Molchanov, Zhizhong Li, Jose M. Alvarez, Arun Mallya, Derek Hoiem, Niraj K. Jha, and Jan Kautz. Dreaming to Distill: Data-free Knowledge Transfer via DeepInversion. *arXiv:1912.08795 [cs, stat]*, June 2020. 5, 8

[30] Jaemin Yoo, Minyong Cho, Taebum Kim, and U Kang. Knowledge Extraction with No Observable Data. page 10. 8

[31] Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated Continual Learning with Weighted Inter-client Transfer. *arXiv:2003.03196 [cs, stat]*, June 2021. 1

[32] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C.-C. Jay Kuo. Class-incremental Learning via Deep Model Consolidation. *arXiv:1903.07864 [cs]*, Jan. 2020. 8