



Explainable lifelong streaming learning based on Episode-to-Concept formation

Chu Kiong Loo ^a,^{*} Yi Chen ^a, Wei Shiung Liew ^a, Stefan Wermter ^b

^a Faculty of Computer Science and Information Technology, Universiti Malaya, Jalan Profesor Diraja Ungku Aziz, 50603 Kuala Lumpur, Malaysia

^b Knowledge Technology, Department of Informatics, University of Hamburg, 22527 Hamburg, Germany

ARTICLE INFO

Keywords:

Explainable AI
Interpretability
Prototype-based models
Lifelong learning
Continual learning
Streaming learning
Transfer learning
Knowledge engineering
Self-organizing neural networks

ABSTRACT

Real-time on-device continuous learning is increasingly used in critical embedded applications, such as medical monitoring, where user trust and transparency are equally as important as accuracy. Current streaming models are often “black boxes,” making it difficult to determine how and why a model arrives at a particular decision. To address this, we propose the Explainable Lifelong Learning (ExLL) model, a neuroscience classifier inspired by the Episodic-to-Concept (EpCon) theoretical model of hippocampal function in concept learning. ExLL is capable of: (1) efficient learning from scarce, streaming data in a single pass; (2) a self-organizing prototype-based architecture dynamically expands as needed, clustering the streaming data into separable groups by similarity and preserves data against catastrophic forgetting; (3) an interpretable architecture that can be represented as IF-THEN rules for inference auditing; and (4) combines local prototype inference with global class-level context using a pairwise decision fusion process to maximize performance. While the design of the architecture incurs a higher memory cost compared to simpler centroid-based models, this is a calculated trade-off to provide explainability functionality. We compare ExLL against selected established lightweight streaming algorithms to evaluate the performance in several learning scenarios with the benchmark datasets OpenLORIS, Places365, and F-SIOL-310. Our findings showed that ExLL outperformed baselines such as SLDA and Replay, in classification accuracy across diverse continual learning scenarios, proving that state-of-the-art streaming performance can be successfully balanced with granular, human-readable explainability.

1. Introduction

In most real-world applications, data arrives continuously in real-time and is often non-repeating unless it is memorized. From this phenomenon, two paradigms are coined: continual learning and streaming learning. Continual learning, also known as lifelong learning [1], refers to the ability to acquire knowledge continuously over a long period of time while retaining previously learned knowledge. Streaming learning [2], on the other hand, is the ability to acquire knowledge from sequential and continuously arriving data streams. The former encompasses machine learning techniques to adapt and reconcile old and new knowledge while minimizing loss of information, and the latter prioritizes quick and efficient knowledge acquisition from high-speed data streams. When developing machine learning applications for use in embedded systems such as portable medical devices, service robots, and autonomous vehicles, models must be able to operate under strict hardware constraints without compromising user safety [3]. Traditional embedded learning methods prioritize computational and

memory efficiency. However, in critical embedded applications, accuracy and efficiency alone are insufficient; the system’s decision-making capability must be transparent to the user to engender trust [4,5]. While state-of-the-art continual learning architectures, such as Streaming Linear Discriminant Analysis (SLDA) [6] or Nearest Class Mean (NCM) [7], have highly-efficient and compressed memory architecture, they become “black boxes” and lack the ability to explain their decisions and to diagnose potentially critical misclassifications [8]. In addition, while conventional learning strategies minimize risk by assuming independent and identically-distributed (IID) samples, continual learning often suffers from catastrophic forgetting. Algorithms like SLDA mitigate inter-class forgetting by maintaining independent class statistics. However, if a singular class contains diverse object types, the averaged class centroid will experience significant intra-class forgetting.

The increasing demand and usage of AIs by laymen highlights the need for explainable algorithms that can show their thought processes step-by-step from input to final decision. Explainable architectures

^{*} Corresponding author.

E-mail address: ckloo.um@um.edu.my (C.K. Loo).

such as eXplainable Deep Neural Networks (xDNN) by Angelov and Soares [9] offer prototype-based rule generation, but they have not been adapted for single-pass streaming conditions. For safety-critical edge environments such as wearable medical devices, it is necessary to trade a calculated increase in memory usage for transparent, human-readable explainability and state-of-the-art accuracy. To achieve this, streaming architectures must learn in a single pass while preserving the data complexity to prevent intra-class forgetting, while also able to explain model decisions at the intermediate and final stages of the decision-making process.

In this study, we investigate explainable continual and streaming learning methods for embedded devices. The paper presents the following research contributions in this field:

1. **Intra-Class Forgetting Mitigation.** We propose the Explainable Lifelong Learning (ExLL) model, inspired by the Episode-to-Concept (EpCon) theoretical model of hippocampal function. Unlike models that averaged all intra-class samples into a single class mean, ExLL uses a prototype mechanism that dynamically grows new prototypes to capture diverse intra-class distributions, to prevent forgetting unique class variations.
2. **Pairwise Decision Fusion.** We introduce a collective inference strategy to enhance classification accuracy by combining inferences from two complementary levels: local inferences at the prototype level (i.e. “Among the examples in class A, which example is the closest match to the input?”) and global inferences at the MegaCloud (category) level (i.e. “Among all the classes, which is the closest match to the input?”).
3. **Antecedent Explainability.** We formulate an explainability framework that uses prototypes to justify model predictions. By providing explicit visual “Hits” (support), “Near Hits” (ambiguity), and “Near Misses” (decision boundaries), the model provides a visualization method allowing users to view and diagnose the model’s decision process.
4. **Benchmark Evaluation of the Efficiency-Trust Trade-off.** We conducted a series of benchmark tests using established datasets (OpenLORIS, F-SIOL-310, Places-365) to observe how ExLL balances high-performance accuracy with the memory costs of explainability, relative to established lightweight continual learning models.

This paper is organized as follows. Related Work covers a number of prior research in the domains of Streaming Learning, On-Device Learning, and Explainable Prototype-Based Learning Models. Then, Methodology goes into detail on how the ExLL is implemented, followed by the Explainability Framework to outline the explanation generation procedure. The Experiment Setup section describes how the proposed method is tested using benchmark datasets OpenLORIS, Places-365, and F-SIOL-310, similar to the experiments performed in Hayes and Kanan [10] to compare the algorithm performance for their ability to learn over time and their processing and memory requirements. Our experiment findings are then discussed in the Results and Discussion sections followed by our concluding statements and future work.

2. Related work

2.1. Streaming learning

In streaming learning scenarios, machine learning algorithms are required to learn from a continuous stream of non-repeating training samples in a single pass. The algorithms must also be capable of being evaluated at any point during the stream and prior training samples are not stored for retraining. Early prototype-based classifiers, such as ARTMAPs [11] were developed to learn from non-stationary data, though their performance was significantly hampered by the

presentation order of training data. Various optimizations were later introduced [12–14] but they were often computationally intensive and therefore unsuitable for real-time applications.

More recently, algorithms that model statistical distributions online have become commonly used for efficient streaming learning. Streaming Linear Discriminant Analysis (SLDA) proposed by Pang et al. [6] extends the conventional Linear Discriminant Analysis (LDA) architecture to support incremental learning by storing a running mean for each unique class and a shared covariance matrix. The Deep-SLDA architecture proposed by Hayes and Kanan [15] paired SLDA with a convolutional neural network (CNN) acting as a feature extractor for high-dimensional inputs such as images. Other fundamental streaming adaptations include Nearest Class Mean (NCM) [7] and Streaming One-vs-Rest (SOvR) [10] classifiers. While earlier algorithmic adaptations like Online Perceptron [16] or Streaming Gaussian Naive Bayes [17] have been historically tested in these environments, modern benchmarks consider NCM and SLDA as the performance upper bounds for distribution-based streaming classifiers [10].

2.2. Continual embedded/on-device learning

Although streaming learning algorithms have been developed to reduce catastrophic forgetting, they do not meet the requirements for embedded applications. Disqualifying criteria include the high storage and computation requirements of batch learning techniques, and the reliance on task labels during inference [15,18]. Another requirement for continual embedded learning is the ability to generalize from a very small number of training samples. Algorithms with this capability are commonly known as “low-shot” continual learning algorithms [19–22]. Lastly, Ahrens et al. [23] suggest that continual learning needs to strike a balance between sufficiently specializing in each task and building a generalized knowledge for transfer.

Recently, the broader continual learning field has shifted heavily towards architectures utilizing Vision Transformers (ViTs) [24] and prompt-based continual learning mechanisms such as L2P [25] and DualPrompt [26]. However, these large-scale transformer models are incompatible with the strict memory, energy, and computational limits for edge devices like mobile phones or portable medical devices. Consequently, lightweight CNNs such as MobileNet [27], SqueezeNet [28], and EfficientNet [29] remain the standard backbones for embedded environments, balancing the accuracy of classification with the speed of processing and low memory footprint.

A comprehensive study was performed to compare several continual learning algorithms and CNNs as feature extractors in multiple scenarios mirroring the work by Hayes and Kanan [10]. The models were tested on their robustness to scale, on imbalanced class distribution, and on temporally correlated video streams. The models were then evaluated on the basis of classification accuracy, number of parameters, and experiment runtime. We use the same experiment protocols to evaluate the performance of our proposed continual learning model against other algorithms.

2.3. Explainable prototype-based learning models

“Antecedent Explainability” in context of learning models refer to the ability to explain the cause-and-effect, or how the model arrives at a particular decision when given a specific input. Decision trees for example have intrinsic antecedent explainability from its underlying architecture, allowing for human-interpretable “IF-THEN-ELSE” explanations. In streaming environments, several antecedent methods have been introduced such as Belief Rule Bases [30,31] and Dynamic Model Trees [32]. While these methods qualify as antecedent explainability methods, they lose significant classification accuracy when used for more complex and multi-dimensional data streams, e.g. image and video streaming.

Convolutional neural networks (CNNs) are uniquely suitable to process high-dimensional inputs. The architecture of CNNs is designed to maximize predictive accuracy through a series of convolutional steps, making them inherently “black box” models due to how difficult it is to explain how they arrive at a specific classification decision. CNNs are typically interpreted post hoc: the model’s decisions are obtained first, and tools like Gradient-weighted CAMs (Grad-CAMs) [33,34] are used to highlight discriminative features. However, these techniques are approximations as opposed to in-depth explanations of the cause-and-effect reasoning. In critical embedded systems where trust is needed, the models require antecedent explainability that can adequately justify the decision boundary of the inference results.

Prototype-based classifiers such as ARTMAPs by Carpenter et al. [11] and self-organizing networks by [35] group training samples according to their proximity in the feature space [36]. The xDNN proposed by Angelov and Soares [9] is a prototype-based classifier with the ability to generate human-interpretable explanations for deep neural networks, where the self-organizing probability distributions based on local densities are used to generate linguistic IF-THEN rules [37]. In this work, the proposed ExLL architecture adopts the explainability framework of the xDNN into a practical single-pass streaming model.

Due to the distinct gap in the current literature regarding algorithms that can successfully combine single-pass streaming for high-dimensional data with antecedent explainability, a direct comparative evaluation against other explainable continual learning methods is currently infeasible. Consequently, in this study, we benchmark ExLL against the state-of-the-art opaque streaming models (SLDA, Replay, NCM) to demonstrate how ExLL implements antecedent explainability without sacrificing deep learning accuracy, and with a moderate reduction of memory and computational efficiency.

3. Methodology

The proposed Explainable Lifelong Learning (ExLL) model is a feed-forward neural network with an incremental learning algorithm and a self-organizing topology. Inspired by the Episodic-to-Concept (EpCon) theoretical model of hippocampal pattern separation, ExLL is designed to mitigate intra-class catastrophic forgetting by dynamically allocating new prototypes for novel feature distributions, effectively learning from streaming data in a single pass. The architecture adopts the functionality of the xDNN model [9] with modifications for implementing a variant of Deep SLDA [15]. A pairwise fusion inferencing strategy [38] combines the MegaCloud-based global inference and prototype-based local inference. Fig. 1 provides a visual representation of the procedure for creating a novel embedding prototype.

3.1. Feature extraction and model architecture

Fig. 2 shows the architecture of the ExLL. Inputs to the network are typically images passed through a convolutional neural network (CNN) to extract discriminative features from the fully-connected layer. In our setup, the intermediate layers of the CNN are frozen after pre-training on ImageNet. While freezing the backbone prevents the feature extractor from dynamically adapting to the current distribution of the streaming data, previous work using Deep SLDA indicates that ImageNet-initialized features are sufficiently robust for standard domain transfer in streaming settings [15]. The frozen backbone will ensure that the model’s plasticity will be localized entirely on the ExLL classifier, ensuring that any catastrophic forgetting will be managed by the prototype handling mechanism rather than the shifting feature representations of the CNN intermediate layers. However, this fixed feature space relies on visual similarity to ImageNet; for highly specialized data streams with significant domain differences (such as images from medical scans), the frozen backbone represents a structural limitation.

The ExLL architecture organizes the feature vectors in a multi-dimensional topology represented as:

- **Prototypes.** Input feature vectors that are highly similar are grouped into a “data cloud”, represented by a composite feature vector known as a prototype or centroid. Prototypes represent local peaks of the data distribution sharing the same class label.
- **MegaClouds.** Adjacent data clouds that has the same class labels can be grouped into a larger structure known as a “MegaCloud”. Where a prototype represents a unique variation of a class (i.e. a red car), a MegaCloud acts as the overarching category-level prototype (i.e. any coloured car).

3.2. Core learning algorithm

Training the ExLL model occurs incrementally as follows:

1. An image I_i at time stamp i and belonging to class k is passed through a CNN. The subsequent feature vector $\tilde{x}_i$ is obtained from the fully-connected layer, and then normalized:

$$x_i = \frac{\tilde{x}_i}{\|\tilde{x}_i\|} \quad (1)$$

where $\|\cdot\|$ is the vector norm.

2. The global meta-parameters (cross-class topology) are updated online:

$$\hat{\mu}_i = \frac{i-1}{i} \hat{\mu}_{i-1} + \frac{1}{i} x_i \quad (2)$$

$$\hat{\mu}_1 = x_1$$

$$\hat{\sigma}_i = \frac{i-1}{i} \hat{\sigma}_{i-1} + \frac{1}{i} \|x_i\|^2 \quad (3)$$

$$\hat{\sigma}_1 = \|x_1\|^2 = 1$$

$$\hat{\xi}_i = \frac{i-1}{i} \hat{\xi}_{i-1} + \frac{1}{i} (x_i - \hat{\mu}_i)(x_i - \hat{\mu}_i)^T \quad (4)$$

$$\hat{\xi}_1 = (x_1)(x_1)^T$$

where $\hat{\mu}$ is the global average of all training samples, $\hat{\sigma}$ is the global variance, and $\hat{\xi}$ is the inter-class global covariance matrix.

3. If x_i introduces a novel class, the number of unique class labels is incremented, $k \leftarrow k+1$, and the local meta-parameters for the new class k are initialized:

$$i_k \leftarrow 1$$

$$g_k \leftarrow 1$$

$$\mu_{k,1} \leftarrow x_i$$

$$\sigma_{k,1} \leftarrow \|x_i\|^2 \quad (5)$$

Here, i_k denotes the number of inputs where class k was observed during training, g_k counts the prototypes in class k , $\mu_{k,1}$ is the class mean, and $\sigma_{k,1}$ is the class scalar product.

The first prototype of a new MegaCloud is also initialized:

$$p_{k,1} \leftarrow x_i$$

$$S_{k,1} \leftarrow 1$$

$$r_{k,1} \leftarrow r^*$$

$$\hat{I}_{k,1} \leftarrow I_i \quad (6)$$

where $p_{k,1}$ is the first prototype for class k , $S_{k,1}$ is the number of training samples associated with the prototype, $r_{k,1}$ is the prototype’s radius of influence initialized to a default value $r^* = \sqrt{2 - 2\cos 30^\circ}$ as defined in Angelov and Gu [37], Soares and Angelov [39].

Meanwhile, $\hat{I}_{k,1}$ is used to store the metadata of the images associated with the prototype, i.e. image file names, without actually storing the images themselves, in order to minimize the memory footprint.

However, if x_i presents a known class, then the local class mean and scalar product are updated recursively.

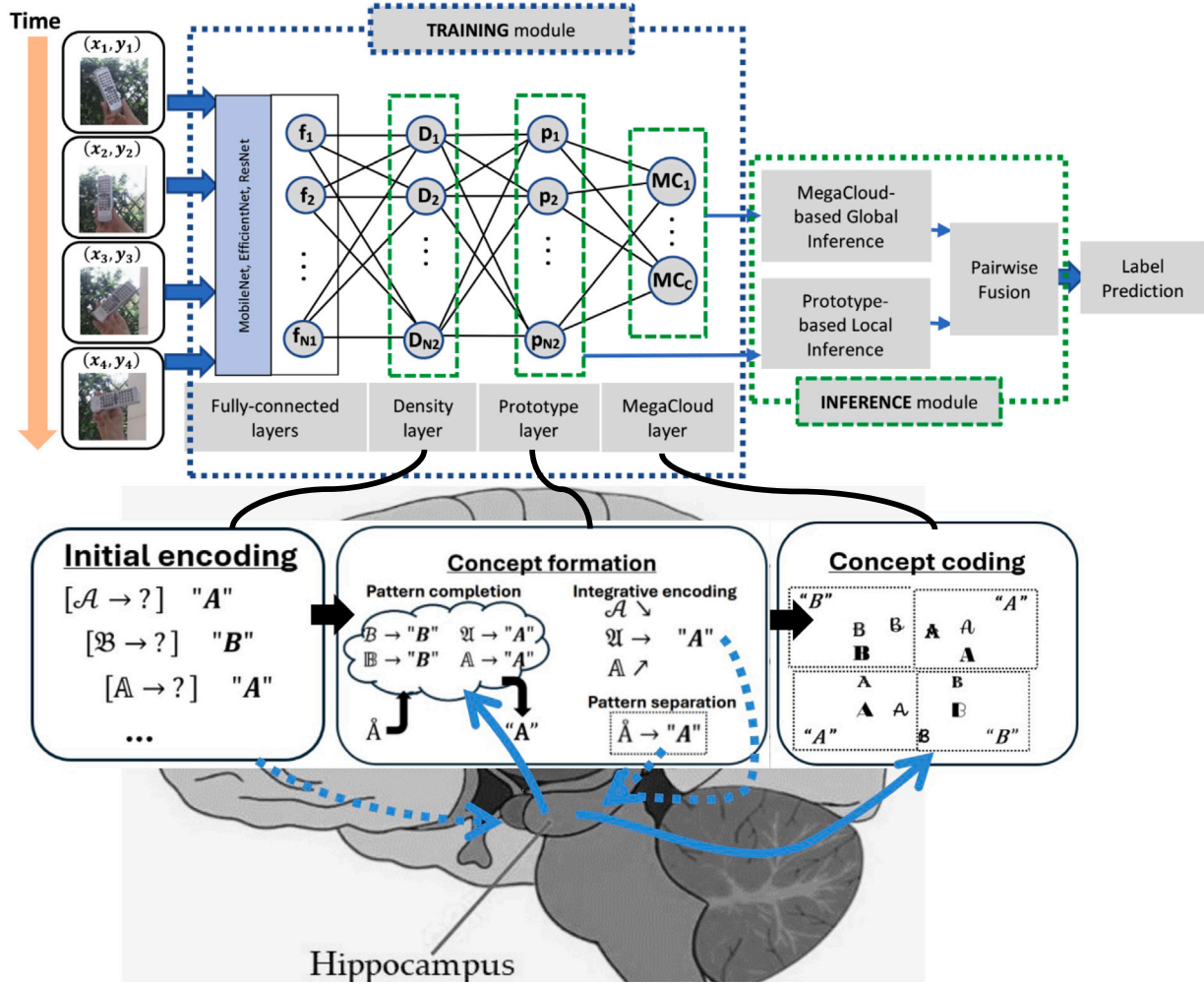


Fig. 1. Computational EpCon Model. When a feature vector is presented, the focus is directed towards specific embedding feature dimensions that align with the task's objectives, guided by the existing knowledge (attentional biasing). The feature information, weighted by attention, retrieves previously learned prototypes that share similarities (i.e. **Pattern Completion**). The outcome depends on the prediction accuracy, as indicated by the memory-based prediction error. If the error is significant, a new and distinct memory is created, unifying the current embedding features with the correct concept (i.e. **Pattern Separation**). However, if the error is minimal, the existing concept representation gets updated to encompass the newly introduced embedding features (i.e. **Pattern Integration**). This updated knowledge state, in turn, influences the approach to attention during subsequent learning encounters.

4. The **density layer** defines the mutual proximity of the incoming training images relative to the global data space. The density of input x_i relative to class k , $D(k, x_i)$ can be computed online [40]:

$$D(k, x_i) = \frac{1}{1 + \|x_i - \mu_{k,i_k}\|^2 + \sigma_{k,i_k} - \|\mu_{k,i_k}\|^2} \quad (7)$$

5. In the **prototype layer**, the nearest and second-nearest within-class prototypes, b_1 and b_2 , are identified using Mahalanobis distance [41].

To mitigate intra-class catastrophic forgetting, ExLL relies on a mechanism inspired by hippocampal pattern separation. ExLL tests the spatial distance of the new input against the nearest prototypes to determine novelty. We evaluate the following conditions, as per Angelov and Soares [9]:

$$\begin{aligned} \text{IF } D(k, x_i) &> \max_{j=1, \dots, g_k} D(k, p_j) \\ \text{OR } D(k, x_i) &\leq \min_{j=1, \dots, g_k} D(k, p_j) \end{aligned} \quad (8)$$

If the condition (8) is met, then the input x_i is considered outside the influence radius of existing prototypes and is sufficiently novel. To prevent destructive overwriting of existing knowledge, a new prototype is therefore allocated (Pattern Separation):

$$\begin{aligned} g_k &\leftarrow g_k + 1 \\ S_{k,g_k} &\leftarrow 1 \\ p_{k,g_k} &\leftarrow x_i \\ r_{k,g_k} &\leftarrow r_0 \\ \hat{I}_{k,g_k} &\leftarrow I_i \end{aligned} \quad (9)$$

This threshold prevents uncontrolled proliferation of prototypes that would take up valuable memory space by ensuring that new prototypes are only initialized when significant visual deviations are encountered.

Otherwise, if condition (8) is not met, x_i is considered sufficiently similar to existing prototypes. The parameters are then

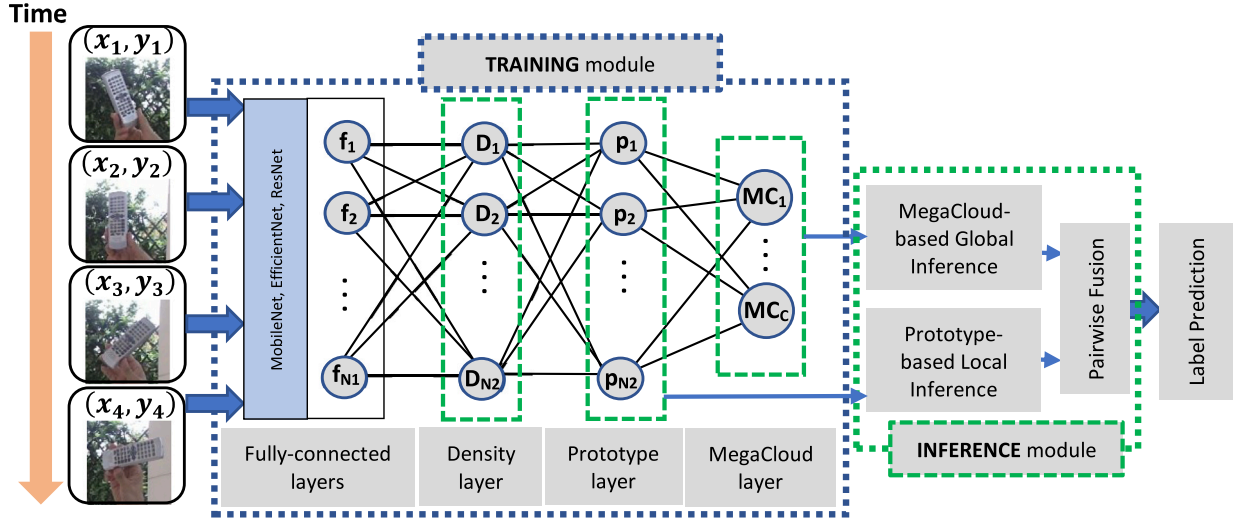


Fig. 2. Explainable Lifelong Learning architecture. Training images produce feature vectors in the fully-connected layer of a pre-trained CNN. The density of the feature vectors is then computed to determine if the training images should be assigned to an existing prototype or to initialize a new prototype. All prototypes belonging to one class label are assigned to one MegaCloud. Inference is performed once at the local level, another time at the global level. Local inference matches the inferred image to the most similar prototype while global inference matches the image to the most similar MegaCloud. Both inference decisions are then combined using a pairwise fusion matrix to obtain the final model decision.

Algorithm 1 ExLL Online Update

Require: $x_i \in \mathbb{R}^d$, $y_i = k$, prototypes $\{p_{k,j}\}$, global meta-parameters $\hat{\mu}, \hat{\xi}, \hat{\sigma}$

Ensure: Updated model

- 1: Normalization: $x_i \leftarrow x_i / \|x_i\|_2$
- 2: Update global meta-parameters $\hat{\mu}, \hat{\xi}, \hat{\sigma}$ incrementally
- 3: **if** k is novel **then**
- 4: Initialize class k : $g_k \leftarrow 1$, $p_{k,1} \leftarrow x_i$, $S_{k,1} \leftarrow 1$, $\mathcal{E}_{k,1} \leftarrow \{x_i\}$
- 5: Initialize MegaCloud: $MC_k \leftarrow \{p_{k,1}\}$
- 6: **else**
- 7: Compute $D(k, x_i)$
- 8: **if** $D(k, x_i) \leq \min_j D(k, p_{k,j})$ **OR** $D(k, x_i) > \max_j D(k, p_{k,j})$ **then**
- 9: $g_k \leftarrow g_k + 1$; $p_{k,g_k} \leftarrow x_i$; $S_{k,g_k} \leftarrow 1$; $\mathcal{E}_{k,g_k} \leftarrow \{x_i\}$
- 10: Update adjacency
- 11: **else**
- 12: Update nearest prototype b_1
- 13: **end if**
- 14: Compute $\text{PrInf } \tilde{l}$, $\text{McInf } \hat{l}$
- 15: $\Phi(k, \hat{l}, \tilde{l}) \leftarrow \Phi(k, \hat{l}, \tilde{l}) + 1$
- 16: **end if**

updated for the closest matching prototype b_1 (Pattern Completion):

$$\begin{aligned}
 S_{k,b_1} &\leftarrow S_{k,b_1} + 1 \\
 p_{k,b_1} &\leftarrow \frac{S_{k,b_1} - 1}{S_{k,b_1}} p_{k,b_1} + \frac{1}{S_{k,b_1}} x_i \\
 r_{k,b_1} &\leftarrow \sqrt{\frac{r_{k,b_1}^2 + (1 - \|p_{k,b_1}\|^2)}{2}} \\
 \hat{l}_{k,b_1} &\leftarrow \hat{l}_{k,b_1} + I_i
 \end{aligned} \tag{10}$$

This ensures that visually redundant images will reinforce existing prototypes rather than expanding the network.

3.2.1. Computational complexity

This section discusses the time and space complexity analyses of the proposed model. Let d be the feature dimension, C the number of classes, g_k the number of allocated prototypes in class k , and $G = \sum g_k$ the total number of prototypes across all classes.

Time Complexity: training per sample:

1. The incoming training image is passed through the CNN backbone, resulting in a feature vector. The time complexity for this step is static, scaling only with the number of training samples.
2. The global meta-parameters are updated from Eq. (2) to (4). Most of the update cost comes from computing the covariance matrix, $O(d^2)$.
3. Then finding the nearest within-class prototype takes linear matching time $O(g_k \cdot d)$.
4. Thus, the per-sample update time complexity is $O(d^2 + g_k \cdot d)$.

Time Complexity: inference per sample:

1. Using pre-computed projection weights, the local prototype-based inference requires scoring the training sample against all current prototypes, taking $O(G \cdot d)$ time.
2. Global inference requires scoring against each class mean, taking $O(C \cdot d)$ time.
3. Lastly, pairwise fusion takes $O(C)$ lookup operations.
4. Thus, the per-sample inference time complexity is $O(C + (G + C) \cdot d)$.

Space Complexity:

1. Static backbone parameters $O(M)$.
2. Class mean $O(C \cdot d)$.
3. Shared covariance matrix $O(d^2)$.
4. Within-class covariance matrices $O(C \cdot d^2)$.
5. Allocated prototypes $O(G \cdot d)$.
6. 3-dimensional pairwise fusion association matrix $O(C^3)$.
7. Thus, the total space complexity is $O(M + C \cdot (d + d^2) + d^2 + (G \cdot d) + C^3)$, scaling with prototype G and class C proliferation.

3.3. Inferring the explainable continual learning model

Prototype-based inference (PrInf) considers the local discriminative ability between individual prototypes while MegaCloud-based inference (McInf) globally discriminates between classes. Both types of inference have their strengths and weaknesses which adapt depending on the class distribution of the used dataset.

Pairwise fusion (PF) [38] is used as a method for combining local and global inferences to achieve better performance than either technique alone. A PF matrix Φ encodes the association between PrInf and McInf during training without prior knowledge of the dataset distribution, making it suitable for online learning.

1. Shrinkage regularization is used to compute the precision matrix from the covariance matrix $\hat{\xi}$:

$$\Lambda = [(1 - \epsilon)\hat{\xi} + (\epsilon)I]^{-1} \quad (11)$$

where I is an identity matrix and $\epsilon = 1e^{-4}$ regulates shrinkage.

2. **Prototype-based Inference (Local).** Local weights $\tilde{W}_k$ and bias $\tilde{b}_k$ are constructed using class prototypes $\tilde{p}_k$ and Λ :

$$\begin{aligned} \tilde{W}_k &= \Lambda \tilde{p}_k \\ \tilde{b}_k &= -\frac{1}{2}(\tilde{p}_k \cdot \tilde{W}_k) \end{aligned} \quad (12)$$

The local label prediction $\tilde{l}$ is then calculated by maximizing the posterior distribution $\tilde{P}(y = k|x_i)$:

$$\begin{aligned} \tilde{P}(y = k|x_i) &= \frac{\exp(\tilde{W}_k^T x_i + \tilde{b}_k^T)}{\sum_{k=1}^C \exp(\tilde{W}_k^T x_i + \tilde{b}_k^T)} \\ \tilde{l} &= \operatorname{argmax}_{k=1, \dots, C} \tilde{P}(y = k|x_i) \end{aligned} \quad (13)$$

3. **MegaCloud-based Inference (Global).** Global weights $\hat{W}$ and bias $\hat{b}$ are constructed using class mean vectors $\hat{\mu}$ and Λ :

$$\begin{aligned} \hat{W} &= \Lambda \hat{\mu} \\ \hat{b} &= -\frac{1}{2}(\hat{\mu} \cdot \hat{W}) \end{aligned} \quad (14)$$

The global label prediction $\hat{l}$ is then calculated from the posterior distribution:

$$\begin{aligned} \hat{P}(y = k|x_i) &= \frac{\exp(\hat{W}_k^T x_i + \hat{b}_k^T)}{\sum_{k=1}^C \exp(\hat{W}_k^T x_i + \hat{b}_k^T)} \\ \hat{l} &= \operatorname{argmax}_{k=1, \dots, C} \hat{P}(y = k|x_i) \end{aligned} \quad (15)$$

4. **Pairwise Fusion.** A 3-dimensional matrix Φ encodes the cumulative interactions between: (a) the actual class label, (b) the global inference $\hat{l}$, and (c) the local inference $\tilde{l}$, during streaming learning.

When performing inference, the global inference $\hat{l}$, local inference $\tilde{l}$, and Φ are used for estimating the class probabilities $P(y = k|x_i, \hat{l}, \tilde{l})$ and the final label prediction L :

$$\begin{aligned} P(y = k|x_i, \hat{l}, \tilde{l}) &= \frac{\Phi(k, \hat{l}, \tilde{l})}{\sum_{k=1}^C \Phi(k, \hat{l}, \tilde{l})} \\ L &= \operatorname{argmax}_{k=1, \dots, C} (P(y = k|x_i, \hat{l}, \tilde{l})) \end{aligned} \quad (16)$$

4. Explainability framework

4.1. Rule generation using global and local inference

Since the feature vectors are not immediately visible to users, this visual information is shown when the IF-THEN rule base is created using training images as a reference. One rule is generated from one prototype, as demonstrated in Fig. C.5(c) in the appendices. Each

prototype records the associated training images in $\hat{I}_{g_k}$, allowing a set of linguistic IF-THEN rules to be formulated as:

$$\text{IF}(I \sim \hat{I}_{k,1}) \text{ OR } \dots \text{ OR } (I \sim \hat{I}_{k,g_k}) \text{ THEN (class is } k) \quad (17)$$

When a model produces many prototypes per class, it can be unwieldy to have a rule base consisting of one rule per prototype. Some prototypes with low support scores (i.e. very few training images activate the prototype) are indicative of outliers. Pruning the outliers, excluding them from the rule base, or merging highly similar prototypes will result in a loss of information. The best alternative to preserve topology will be to generate the rule base on the basis of MegaClouds.

Using Fig. C.5(a) and (b) in the appendices as an example, neighbouring prototypes associated with the class label “Airfield” can be formed into a MegaCloud. Subsequently, each IF-THEN rule can be constructed from one MegaCloud and each explanatory image in the rule is a representative image from a different prototype in the MegaCloud. Using Figs. C.5(c) and (d) as examples, the explainable IF-THEN rule for MegaCloud #1 is constructed using images from Prototypes #1, #2, and #3.

Using this method to construct the explainable rule base has several advantages. The size of the rule base is more compact in the case where a large number of prototypes are involved by reducing redundancies. Each MegaCloud consists of multiple adjacent prototypes, meaning they are highly similar. However, the visual information associated with each prototype is less obvious when converted into the MegaCloud rule base. Therefore, it is useful to have both explainable rule bases simultaneously. One to facilitate inter-class explanations sourced from the global MegaCloud-based inferences and another to provide intra-class explanations sourced from local prototype-based inferences.

4.2. Visualizing decision boundaries: Hits, near hits, and near misses

ExLL incorporates the element of explainability at the inference stage so that label predictions can be explained in terms of “Hits”, “Near Hits” and “Near Misses” as proposed by Herchenbach et al. [42].

When a winning prototype $b_{L1,1}$ is selected for the winning label L_1 during inference, its training records $\hat{I}_{b_{L1,1}}$ is referenced to retrieve the images used to train the prototype. The retrieved images are then shown as a visual explanation, “Hit”, to justify why the inferred image was assigned to that prototype based on spatial and visual similarity.

To provide context on alternative decisions, “Near Hit” shows the training images $\hat{I}_{b_{L1,2}}$ associated with the second-best matching within-class prototype $b_{L1,2}$. Lastly, “Near Miss” shows the training images $\hat{I}_{b_{L2,1}}$ associated with the winning prototype $b_{L2,1}$ for the next-best alternative class L_2 . The visual explanations provided by the “Near Hits” and “Near Misses” describe the decision boundary of the ExLL’s prediction. In edge cases where the predictions are ambiguous, the visual comparison of “Hits”, “Near Hits”, and “Near Misses” informs the user of possible alternatives. For incorrect predictions, such as False Positives or False Negatives, the images allow users to determine how and why the model made a mistake.

Fig. 3 demonstrates an example of a correct prediction and two wrong predictions. The top row illustrates the explanation for a True Positive prediction. An image of a shampoo bottle is correctly classified and the training images shown under “Hits” justify the selection of the best-matching prototype due to their visual similarity. The training image from the second-best matching prototype, shown under “Near Hits”, also explains why that prototype is not selected due to the visual dissimilarity to the inferred image. Lastly, “Near Misses” show why the test image is almost incorrectly classified as “Lotion” by showing the associated training images of the best prototype from the next-best class label.

For incorrect predictions such as the False Negative result in the middle row and the False Positive result in the bottom row, the images

Test Image	Hits	Near Hits	Near Misses
 “Shampoo” correctly predicted as “Shampoo”	 “Shampoo”	 “Shampoo”	 “Lotion”
 “Shampoo” Wrongly predicted as “Lotion”	 “Lotion”	 “Lotion”	 “Toothpaste”
 “Toothpaste” Wrongly predicted as “Shampoo”	 “Shampoo”	 “Shampoo”	 “Book”

Fig. 3. Example of “Hits”, “Near Hits”, and “Near Misses” for the F-SIOL-310 dataset. The test image in the top row is an example of a True Positive result, the test image in the middle row is a False Negative result, and the test image in the bottom row is a False Positive result.

shown for “Hits”, “Near Hits”, and “Near Misses” explain why the ExLL made the mistake. For example, in the False Negative result in the middle row, the visual explanation for “Hits” explained that the incorrect prediction is due to the colour and shape similarity.

5. Experiment setup

Given a continuous stream of images where X_t is an image at time t , a neural network classifier $F(\cdot)$ is trained incrementally using supervised online continual learning, producing a predicted label $\hat{y}_t = F(G(X_t))$. The backbone CNN $G(\cdot)$ is pre-trained on the ImageNet-1k dataset [43]. Subsequently, feature vectors are obtained from the last hidden fully-connected layer in response to training images fed to $G(\cdot)$, which are then passed to the streaming classifier $F(\cdot)$ for learning. The intermediate layers in $G(\cdot)$ are frozen after pre-training to prevent knowledge drift. While freezing the backbone prevents the feature extractor from adapting to the more recent distribution of the streaming data (i.e. domain shift), previous work using Deep SLDA [15] indicated that the ImageNet-initialized networks are sufficiently robust for domain transfer in streaming settings. Freezing the backbone will minimize plasticity from the feature extractor in response to domain shift, leaving only the ExLL classifier to manage any catastrophic forgetting.

5.1. Experiment protocol

We evaluate the proposed ExLL model under edge computing constraints, using the embedded continual learning benchmark protocol established by Hayes and Kanan [10]. The protocol evaluated algorithms based on their ability to learn instance-by-instance in a single pass without storing past data for full retraining.

5.2. Datasets and tasks

Online continual learners are evaluated using the following image classification datasets, which test the robustness of the models to scale, class imbalance, and fine-grained instance variations:

OpenLORIS [44] consists of videos and image sequences of 40 different household items recorded from varying angles and distances, and 121 object instances. Each object instance is recorded under one of the following environmental conditions: the object is surrounded by clutter; the object is illuminated by several light sources; the object is partially occluded; and the object is nearer to or further away from the camera. Two ordering strategies are employed: “Instance Ordering” tests fine-grained instance-level discrimination, while “Low-Shot Instance Ordering” presents only one sequence of images from each object instance to evaluate out-of-domain generalization.

Places-365 [45] consists of 1.8 million images of locations divided into 365 categories, serving as a test for large-scale generalization of many diverse class labels. **Places-Long-Tail** (Places-LT) is a subset of Places-365 with a skewed distribution of images (5 to 4980 images per class), designed to evaluate a model’s robustness against severe data imbalance.

Few-Shot Incremental Object Learning (F-SIOL-310) [20] consists of multiple instances of 22 household items, totalling 310 object instances. Similar to OpenLORIS, this dataset is utilized to test few-shot fine-grained instance discrimination, specifically under 5-shot and 10-shot learning scenarios.

5.3. Streaming learning baseline models

We assess how well ExLL performs against the selected algorithms for lightweight, distribution-based streaming classification.

Online Perceptron (Pcpt) [16] maintains a single weight vector for every unique class it encounters, classifying an incoming feature vector by assigning it to the class that produces the highest dot product score. During training, the model only updates its vectors when a misclassification occurs, by adjusting the true class vector towards the input and shifting the incorrectly predicted class vector away from it.

Fine-Tune (FT) [10] incrementally adjusts a CNN’s fully-connected layer. A stochastic gradient descent optimization strategy is used and progress is measured using cross-entropy loss of the CNN’s predictions.

Streaming Gaussian Naive Bayes (NB) [17] is an extension of the Gaussian Naive Bayes algorithm that learns from continuous data

streams by incrementally updating a running mean and variance vector for each unique class. During inference, Bayes' theorem is applied to obtain the posterior probability of each class.

Streaming One-vs-Rest (SOvR) [10] maintains a series of binary classifiers, one for each unique class it encounters during training. As each new feature vector continuously arrives in a streaming scenario, the classifier for the relevant class of the current input is updated incrementally. During inference, each classifier outputs a confidence score on whether the inferred feature vector belongs to that class. The final predicted class is selected from the classifier with the best confidence score.

Nearest Class Mean (NCM) [7] keeps a cumulative average vector for every unique class, classifying inputs based on proximity to these centroids.

Replay [46] A rehearsal technique that stores a strict subset of previous training feature vectors in a memory buffer, learning equally from incoming and stored vectors to mitigate forgetting.

Streaming Linear Discriminant Analysis (SLDA) [15] maintains running means for each class and a shared covariance matrix to model data distribution, effectively finding a linear projection that maximizes class separation.

5.4. Backbone architectures

To ensure the findings apply to embedded environments, the algorithms are tested across three compact CNN families. Again, the choice of backbones mirror the selection in the experiment by Hayes and Kanan [10]:

MobileNet-v3 by Howard et al. [47] is the successor of two previous architectures created for mobile and embedded applications [27, 48]. MobileNet-v3 Small (**MNet-S**), with 2.54M parameters, is more efficient but displays worse classification performance compared to MobileNet-v3 Large (**MNet-L**), with 5.47M parameters, is more resource-intensive but shows better classification performance.

EfficientNet by Tan and Le [29] is a family of models with varying complexity. Two models with the least complexity are compared in this study: EfficientNet-B0 (**ENet-B0**) with 5.3M parameters and EfficientNet-B1 (**ENet-B1**) with 7.8M parameters.

ResNet by He et al. [49] includes several types of models with varying complexity. The smallest model, ResNet-18 (**RN-18**), is selected for this study as the most suitable ResNet candidate to be used in embedded systems and has been extensively tested in continual learning studies [18,50–53]. Of the selected backbones, ResNet-18 is the largest with 11.5M parameters.

6. Results and discussion

We present findings across standard and fine-grained streaming tasks. For OpenLORIS, Places-LT, and F-SIOL-310, the results are reported from the average performance across three permutations for each data ordering technique. However, Places-365 is run only once for each ordering due to the long time needed to complete a single run. Due to space constraints, some results for the Places and F-SIOL-310 datasets are moved to the Appendices at the end of the manuscript.

6.1. Results on OpenLORIS

Online continual learners are evaluated on OpenLORIS using two data ordering methods. **Instance ordering** trains learners on all object instances while **low-shot instance ordering** trains learners on one object instance from each object class. Performance is evaluated by combining the top-1 accuracy scores of the top-ranked choice for each learner. The scores are then averaged across all CNN architectures to compare how various orderings affect learner performance, as shown in Fig. 4.

Table 1

NetScore results on the OpenLORIS dataset with the **low-shot instance ordering**. Higher NetScores are better. The right-most column reports the averaged ranking of each model across the five CNN backbones, where lower average ranks are better. Results are highlighted as follows for the **first**, **second**, and **third** best results.

Method	MNetS	MNetL	ENetB0	ENetB1	RN18	Avg. Rank
Pcpt	-115.9	-106.5	-91.0	-96.5	-147.8	6.4
FT	-149.0	-142.9	-96.3	-103.7	-187.8	7.4
NB	-83.7	-77.3	-75.8	-84.2	-204.5	5.6
SOvR	-80.0	-83.9	-74.0	-78.7	-111.4	4.4
NCM	-55.5	-64.7	-65.3	-72.5	-78.7	1.0
Replay	-58.9	-66.5	-66.6	-72.8	-80.5	2.2
SLDA	-58.3	-69.1	-72.3	-79.3	-80.8	3.0
ExLL	-75.3	-91.4	-99.5	-106.8	-108.1	6.0

Table 2

Accuracy results on OpenLORIS with the **instance ordering**. Results are highlighted as follows for the **first**, **second**, and **third** best results.

Method	MNet-S	MNet-L	ENet-B0	ENet-B1	RN-18	Mean
Pcpt	0.793	0.880	0.935	0.942	0.796	0.869
FT	0.835	0.915	0.958	0.963	0.809	0.896
NB	0.311	0.526	0.780	0.787	0.015	0.483
SOvR	0.374	0.477	0.739	0.723	0.346	0.531
NCM	0.729	0.789	0.859	0.867	0.797	0.808
Replay	0.921	0.956	0.977	0.978	0.929	0.952
SLDA	0.956	0.982	0.988	0.988	0.950	0.972
ExLL-M	0.944	0.973	0.982	0.982	0.940	0.964
ExLL-P	0.951	0.968	0.982	0.983	0.961	0.969
ExLL-F	0.987	0.993	0.996	0.996	0.988	0.992

6.1.1. Netscore performance

The models are evaluated on three axes: classification accuracy, number of parameters, and experiment runtime. The performance of an online learner $\mathcal{M}$ is computed as a modified NetScore metric [54] combining all three metrics into one score as follows:

$$\Omega(\mathcal{M}) = s \log\left(\frac{a(\mathcal{M})^\alpha}{p(\mathcal{M})^\beta c(\mathcal{M})^\gamma}\right) \quad (18)$$

where $a(\mathcal{M})$ is the learner's testing accuracy, $p(\mathcal{M})$ is the learner's size, $c(\mathcal{M})$ is the time taken to complete the experiment from start to finish, and α, β, γ are user-defined constants for controlling the contributions of accuracy, number of parameters, and the experiment runtime towards computing the NetScore Ω .

The NetScore parameters follow the original parameter settings as suggested by Hayes and Kanan [10]: $s = 20$ and $\alpha = 2$ to prioritize classification accuracy, and $\beta = \gamma = 0.25$ to moderate the large values of $p(\mathcal{M})$ and $c(\mathcal{M})$ as shown in Wong [54]. Higher NetScores indicate better performance.

In Table 1, the top three performing online continual learners are NCM, Replay, and SLDA. The NCM algorithm is the most efficient in terms of memory and computation requirements since it only stores and updates the class mean vectors. Replay needed additional computation and memory for replaying stored samples, while SLDA needed additional computation and memory for the covariance matrix. As memory requirements form the basis of NetScores, all models performed poorly on high-dimensional backbone architectures such as ResNet-18.

The ExLL results are averaged across all three inference strategies, and ranks 6th on average among the eight algorithms. While ExLL is nominally similar to SLDA, ExLL required significantly more memory to store prototypes and training records, and thus performs poorly for datasets with large number of classes.

6.1.2. Backbone CNN comparisons

Table 2 presents the best accuracy scores of online continual learning models across different CNN backbones when trained using **instance ordering**. Among the backbone architectures, the EfficientNet

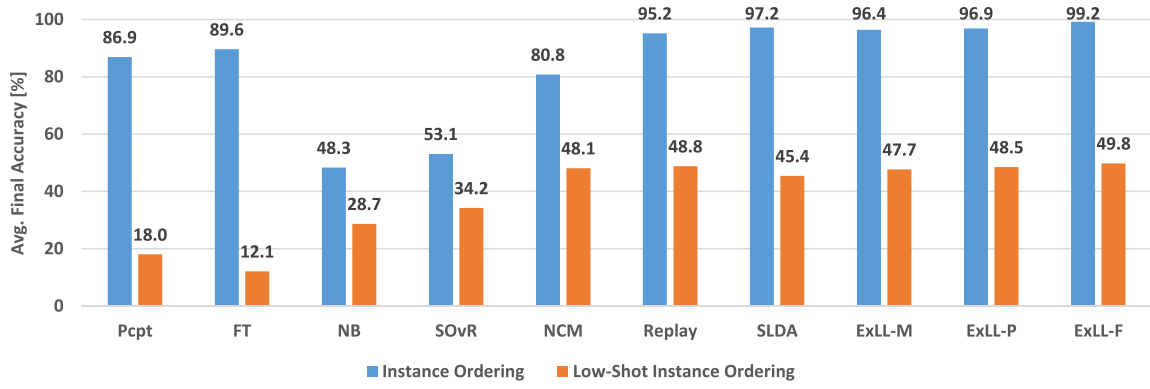


Fig. 4. Accuracy results averaged across CNN architectures comparing online continual learners' performance on OpenLORIS with **instance ordering** and **low-shot instance ordering**.

Table 3

Accuracy results on OpenLORIS with the **low-shot instance ordering**. Results are highlighted as follows for the first, second, and third best results.

Method	MNet-S	MNet-L	ENet-B0	ENet-B1	RN-18	Mean
Pcpt	0.098	0.167	0.272	0.283	0.082	0.180
FT	0.043	0.066	0.238	0.232	0.030	0.121
NB	0.232	0.366	0.421	0.399	0.021	0.287
SOvR	0.259	0.323	0.449	0.459	0.224	0.342
NCM	0.442	0.474	0.516	0.514	<u>0.463</u>	0.481
Replay	0.453	0.480	0.529	0.532	0.446	0.488
SLDA	0.445	0.454	0.472	0.460	0.442	0.454
ExLL-M	<u>0.463</u>	<u>0.493</u>	0.504	0.487	0.440	0.477
ExLL-P	0.470	0.501	0.500	0.482	0.475	<u>0.485</u>
ExLL-F	0.481	0.511	<u>0.511</u>	<u>0.495</u>	0.492	0.498

architectures performed better compared to MobileNet and ResNet architectures for all of the benchmarked continual learning methods. ExLL-F displays the best accuracy scores among the methods for all backbones, while SLDA is on average the second-best performing method outperforming ExLL-M and ExLL-P.

For comparison, Table 3 presents the performance of the models across different CNN architectures for **low-shot instance ordering**. Learners that show a bigger drop in low-shot instance ordering compared to instance ordering demonstrate poor generalization to out-of-domain inputs.

In the instance ordering setting, ExLL-F is the best performing model, followed by SLDA, while ExLL-P and ExLL-M are third- and fourth-best respectively. Under the low-shot instance ordering, however, ExLL-P is the third-best after Replay and ExLL-M is the fifth-best after NCM. In some cases, such as when using the EfficientNet backbone architectures, ExLL models are outperformed by NCM and Replay methods.

This difference compared to the results in the previous table highlights the limitations of the ExLL ability to generalize to unknown domains when given limited training data. Since SLDA is observed to be similarly outperformed by NCM and Replay when using EfficientNet, we theorized that the inter-class covariance matrices used by SLDA and ExLL perform poorly when few training samples are provided and when the dimensionality of the data is high. Both EfficientNet backbones have a higher feature dimensionality (1280) compared to the MobileNets (576 and 960) and ResNet (512). Historically, SLDA performed poorly when high-dimension features are involved [55].

Overall, ExLL-F outperforms all other models in both ordering methods. The ExLL's local and global inferencing performed poorly when limited data is available. However, the pairwise decision fusion method

Table 4

Accuracy results on Places-365 and Places-LT averaged across CNN architectures. The IID and Class IID results are computed via arithmetic mean across all CNN architectures and the overall performance is computed as the harmonic mean (**H-Mean**) of the two numbers.

Method	Places-365			Places-LT		
	IID	C-IID	H-Mean	IID	C-IID	H-Mean
Pcpt	0.327	0.007	0.014	0.181	0.041	0.067
FT	0.251	0.004	0.008	0.166	0.037	0.061
NB	0.125	0.125	0.125	0.115	0.096	0.105
NCM	0.331	0.308	0.319	0.308	0.308	0.308
Replay	0.315	0.271	0.292	0.260	0.261	0.260
SLDA	0.388	0.388	0.388	<u>0.315</u>	<u>0.315</u>	<u>0.315</u>
ExLL-M	0.352	0.361	0.357	0.383	0.331	0.355
ExLL-P	<u>0.366</u>	<u>0.365</u>	<u>0.365</u>	0.292	0.291	0.291
ExLL-F	0.465	0.464	0.465	0.343	0.343	0.343

is able to complement the information from both inferences to produce a more robust prediction accuracy. Lastly, ExLL models have poor out-of-domain generalization performance compared to state-of-the-art.

6.2. Results on Places-365 and Places-LT

Table 4 presents the average top-1 accuracies on both Places datasets for all continual learning methods, averaged across all backbone architectures. The overall accuracy is calculated using the harmonic mean of IID and Class IID results to highlight the significance of performing effectively in either extremes. Models which perform well for IID but poorly for Class IID demonstrate a poor ability to retain information, i.e. catastrophic forgetting. In addition, the differences of the results between Places-365 and Places-LT highlights the models' resistance to highly imbalanced classes. Due to space constraints, detailed results for the Places datasets are moved to the appendices.

SLDA outperformed ExLL-P and ExLL-M for Places-365. However, when pairwise fusion was used for combining the results from the two ExLL methods, ExLL-F was able to outperform SLDA, ExLL-P, and ExLL-M by a significant margin (0.465 v.s. 0.388, 0.365 and 0.357). This suggests that the local and global inferences in ExLL-F contain highly complementary information and are able to address each other's weaknesses when pairwise fusion is used. This is a similar finding from the results in Tables 2 and 3.

Of the three ExLL methods, global inference was the least affected by dataset imbalance while prototype-based inference and pairwise fusion showed a 7.4% and 12.2% loss in performance respectively when

trained with Places-LT. This finding suggests that global inference is more robust against dataset imbalance than the other ExLL inference strategies.

6.3. Summary of results

6.3.1. Classification performance on streaming benchmarks

The fusion model ExLL-F achieves the highest mean accuracy across all backbone architectures on the OpenLORIS instance ordering, averaging 99.2% classification accuracy. The fine-grained instance discrimination task highlights ExLL's primary advantage over single-centroid models like NCM or SLDA: by initializing new prototypes by novelty for pattern separation. This preserves the multi-modal variations within a class (e.g., distinct angles or lighting of the same object) and reducing the impact of intra-class catastrophic forgetting that degrades the performance of models forced to average diverse views into a single running mean.

Similarly, on the large-scale Places-365 benchmark, ExLL-F achieves a state-of-the-art overall harmonic mean accuracy of 0.465, significantly outperforming the top baselines SLDA (0.388) and Replay (0.315) in the IID setting.

6.3.2. NetScore: Efficiency vs transparency

In Table 1, NetScores are used for evaluating the continual learning methods by balancing classification performance against model efficiency metrics such as memory storage and computational time. The ExLL ranks 6th on average among the tested learners, heavily penalized by its memory usage compared to highly efficient methods like NCM (Rank 1) and SLDA (Rank 3).

We argue that the NetScore metric biases towards simple efficiency. While NCM and SLDA require minimal memory because they compress entire classes into single abstract mean vectors or shared covariance matrices, this extreme compression acts as a "black box" that makes a model opaque against explainability methods. In critical applications such as medical monitoring or autonomous service robots, extreme efficiency should be balanced against the needs for user safety and diagnostic transparency. Therefore, the model inefficiency caused by ExLL's storing multiple discrete prototypes and their associated training records, is a deliberate design choice: a necessary increase in memory and computing footprint in exchange for the ability to explain the model predictions (i.e. via Hits, Near Hits, and Near Misses) explained in Section 4.2.

6.3.3. Ablation study of inference strategies

To determine the contributions of the global and local inference mechanisms within the proposed architecture, we performed an ablation study comparing the three variations of the model across the benchmark datasets:

- **ExLL-P** uses local, prototype-level inferences based on spatial matching (Eq. (13)).
- **ExLL-M** uses global, MegaCloud-level inferences based on category-level matching (Eq. (15)).
- **ExLL-F** combines the local and global decisions using the Pairwise Fusion matrix Φ (Eq. (16)).

The results presented in show that pairwise fusion contributes a significant accuracy gain over local and global inferences (0.992 for ExLL-F v.s. 0.964 for ExLL-M and 0.969 for ExLL-P). This indicates that the local prototype representations and global MegaCloud representations contain complementary information, allowing the fusion matrix to successfully correct localized errors using global context.

Across standard streaming benchmarks, the results demonstrated that combining local and global contexts using pairwise fusion provides a significant accuracy gain over using either component individually. On the fine-grained OpenLORIS "Instance Ordering" task (Table 2),

ExLL-F achieved a near-perfect mean accuracy of 0.992, successfully correcting errors made by the standalone ExLL-P (0.969) and ExLL-M (0.964) models.

The synergistic effect is also demonstrated in large-scale dataset learning. On Places-365 (Table 4), the standalone ExLL-P and ExLL-M models achieved overall harmonic mean accuracies of 0.365 and 0.357, respectively. However, when pairwise fusion was applied, ExLL-F's performance increased to 0.465. This significant improvement validates the hypothesis that local prototype inferences and global MegaCloud inferences capture highly complementary information, allowing the fusion matrix to address each strategy's individual weaknesses.

When evaluating the models under dataset imbalance conditions as shown for Places-LT in Table 4, ExLL-P suffered a significant performance drop when transitioning from Places-365 to Places-LT, dropping to a harmonic mean of 0.291. The global ExLL-M was not as affected by dataset imbalance, achieving a harmonic mean of 0.355. This finding suggests that while local prototype matching (ExLL-P) is highly sensitive to the under-representation, global MegaCloud matching (ExLL-M) is resistant against dataset imbalance. Furthermore, the ExLL-F model was able to leverage the stability of the global context to address the weaknesses of the local context, achieving a slightly more modest harmonic mean of 0.343.

In summary, this ablation study highlights the limitations of both local and global strategies under extreme out-of-domain data scarcity (OpenLORIS Low-Shot Instance Ordering), compared to the more standardized data scarcity conditions imposed by the F-SIOL-310 dataset. When the models were given only one training video per class but tested across severe environmental shifts, both ExLL-M (0.477) and ExLL-P (0.485) performed poorly. Although ExLL-F managed to extract complementary information to obtain the best prediction among the three variants (0.498), the overall drop in performance highlights that neither local nor global inference can entirely overcome the domain shift limitations imposed by the frozen backbone feature extractor.

6.3.4. Model limitations in low-shot generalization and domain shift

The findings in Tables 3 and B.7 show a contrasting limitation under data scarcity conditions. While ExLL models exhibit poor out-of-domain generalization in OpenLORIS's "Low-Shot Instance Ordering" setting compared to Replay and NCM methods, they are able to achieve near state-of-the-art for the F-SIOL-310 5-shot and 10-shot learning scenarios.

This can be explained by the different aims of the two datasets. Where F-SIOL-310 tests standard few-shot discrimination, OpenLORIS tests out-of-domain generalization by training on a single instance but testing across significant variations in illumination, clutter, and occlusion. The differences in the ExLL performance are indicative of ExLL's effective ability to model few-shot data within a consistent domain (i.e. F-SIOL-310) but is unable to account for a significant domain shift, especially since the backbone feature extractors are frozen beforehand. Since the CNN backbone is frozen to prevent knowledge drift, ExLL relies entirely on a fixed metric space optimized for ImageNet. While ExLL's prototype allocation efficiently models few-shot data within a consistent domain, a single training instance does not provide the prototypes enough manifold coverage to bridge a severe domain shift. This vulnerability will be significantly exacerbated if the model is deployed on data streams with severe domain differences from ImageNet, such as medical imaging (e.g., CT scans or MRIs) or remote sensing. In these specialized environments, the fixed ImageNet feature space lacks the domain-specific discriminability required for the ExLL prototypes to separate classes effectively.

7. Potential practical and engineering applications

To validate the real-world utility of the proposed model, we identified three distinct engineering domains where the ExLL's single-pass learning and antecedent explainability would be beneficial.

Wearable Medical Monitoring Devices. In physiological signal monitoring (e.g., continuous on-body ECG or wearable pulse-oximetry), classifiers must adapt to patient-specific cardiac baselines and physiological drift in real-time. Standard deep neural networks are unfeasible due to catastrophic forgetting and lack of transparency. ExLL enables on-device, single-pass adaptation to patient-specific indicators while minimizing catastrophic forgetting on globally-learned medical representations. In addition, its antecedent explainability framework allows medical specialists to audit diagnostic predictions by directly observing the “Hits” (matching training prototypes) and “Near Misses” (borderline counter-examples), preventing “black-box” medical errors.

Personalized Service Robotics. Service and assistive robotics must quickly adapt to its current household environment, identifying personal objects such as a user’s specific medicine bottle, from few on-the-spot demonstrations (few-shot streaming). ExLL’s “allocate-on-novelty” mechanism allows the robot to instantly allocate new prototypes to capture diverse visual angles and lighting conditions. By referencing “Near Hits”, the robot can explain its recognition boundaries to the user, ensuring operational safety and user trust.

Autonomous Vehicle Edge Inspection. In automated structural or manufacturing quality control, models must classify defects under changing environmental settings. ExLL can be deployed on embedded edge cameras to learn and classify visual anomalies. When a novel defect is encountered, ExLL adapts without forgetting previously-learned defect classes. Engineers can then audit misclassified anomalies on the factory floor using the contrastive “Near Miss” boundaries to determine if the failure was caused by a domain shift or a physical structural defect.

8. Conclusion and future research direction

We propose an explainable neural network architecture, the Explainable Lifelong Learning (ExLL) model, for online continual learning in edge devices. Inspired by the Episodic-to-Concept theoretical model of hippocampal pattern separation, ExLL facilitates single-pass learning from continuous data streams. Using a dynamic “allocate-on-novelty” prototype architecture, ExLL mitigates intra-class catastrophic forgetting, preserving multi-modal feature variations that are typically lost by single-centroid models like SLDA. Furthermore, our collective inference strategy, which combines global MegaCloud context with local prototype matching via pairwise decision fusion, allows ExLL to achieve state-of-the-art predictive accuracy across diverse streaming scenarios, including fine-grained instance discrimination and large-scale dataset learning.

The classifier’s performance was benchmarked against established lightweight streaming models under strict embedded hardware constraints. While methods such as Nearest Class Mean (NCM) and SLDA achieve higher overall NetScores due to their extremely minimal memory footprints, they inherently operate as opaque “black boxes” by compressing entire classes into abstract statistical summaries. In this study, we demonstrate that ExLL trades extreme memory efficiency for the granular transparency required in high-stakes environments. The calculated memory overhead of storing multiple prototypes and associated reference images enables ExLL to provide antecedent explainability. By visualizing the decision boundaries through concrete “Hits”, “Near Hits”, and “Near Misses”, ExLL allows users to debug the model’s reasoning. Therefore, while NCM and SLDA may be more efficient, ExLL is more suitable for medical, autonomous, and service robotics applications where diagnostic transparency and user trust are important.

The ExLL is developed in part thanks to the following researchers: Angelov and Soares [9] for the development of their eXplainable Deep Neural Networks (xDNN) architecture¹ as well as Hayes and Kanan

[15] for the development of their Deep Streaming Linear Discriminant Analysis (Deep SLDA) architecture.²

One identified limitation of our current approach is the proliferation of prototypes over a long period of time. The linear $O(g_k \cdot d)$ time complexity of the matching step will inevitably become a computational bottleneck as the number of prototypes increases. To lower the computational load in very large-scale, long-term streaming environments, future implementations should incorporate methods such as Approximate Nearest Neighbour (ANN) search algorithms to reduce the matching complexity to sub-linear time. Pruning strategies may also help identify low-utility prototypes that can be pruned without significant catastrophic forgetting, thus reducing the number of parameter requirements of the model [56–58].

Dynamic pruning strategies can be employed to actively identify obsolete prototypes based on activation frequency (support) and age of prototypes to discard low-utility outliers that are no longer relevant to the stream. Where pruning prototypes may lead to loss of institutional knowledge of edge cases, another strategy involves merging within-class prototypes that are spatially-adjacent to preserve an approximation of their knowledge without fully discarding them. Developing these strategies will help optimize ExLL’s memory requirements, ensuring long-term computational efficiency without sacrificing its foundational explainability.

Other explainability techniques can be applied to enhance interpretability, including the use of gradient class activation maps to visualize discriminative image features [33,59]. Combined with selective feature weighing to ignore redundant features [60], this may help reduce the dimensionality and computation required by the model.

One potential utility that can be added to the ExLL is the ability to identify unknown classes, i.e. out-of-vocab (OOV) classification. In continual lifelong learning, the model would sometimes be given objects during the inferring phase that it has not been trained on during the training phase. Currently the ExLL classifies novel objects by assigning them to the closest known class labels. A confidence threshold calculated during inference, for example, can be used to differentiate between known and unknown objects [61]. Research in this area is currently ongoing.

Currently, ExLL relies on a frozen CNN backbone pretrained via supervised learning on ImageNet. If the model is presented with a different dataset however, this introduces a domain shift if the new dataset has a different environment (i.e. occluded objects, lighting changes). Replacing this supervised backbone with modern Self-Supervised Learning feature extractors such as DINOv2, is another promising direction in order to provide a more generalized and robust metric space for prototype matching without needing on-device backpropagation.

Alternatively, to deploy ExLL in specialized domains where ImageNet or generalized features are insufficient, partially adaptive backbone networks may address the issue of domain shift. By selectively fine-tuning the final convolutional layers, the network would be able to adapt to the new domain. However, implementing these mechanisms require careful optimization to balance the stability-plasticity dilemma and manage the computational overhead of partial backpropagation on embedded devices.

CRedit authorship contribution statement

Chu Kiong Loo: Writing – review & editing, Writing – original draft, Supervision, Resources, Project administration, Methodology, Funding acquisition, Formal analysis, Conceptualization. **Yi Chen:** Software, Methodology, Investigation, Data curation. **Wei Shiung Liew:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation. **Stefan Wermter:** Writing – review & editing, Supervision.

¹ <https://github.com/Plamen-Eduardo/xDNN-Python>

² <https://github.com/tyler-hayes/SLDA-Tutorial>

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors acknowledge the support from the German Research Foundation (Deutsche Forschungsgemeinschaft/DFG) under project CML (TRR 169), the TRANSPARENT, InterpretABLE Robots (TRAIL) EU project, the BMWK under project VeriKAS, and Equipment Subsidies from the Alexander von Humboldt-Foundation (3.4-1191526-MYS-GA) and UM Matching Grant (IF001-2024).

Appendix A. Additional results for Places-365 and Places-LT

Table A.5 shows the detailed top-1 accuracies achieved by each continual learning method and backbone architecture using Places-365 using two data ordering methods, IID and Class IID. For the IID data ordering method, the results show that the ExLL-F method consistently outperforms the other methods across all architectures. Specifically, ExLL-F achieves an overall mean accuracy of 0.465. In comparison, SLDA is the second-best performer with an overall mean accuracy of 0.388, outperforming ExLL-P (0.366) and ExLL-M (0.352). Under the Class-IID data ordering method, ExLL-F achieves an overall mean accuracy of 0.464 while the second-best performer SLDA achieves an overall mean accuracy of 0.388, again outperforming ExLL-P (0.365) and ExLL-M (0.361).

These results demonstrate the effectiveness of the ExLL-F method in maintaining high accuracy across different architectures and data ordering methods. As mentioned earlier, the difference between IID and Class IID methods is the distribution of samples. Models that achieve similar results for both methods are robust against different data distributions and perform well in situations such as real-world conditions and imbalanced class distributions. The results from the table showed that ExLL-F is not only state-of-the-art but also robust and generalizable across different data distributions. Similarly, Table A.6

Table A.5

Accuracy results on Places-365 for two data ordering methods IID and Class IID. Results are highlighted as follows for the first, second, and third best results.

Method	IID					
	MNet-S	MNet-L	ENet-B0	ENet-B1	RN-18	Mean
Pcpt	0.303	0.344	0.352	0.340	0.294	0.327
FT	0.214	0.252	0.293	0.280	0.217	0.251
NB	0.028	0.093	0.250	0.249	0.003	0.125
NCM	0.285	0.332	0.361	0.356	0.322	0.331
Replay	0.289	0.323	0.354	0.348	0.261	0.315
SLDA	<u>0.362</u>	<u>0.397</u>	<u>0.412</u>	<u>0.405</u>	<u>0.362</u>	<u>0.388</u>
ExLL-M	<u>0.375</u>	0.347	<u>0.392</u>	0.336	0.312	0.352
ExLL-P	0.354	<u>0.380</u>	0.381	<u>0.370</u>	<u>0.343</u>	<u>0.366</u>
ExLL-F	<u>0.444</u>	<u>0.478</u>	<u>0.488</u>	<u>0.476</u>	<u>0.440</u>	<u>0.465</u>

Method	Class-IID					
	MNet-S	MNet-L	ENet-B0	ENet-B1	RN-18	Mean
Pcpt	0.004	0.003	0.012	0.013	0.005	0.007
FT	0.003	0.003	0.006	0.006	0.003	0.004
NB	0.028	0.093	0.250	0.249	0.003	0.125
NCM	0.265	0.309	0.336	0.329	0.300	0.308
Replay	0.251	0.279	0.297	0.295	0.235	0.271
SLDA	<u>0.362</u>	<u>0.397</u>	<u>0.412</u>	<u>0.405</u>	<u>0.362</u>	<u>0.388</u>
ExLL-M	0.347	0.362	<u>0.381</u>	0.367	<u>0.349</u>	0.361
ExLL-P	<u>0.352</u>	<u>0.378</u>	<u>0.381</u>	<u>0.373</u>	<u>0.343</u>	<u>0.365</u>
ExLL-F	<u>0.444</u>	<u>0.473</u>	<u>0.486</u>	<u>0.479</u>	<u>0.439</u>	<u>0.464</u>

Table A.6

Accuracy results on Places-LT for two data ordering methods IID and Class IID. Accuracy scores are averaged over three runs with different data permutations. Results are highlighted as follows for the first, second, and third best results.

Method	IID					
	MNet-S	MNet-L	ENet-B0	ENet-B1	RN-18	Mean
Perceptron	0.152	0.185	0.213	0.206	0.149	0.181
Fine-Tune	0.136	0.163	0.197	0.191	0.141	0.166
Naive Bayes	0.015	0.050	0.199	0.213	0.100	0.115
SOvR	0.089	0.149	0.262	0.245	0.146	0.178
NCM	0.265	0.309	0.336	<u>0.329</u>	<u>0.300</u>	0.308
Replay	0.239	0.267	0.290	0.282	0.223	0.260
SLDA	<u>0.290</u>	<u>0.318</u>	<u>0.338</u>	0.328	<u>0.300</u>	<u>0.315</u>
ExLL-M	<u>0.356</u>	<u>0.392</u>	<u>0.407</u>	<u>0.400</u>	<u>0.360</u>	<u>0.383</u>
ExLL-P	0.265	0.297	0.315	0.305	0.277	0.292
ExLL-F	<u>0.324</u>	<u>0.349</u>	<u>0.362</u>	<u>0.351</u>	<u>0.331</u>	<u>0.343</u>

Method	Class-IID					
	MNet-S	MNet-L	ENet-B0	ENet-B1	RN-18	Mean
Perceptron	0.017	0.028	0.071	0.073	0.015	0.041
Fine-Tune	0.015	0.021	0.071	0.075	0.004	0.037
Naive Bayes	0.015	0.050	0.199	0.213	0.001	0.096
SOvR	0.089	0.149	0.262	0.245	0.146	0.178
NCM	0.265	0.309	0.336	<u>0.329</u>	<u>0.300</u>	0.308
Replay	0.241	0.268	0.306	0.295	0.193	0.261
SLDA	<u>0.290</u>	<u>0.319</u>	<u>0.338</u>	0.328	<u>0.300</u>	<u>0.315</u>
ExLL-M	<u>0.311</u>	<u>0.331</u>	<u>0.338</u>	<u>0.331</u>	<u>0.345</u>	<u>0.331</u>
ExLL-P	0.265	0.300	0.314	0.303	0.273	0.291
ExLL-F	<u>0.325</u>	<u>0.351</u>	<u>0.359</u>	<u>0.350</u>	<u>0.330</u>	<u>0.343</u>

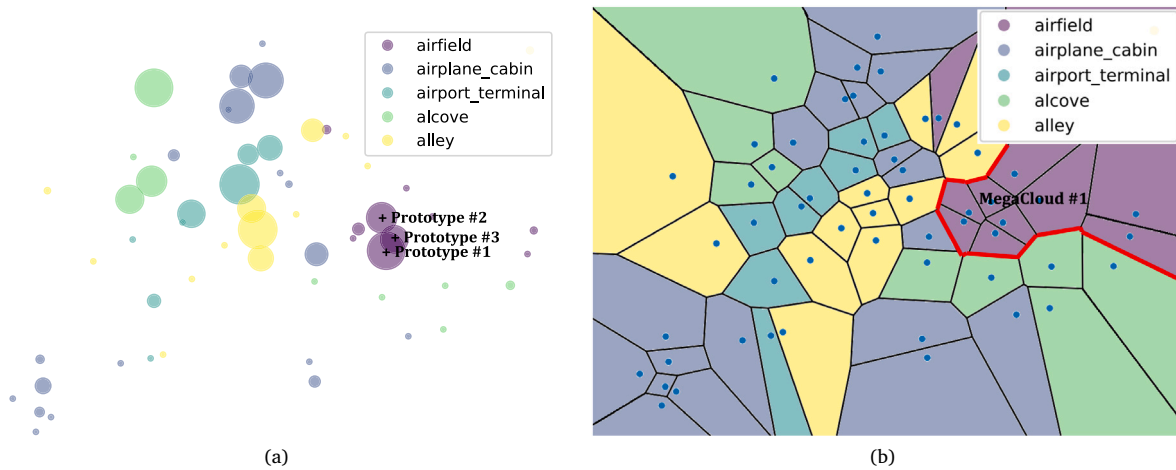
Table B.7

Accuracy results on F-SIOL-310 using Class IID data ordering for 5-shot learning and 10-shot learning scenarios. Accuracy scores are averaged over three runs with different data permutations. Results are highlighted as follows for the first, second, and third best results.

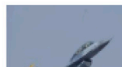
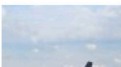
Method	5-Shot					
	MNet-S	MNet-L	ENet-B0	ENet-B1	RN-18	Mean
Pcpt	0.181	0.177	0.406	0.454	0.049	0.253
FT	0.183	0.205	0.416	0.460	0.090	0.270
NB	0.344	0.554	0.816	0.828	0.035	0.515
SOvR	0.592	0.666	0.679	0.693	0.428	0.611
NCM	<u>0.853</u>	0.871	<u>0.886</u>	0.885	0.885	<u>0.876</u>
Replay	0.541	0.632	0.594	0.612	0.624	0.600
SLDA	0.880	0.899	<u>0.912</u>	<u>0.903</u>	0.854	0.889
ExLL-M	0.842	0.873	0.863	0.847	0.851	0.855
ExLL-P	0.827	0.832	0.799	0.755	0.803	0.803
ExLL-F	0.889	<u>0.905</u>	0.887	<u>0.854</u>	0.885	0.884

Method	10-Shot					
	MNet-S	MNet-L	ENet-B0	ENet-B1	RN-18	Mean
Pcpt	0.158	0.223	0.354	0.458	0.051	0.248
FT	0.127	0.199	0.389	0.453	0.090	0.251
NB	0.320	0.537	0.806	0.854	0.015	0.506
SOvR	0.561	0.702	0.650	0.752	0.504	0.633
NCM	0.883	0.906	0.893	0.913	0.896	0.898
Replay	0.625	0.694	0.714	0.722	0.731	0.697
SLDA	0.924	0.948	0.938	0.936	0.910	<u>0.931</u>
ExLL-M	<u>0.926</u>	<u>0.942</u>	0.938	<u>0.928</u>	0.948	0.936
ExLL-P	0.927	0.934	<u>0.897</u>	0.879	<u>0.930</u>	0.913
ExLL-F	<u>0.961</u>	0.966	<u>0.952</u>	0.943	0.968	0.958

presents the top-1 accuracies achieved using the Places-LT dataset, a subset of Places-365 with uneven distribution of images across all classes, designed to evaluate a model's ability to generalize from highly imbalanced data distributions. For the IID data ordering method, ExLL-M and ExLL-F consistently outperform the other methods across all architectures with an overall mean accuracy of 0.383 and 0.343 respectively. In comparison, the third-best performer SLDA has an overall

[illegible]

(c)

MegaCloud	IF Rules						THEN Class	
1	IF		OR		OR		OR	THEN “airfield”
			OR		OR		OR	
			OR		OR		OR ...	

(d)

Fig. C.5. Example prototype topology (top left) for five image classes from the Places-365 dataset and the corresponding MegaCloud topology (top right). The “+” symbols highlight the example prototypes used for generating the prototype rule base. The red lines in the top-right sub-figure denote the boundaries for a MegaClouds consisting of the prototypes for the image class “airfield”. Rule bases are generated using training images associated with each prototype or MegaCloud to construct each IF-THEN rule. For example, MegaCloud #1 encompasses Prototypes #1, #2, and #3. Therefore, the IF-THEN rule for MegaCloud #1 is constructed using a representative image from each Prototype #1, #2, and #3.

mean accuracy of 0.315 while ExLL-P performed poorly (0.292). Under the Class IID data ordering method, the performance trends similarly highlight the robustness of the ExLL models. ExLL-F shows an overall mean accuracy of 0.343 while ExLL-M is the second-best performer with a mean accuracy of 0.331 followed by SLDA (0.315).

The results demonstrate the superiority of the ExLL variants, particularly ExLL-M and ExLL-F, in handling both IID and Class IID data ordering methods. When compared to the results in [Table A.5](#), ExLL-M showed a degraded performance under the Class IID condition, suggesting that the model is somewhat vulnerable to differences in

class distribution. ExLL-F in comparison is robust in both, IID and Class IID conditions, even though its performance is somewhat reduced compared to its performance with Places-365. Finally, ExLL-P showed the largest drop in performance under the Places-LT conditions while unaffected by the IID and Class IID conditions. In conclusion, ExLL-M and ExLL-F methods demonstrate high accuracy and robustness compared to other learners. Their ability to maintain high performance across different architectures and data ordering methods highlights their applicability and reliability in various machine learning tasks.

Appendix B. Results for F-SIOL-310

The F-SIOL-310 dataset was selected to observe how the online continual learning methods perform in low-shot continual learning applications. Table B.7 presents the performance for all continual learning methods, backbone CNNs, and learning scenarios.

In the 5-shot learning scenario, SLDA achieved the best result while ExLL-F came in second. The other two ExLL inferencing strategies meanwhile performed worse than NCM in the 5-shot learning scenario. When more training data is provided in the 10-shot scenario however, ExLL-F and ExLL-M outperformed all other methods.

Across both 5-shot and 10-shot learning scenarios, ExLL-F consistently outperforms nearly all other methods, despite having to combine the inferences from the lower-performing ExLL-M and ExLL-P. This is indicative of the pairwise fusion's ability to combine complementary inferencing strategies at the global and local level to improve the robustness of the predictions.

Appendix C. Visualization of the places-365 topology

Fig. C.5 visualizes the topology of the prototypes encoded using ExLL-F. In Fig. C.5(a), the high-dimensional centroids are transformed into 2-dimensional scatter plots using t-SNE [62]. Each circle denotes one prototype and the radius of the circle indicates the support or size of the prototype's data cloud. There are multiple instances where neighbouring prototypes displayed overlapping areas of influences. The presence of overlap suggests that some prototypes are highly similar to each other even after self-organization. In addition, there are many tiny circles which are isolated from the other prototypes. These represent prototypes of highly-distinctive images in the dataset that have little similarity with other images in the same class.

In Fig. C.5(b), the same 2-dimensional prototype topology is represented using a Voronoi tessellation graph which makes it easier to observe how MegaClouds are formed from adjacent same-class prototypes. For instance, the bright-yellow grouping in the middle-left area corresponding to the "Alley" label indicates the relative similarity of the prototypes while also showing how a few other "Alley" prototypes are located some distance away, too far to belong to the same MegaCloud. The explainability facilitated through the use of MegaClouds is more useful than by using prototypes alone. If given an image which was classified as belonging to a prototype within the central yellow MegaCloud, the image can be considered as having a high typicality or a representative image of the class. If however the image is classified as belonging to the left yellow MegaCloud, it can be considered as an atypical representation.

Data availability

Data will be made available on request.

References

- [1] S. Thrun, T.M. Mitchell, Lifelong robot learning, *Robot. Auton. Syst.* 15 (1–2) (1995) 25–46.
- [2] P. Domingos, G. Hulten, Mining high-speed data streams, in: *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000, pp. 71–80.
- [3] M.T. Ribeiro, S. Singh, C. Guestrin, 'Why should I trust you?' Explaining the predictions of any classifier, in: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144.
- [4] T. Nomura, K. Kawakami, Relationships between robot's self-disclosures and human's anxiety toward robots, in: *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, vol. 3, IEEE, 2011, pp. 66–69.
- [5] O. Biran, C. Cotton, Explanation and justification in machine learning: A survey, in: *IJCAI-17 Workshop on Explainable AI (XAI)*, vol. 8, 2017, pp. 8–13.
- [6] S. Pang, S. Ozawa, N. Kasabov, Incremental linear discriminant analysis for classification of data streams, *IEEE Trans. Syst. Man Cybern.* 35 (5) (2005) 905–914.
- [7] T. Mensink, J. Verbeek, F. Perronnin, G. Csurka, Metric learning for large scale image classification: Generalizing to new classes at near-zero cost, in: *European Conference on Computer Vision*, Springer, 2012, pp. 488–501.
- [8] T. Kulesza, M. Burnett, W.-K. Wong, S. Stumpf, Principles of explanatory debugging to personalize interactive machine learning, in: *Proceedings of the 20th International Conference on Intelligent User Interfaces*, 2015, pp. 126–137.
- [9] P. Angelov, E. Soares, Towards explainable deep neural networks (xDNN), *Neural Netw.* 130 (2020) 185–194.
- [10] T.L. Hayes, C. Kanan, Online continual learning for embedded devices, in: *Proceedings of the 1st Conference on Lifelong Learning Agents*, vol. 199, 2022, pp. 744–766.
- [11] G.A. Carpenter, S. Grossberg, J.H. Reynolds, ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network, *Neural Netw.* 4 (5) (1991) 565–588.
- [12] R. Palaniappan, C. Eswaran, Using genetic algorithm to select the presentation order of training patterns that improves simplified fuzzy ARTMAP classification performance, *Appl. Soft Comput.* 9 (1) (2009) 100–106.
- [13] M. Yaghini, M.A. Shadmani, GOFAM: A hybrid neural network classifier combining fuzzy ARTMAP and genetic algorithm, *Artif. Intell. Rev.* 39 (2013) 183–193.
- [14] W.S. Liew, M. Seera, C.K. Loo, E. Lim, Affect classification using genetic-optimized ensembles of fuzzy ARTMAPs, *Appl. Soft Comput.* 27 (2015) 53–63.
- [15] T.L. Hayes, C. Kanan, Lifelong machine learning with deep streaming linear discriminant analysis, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 220–221.
- [16] Y. Freund, R.E. Schapire, Large margin classification using the perceptron algorithm, in: *Proceedings of the 11th Annual Conference on Computational Learning Theory*, 1998, pp. 209–217.
- [17] A. Mukherjee, S. Mondal, N. Chaki, S. Khatua, Naive Bayes and decision tree classifier for streaming data using hbase, in: *Advanced Computing and Systems for Security: Volume Seven*, Springer, 2018, pp. 105–116.
- [18] T.L. Hayes, K. Kafle, R. Shrestha, M. Acharya, C. Kanan, Remind your neural network to prevent catastrophic forgetting, in: *Computer Vision: 16th European Conference, ECCV 2020*, Springer, 2020, pp. 466–483.
- [19] A. Ayub, A.R. Wagner, Cognitively-inspired model for incremental learning using a few examples, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 222–223.
- [20] A. Ayub, A.R. Wagner, F-SIOL-310: A robotic dataset and benchmark for few-shot incremental object learning, in: *2021 IEEE International Conference on Robotics and Automation, ICRA, IEEE*, 2021, pp. 13496–13502.
- [21] X. Tao, X. Chang, X. Hong, X. Wei, Y. Gong, Topology-preserving class-incremental learning, in: *Computer Vision: 16th European Conference, ECCV 2020*, Springer, 2020, pp. 254–270.
- [22] X. Tao, X. Hong, X. Chang, S. Dong, X. Wei, Y. Gong, Few-shot class-incremental learning, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12183–12192.
- [23] K. Ahrens, L. Bengtson, J.H. Lee, S. Wermter, Visually grounded continual language learning with selective specialization, 2023, arXiv preprint [arXiv:2310.15571](https://arxiv.org/abs/2310.15571).
- [24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, 2020, arXiv preprint [arXiv:2010.11929](https://arxiv.org/abs/2010.11929).
- [25] Z. Wang, Z. Zhang, C.-Y. Lee, H. Zhang, R. Sun, X. Ren, G. Su, V. Perot, J. Dy, T. Pfister, Learning to prompt for continual learning, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 139–149.

- [26] Z. Wang, Z. Zhang, S. Ebrahimi, R. Sun, H. Zhang, C.-Y. Lee, X. Ren, G. Su, V. Perot, J. Dy, et al., Dualprompt: Complementary prompting for rehearsal-free continual learning, in: *European Conference on Computer Vision*, Springer, 2022, pp. 631–648.
- [27] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, MobileNets: Efficient convolutional neural networks for mobile vision applications, 2017, arXiv preprint [arXiv:1704.04861](https://arxiv.org/abs/1704.04861).
- [28] F.N. Iandola, S. Han, M.W. Moskewicz, K. Ashraf, W.J. Dally, K. Keutzer, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size, 2016, arXiv preprint [arXiv:1602.07360](https://arxiv.org/abs/1602.07360).
- [29] M. Tan, Q. Le, EfficientNet: Rethinking model scaling for convolutional neural networks, in: *International Conference on Machine Learning*, PMLR, 2019, pp. 6105–6114.
- [30] A.A. Monrat, R.U. Islam, M.S. Hossain, K. Andersson, A belief rule based flood risk assessment expert system using real time sensor data streaming, in: *2018 IEEE 43rd Conference on Local Computer Networks Workshops (LCN Workshops)*, IEEE, 2018, pp. 38–45.
- [31] X. Gu, P.P. Angelov, Q. Shen, Self-organizing fuzzy belief inference system for classification, *IEEE Trans. Fuzzy Syst.* 30 (12) (2022) 5473–5483.
- [32] A. Almeida, S. Brás, S. Sargento, F.C. Pinto, Time series big data: a survey on data stream frameworks, analysis and algorithms, *J. Big Data* 10 (1) (2023) 83.
- [33] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba, Learning deep features for discriminative localization, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929.
- [34] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-CAM: Visual explanations from deep networks via gradient-based localization, in: *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 618–626.
- [35] G.I. Parisi, J. Tani, C. Weber, S. Wermter, Lifelong learning of spatiotemporal representations with dual-memory recurrent self-organization, *Front. Neurobotics* (2018) 78.
- [36] M. Biehl, B. Hammer, T. Villmann, Prototype-based models in machine learning, *Wiley Interdiscip. Rev.: Cogn. Sci.* 7 (2) (2016) 92–111.
- [37] P.P. Angelov, X. Gu, Empirical approach to machine learning, Springer, 2019.
- [38] A.H. Ko, R. Sabourin, A.D. Britto Jr., L. Oliveira, Pairwise fusion matrix for combining classifiers, *Pattern Recognit.* 40 (8) (2007) 2198–2210.
- [39] E. Soares, P. Angelov, Novelty detection and learning from extremely weak supervision, 2019, arXiv preprint [arXiv:1911.00616](https://arxiv.org/abs/1911.00616).
- [40] P. Angelov, Autonomous learning systems: From data streams to knowledge in real-time, John Wiley & Sons, 2012.
- [41] G.J. McLachlan, Mahalanobis distance, *Resonance* 4 (6) (1999) 20–26.
- [42] M. Herchenbach, D. Müller, S. Scheele, U. Schmid, Explaining image classifications with near misses, near hits and prototypes: Supporting domain experts in understanding decision boundaries, in: *Pattern Recognition and Artificial Intelligence: Third International Conference, ICPRAI 2022*, Springer, 2022, pp. 419–430.
- [43] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, F.-F. Li, ImageNet Large Scale Visual Recognition Challenge, *Int. J. Comput. Vis.* 115 (2015) 211–252.
- [44] Q. She, F. Feng, X. Hao, Q. Yang, C. Lan, V. Lomonaco, X. Shi, Z. Wang, Y. Guo, Y. Zhang, F. Qiao, R.H.M. Chan, OpenLORIS-Object: A robotic vision dataset and benchmark for lifelong deep learning, in: *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 4767–4773.
- [45] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, A. Torralba, Places: A 10 million image database for scene recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* 40 (6) (2017) 1452–1464.
- [46] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, 2013, arXiv preprint [arXiv:1312.5602](https://arxiv.org/abs/1312.5602).
- [47] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q.V. Le, A. Hartwig, Searching for MobileNetV3, in: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1314–1324.
- [48] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, MobileNetV2: Inverted residuals and linear bottlenecks, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.
- [49] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
- [50] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, C.H. Lampert, ICaRL: Incremental classifier and representation learning, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.
- [51] F.M. Castro, M.J. Marín-Jiménez, N. Guil, C. Schmid, K. Alahari, End-to-end incremental learning, in: *Computer Vision: 15th European Conference, ECCV 2018*, 2018, pp. 233–248.
- [52] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, Y. Fu, Large scale incremental learning, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 374–382.
- [53] A. Douillard, M. Cord, C. Ollion, T. Robert, E. Valle, PODNet: Pooled outputs distillation for small-tasks incremental learning, in: *Computer Vision: 16th European Conference, ECCV 2020*, Springer, 2020, pp. 86–102.
- [54] A. Wong, NetScore: Towards universal metrics for large-scale performance analysis of deep neural networks for practical on-device edge usage, in: *Image Analysis and Recognition: 16th International Conference, ICIAR 2019*, Springer, 2019, pp. 15–26.
- [55] U. Michieli, M. Ozay, Online continual learning for robust indoor object recognition, in: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2023, pp. 3849–3856.
- [56] W.S. Liew, C.K. Loo, V. Gryshchuk, C. Weber, S. Wermter, Effect of pruning on catastrophic forgetting in growing dual memory networks, in: *2019 International Joint Conference on Neural Networks, IJCNN*, IEEE, 2019, pp. 1–8.
- [57] C. Wiwatcharakoses, D. Berrar, SOINN+, a self-organizing incremental neural network for unsupervised learning from noisy data streams, *Expert Syst. Appl.* 143 (2020) 113069.
- [58] A. Logaciov, M. Kerzel, S. Wermter, Learning then, learning now, and every second in between: lifelong learning with a simulated humanoid robot, *Front. Neurobotics* 15 (2021) 669534.
- [59] A.H. Gee, D. Garcia-Olano, J. Ghosh, D. Paydarfar, Explaining deep classification of time-series data with learned prototypes, in: *CEUR Workshop Proceedings*, vol. 2429, NIH Public Access, 2019, p. 15.
- [60] E.M. Kenny, M.T. Keane, Explaining deep learning using examples: Optimal feature weighting methods for twin systems using post-hoc, explanation-by-example in XAI, *Knowl.-Based Syst.* 233 (2021) 107530.
- [61] L. Peng, H. Wang, J. Li, Uncertainty evaluation of object detection algorithms for autonomous vehicles, *Automot. Innov.* 4 (3) (2021) 241–252.
- [62] L. Van der Maaten, G. Hinton, Visualizing data using t-SNE, *J. Mach. Learn. Res.* 9 (11) (2008) 2579–2605.