Cognitively Inspired Learning of Incremental Drifting Concepts

Mohammad Rostami, **Aram Galstyan**
University of Southern California
{mrostami,galstyan}@isi.edu

Abstract

Humans continually expand their learned knowledge to new domains and learn new concepts without any interference with past learned experiences. In contrast, machine learning models perform poorly in a continual learning setting, where input data distribution changes over time. Inspired by the nervous system learning mechanisms, we develop a computational model that enables a deep neural network to learn new concepts and expand its learned knowledge to new domains incrementally in a continual learning setting. We rely on the *Parallel Distributed Processing* theory to encode abstract concepts in an embedding space in terms of a multimodal distribution. This embedding space is modeled by internal data representations in a hidden network layer. We also leverage the *Complementary Learning Systems theory* to equip the model with a memory mechanism to overcome catastrophic forgetting through implementing pseudo-rehearsal. Our model can generate pseudo-data points for experience replay and accumulate new experiences to past learned experiences without causing cross-task interference.

1 Introduction

Humans continually abstract *concept classes* from their input sensory data to build semantic descriptions, and then update and expand these concepts as more experiences are accumulated [Widmer and Kubat, 1996], and use them to express their ideas and communicate with each other [Gennari *et al.*, 1989; Lake *et al.*, 2015]. For example, "cat" and "dog" are one of the first concept classes that many children learn to identify. Most humans expand these concepts as *concept drift* occurs, e.g., incorporating many atypical dog breeds into the "dog" concept, and also incrementally learn new concept classes, e.g. "horse" and "sheep," as they acquire more experiences. Although this concept learning procedure occurs continually in humans, continual and incremental learning of concept classes remains a major challenge in artificial intelligence (AI). AI models are usually trained on a fixed number of classes and the data distribution is assumed to be stationary during model execution. Hence, when an AI model is trained or updated on sequentially observed tasks with diverse distributions or is trained on new classes, we generally need new annotated data points from the new classes [Rostami *et al.*, 2018] and the model also would tend to forget what has been learned before due to cross-task interference, known as the phenomenon of *catastrophic forgetting* [French, 1991].

Inspired by the Parallel Distributed Processing (PDP) paradigm [McClelland *et al.*, 1986; McClelland and Rogers, 2003], our goal is to enable a deep neural network to learn *drifting concept classes* [Gama *et al.*, 2014; Rostami and Galstyan, 2023] incrementally and continually in a sequential learning setting. PDP hypothesizes that abstract concepts are encoded in higher layers of the nervous system [McClelland and Rogers, 2003; Saxe *et al.*, 2019]. Similarly, and based on behavioral similarities between artificial deep neural networks and the nervous system [Morgenstern *et al.*, 2014] , we can assume that the data representations in hidden layers of a deep network encode semantic concepts with different levels of abstractions. We model these representations as an embedding space in which semantic similarities between input data points are encoded in terms of geometric distances [Jiang and Conrath, 1997], i.e., data points that belong to the same concept class are mapped into separable clusters in the embedding space. When a new concept is abstracted, a new distinct cluster should be formed in the embedding space to encode that new class. Incremental concepts learning is feasible by tracking and remembering the representation clusters that are formed in the embedding space and by considering their dynamics as more experiences are accumulated in new domains.

We benefit from the Complementary Learning Systems (CLS) theory [McClelland *et al.*, 1995] to mitigate catastrophic forgetting. CLS is based on empirical evidences that suggest experience replay of recently observed patterns during sleeping and waking periods in the human brain helps to accumulate the new experiences to the past learned experiences without causing interference [McClelland *et al.*, 1995; Robins, 1995]. According to this theory, hippocampus plays the role of a short-term memory buffer that stores samples of recent experiences and catastrophic forgetting is prevented by replaying samples from the hippocampal storage to implement pseudo-rehearsal in the neocortex during sleeping periods through enhancing past learned knowledge. Unlike AI memory buffers that store raw input data point, hippocampal storage can only store encoded abstract representations.

Inspired by the above two theories, we expand a base neural classifier with a decoder network, which is amended from a hidden layer, to form an autoencoder with the hidden layer as its bottleneck. The bottleneck is used to model the discriminative embedding space. As a result of supervised learning, the embedding space becomes discriminative, i.e. a data cluster is formed for each concept class in the embedding space [McClelland and Rogers, 2003; Rostami, 2021b]. These clusters can be considered analogous to neocortical representations in the brain, where the learned abstract concepts are encoded [McClelland *et al.*, 1986]. We use a multi-modal distribution to estimate this distribution [Stan and Rostami, 2021; Rostami, 2021a]. We update this parametric distribution to accumulate new experiences to past learned experiences consistently. Since our model is generative, we can implement the offline memory replay process to prevent catastrophic forgetting [McClelland *et al.*, 1995; Rasch and Born, 2013]. When a new task arrives, we draw random samples from the multi-modal distribution and feed them into the decoder to generate representative pseudo-data points. These pseudo-data points are then used to implement pseudo-rehearsal for experience replay [Robins, 1995].

2 Related Work

Continual learning: the major challenge of continual learning is tackling catastrophic forgetting. Previous works in the literature mainly rely on experience replay [Li and Hoiem, 2018]. The core idea of experience replay is to implement pseudo-rehearsal by replaying representative samples of past tasks along with the current task data to retain the learned distributions. Since storing these samples requires a memory buffer, the challenge is selecting the representative samples to meet the buffer size limit. For example, selecting uncommon samples that led to maximum effect in past experiences has been found to be effective [Schaul *et al.*, 2016]. However, as more tasks are learned, selecting the effective samples becomes more complex. The alternative approach is to use generative models that behave more similar to humans [French, 1999]. Shin et al. ([Shin *et al.*, 2017]) use a generative adversarial structure to mix the distributions of all tasks. It is also feasible to couple the distributions of all tasks in the bottleneck of an autoencoder [Rostami *et al.*, 2019; Rostami *et al.*, 2020b]. The shared distribution then can be used to generate pseudo-samples [Rannen *et al.*, 2017].Weight consolidation using structural plasticity [Lamprecht and LeDoux, 2004; Zenke *et al.*, 2017; Kirkpatrick *et al.*, 2017] is another approach to approximate experience replay. The idea is to identify important weights that retain knowledge about a task and then consolidate them according to their relative importance for past tasks. Continual learning of sequential tasks can be improved used high-level tasks descriptors to compensate for data scarcity [Rostami *et al.*, 2020a].

Incremental learning: forgetting in *incremental learning*stems from updating the model when new classes are incorporated, rather concept drifts in a fixed number of learned classes. Hence, the goal is to learn new classes such that knowledge about the past learned classes is not overwritten. A simple approach is to expand the base network as new classes are observed. Tree-CNN [Roy *et al.*, 2020] proposes a hierarchical structure that grows like a tree when new classes are observed. The idea is to group new classes into feature-driven super-classes and find the exact label by limiting the search space. As the network grows, the new data can be used to train the expanded network. Sarwar et al. [Sarwar *et al.*, 2019] add new convolutional filters in all layers to learn the new classes through new parameters. The alternative approach is to retain the knowledge about old classes in an embedding feature space. Rebuffi et al. [Rebuffi *et al.*, 2017] proposed iCarl which maps images into a feature space that remains discriminative as more classes are learned incrementally. A fixed memory buffer is used to store exemplar images for each observed class. Each time a new class is observed, these images are used to learn a class-level vector in the feature space such that the testing images can be classified using nearest neighbor with respect to these vectors.

Gaussian mixture model : are useful for modeling distributions that exhibit multiple modes or clusters. GMMs assume that the data is generated by a mixture of several Gaussian distributions, each representing a different cluster or mode in the data. The model is trained by estimating the parameters of the component Gaussians, including their means and variances, as well as the mixture weights that determine the relative contribution of each Gaussian to the overall distribution. GMMs are widely used in a variety of applications, including continual learning [Rostami *et al.*, 2019].

Contributions: We develop a unified framework that addresses challenges of both incremental learning and lifelong learning. Our idea is based on tracking and consolidating the multimodal distribution that is formed by the internal data representations of sequential tasks in hidden layers of a neural network. We model this distribution as a Gaussian mixture model (GMM) with time-dependent number of components. Concept drifts are learned by updating the corresponding GMM component for a particular class and new concepts are learned by adding new GMM components. We also make the model generative to implement experience replay.

3 Problem Statement

Consider a learning agent which observes a sequence of observed tasks $\{\mathcal{Z}^{(t)}\}_{t=1}^{T}$ [Chen and Liu, 2016] and after learning each task moves forward to learn the next task. Each task is a classification problem in a particular domain and each class represents a concept. The classes for each task can be new unobserved classes, i.e., necessitating incremental learning [Rebuffi *et al.*, 2017], or drifted forms of the past learned classes, i.e., necessitating lifelong learning [Chen and Liu, 2016], or potentially a mixture of both cases. Formally, a task is characterized by a dataset $\mathcal{D}^{(t)} = \langle \boldsymbol{X}^{(t)}, \boldsymbol{Y}^{(t)} \rangle$, where $\boldsymbol{X}^{(t)} = [\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_n^t] \in \mathbb{R}^{d \times n_t}$ and $\boldsymbol{Y}^{(t)} \in \mathbb{R}^{k_t \times n_t}$ are the data points and one-hot labels, respectively. The goal is to train a time-dependent classifier function $f^{(t)}(\cdot) : \mathbb{R}^d \to\subset \mathbb{R}^{k_t}$- where k_t is the number of classes for the t-th task and is fixed for each task- such that the classifier continually generalizes on the tasks seen so far. The data points $\boldsymbol{x}_i^{(t)} \sim q^{(t)}(\boldsymbol{x})$ are assumed to be drawn i.i.d. from an unknown task distri-

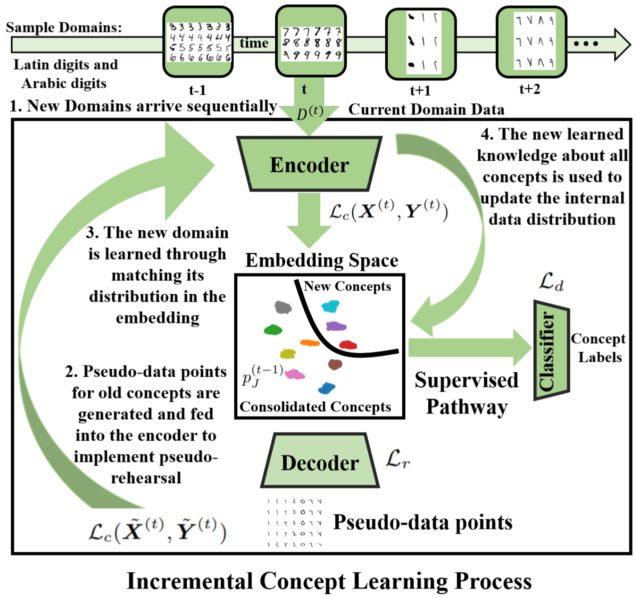


Figure 1: Block-diagram visualization of the proposed Incremental Learning System. (Best viewed enlarged on screen and in color. Enlarged version is included in the Appendix)

bution $q^{(t)}(\boldsymbol{x})$. Figure 1 visualizes a block-diagram of this continual and dynamic learning procedure. The agent needs to expand its knowledge about all the observed concepts such that it can perform well on all the previous learned domains.

Learning each task in isolation is a standard supervised learning problem. After selecting a suitable parameterized family of functions $f_\theta^{(t)} : \mathbb{R}^d \to \mathbb{R}^{k_t}$ with learnable parameters θ, e.g. a deep neural network with learnable weight paramters θ, we can solve for the optimal parameters using the empirical risk minimization (ERM): $\hat{\theta}^{(t)} = \arg\min_\theta \hat{e}_\theta^{(t)} = \arg\min_\theta \sum_i \mathcal{L}_d(f_\theta^{(t)}(\boldsymbol{x}_i^{(t)}), \boldsymbol{y}_i^{(t)})$, where $\mathcal{L}_d(\cdot)$ is a proper loss function. If n_t is large enough, the empirical risk expectation would be a good approximation of the real expected risk function $e^{(t)}(\theta) = \mathbb{E}_{\boldsymbol{x}\sim q^{(t)}(\boldsymbol{x})}(\mathcal{L}_d(f_{\theta^{(t)}}(\boldsymbol{x}), f(\boldsymbol{x})))$. As a result, if the base parametric family is rich and complex enough for learning the task function, then the ERM optimal model generalizes well on unseen test samples that are drawn from $q^{(t)}(\boldsymbol{x})$.

For the rest of the paper, we consider the base model $f_{\theta^{(t)}}$ to be a deep neural network with an increasing output size to encode incrementally observed classes. As stated, we rely on the PDP paradigm. Hence, we decompose the deep network into an encoder sub-network $\phi_{\boldsymbol{v}}(\cdot) : \mathbb{R}^d \to \mathcal{Z} \subset \mathbb{R}^f$ with learnable parameter $\boldsymbol{v}$, e.g., convolutional layers of a CNN, and a classifier sub-network $h_{\boldsymbol{w}}(\cdot)^{k_t} : \mathbb{R}^f \to \mathbb{R}^{k_t}$ with learnable parameters $\boldsymbol{w}$, e.g., fully connected layers of a CNN, where $\mathcal{Z}$ denotes the embedding space in which the concepts will be be formed as separable clusters.

The concepts for each task are known a priori and hence new nodes are added to the classifier sub-network output to incorporate the new classes at time t. We use a softmax layer as the last layer of the classifier subnetwork. Hence, we can consider the classifier to be a a maximum *a posteriori* (MAP) estimator after training. This means that the encoder network transforms the input data distribution into an internal multi-modal distribution with k_t modes in the embedding space because the embedding space $\mathcal{Z}$ should be concept-discriminative for good generalization. Each concept class is represented by a single mode of this distribution. We use a Gaussian mixture model (GMM) to model and approximate this distribution (see Figure 1, middle panel). Catastrophic forgetting is the result of changes in this internal distribution when changes in the input distribution leads to updating the internal distribution heuristically. Our idea is to track changes in the data distribution and update and consolidate the internal distribution such that the acquired knowledge from past experiences is not overwritten when learning new tasks.

The main challenge is to adapt the network $f_\theta^{(t)}(\cdot)$ and the standard ERM such that we can track the internal distribution continually and accumulate the new acquired knowledge consistently to the past learned knowledge with minimum interference. For this purpose, we form a generative model by amending the base model with a decoder $\psi_{\boldsymbol{u}} : \mathcal{Z} \to \mathbb{R}^d$, with learnable parameters $\boldsymbol{u}$. This decoder maps back the internal representations to reconstruct the input data point in the input space such that the pair $(\phi_{\boldsymbol{u}}, \psi_{\boldsymbol{u}})$ forms an autoencoder. According to our previous discussion, a multi-modal distribution would be formed in the bottleneck of the autoencoder upon learning each task. This distribution encodes the learned knowledge about the concepts that have been learned from past experiences so far. If we approximate this distribution with a GMM, we can generate pseudo-data points that represent the previously learned concepts and use them for pseudo-rehearsal. For this purpose, we can simply draw samples from all modes of the GMM and feed these samples into the decoder subnetwork to generate a pseudo-dataset (see Figure 1). After learning each task, we can update the GMM estimate such that the new knowledge acquired is accumulated to the past gained knowledge consistently to avoid interference. By doing this procedure continually, our model is able to learn drifting concepts incrementally. Figure 1 visualizes this repetitive procedure in our setting.

4 Proposed Algorithm

When the first task is learned, there is no prior experience and hence learning reduces the following:

$$\min_{\boldsymbol{v},\boldsymbol{w},\boldsymbol{u}} \mathcal{L}_c(\boldsymbol{X}^{(1)}, \boldsymbol{Y}^{(1)}) = \min_{\boldsymbol{v},\boldsymbol{w},\boldsymbol{u}} \frac{1}{n_1} \sum_{i=1}^{n_1} \Bigg(\mathcal{L}_d\Big(h_{\boldsymbol{w}}(\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(1)})), \boldsymbol{y}_i^{(1)}\Big) + \gamma \mathcal{L}_r\Big(\psi_{\boldsymbol{u}}(\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(1)})), \boldsymbol{x}_i^{(1)}\Big) \Bigg), \tag{1}$$

where $\mathcal{L}_d$ is the discrimination loss, e.g., cross-entropy loss, $\mathcal{L}_r$ is the reconstruction loss for the autoencoder, e.g., ℓ_2-norm, $\mathcal{L}_c$ is the combined loss, and γ is a trade-off parameter between the terms. When the first task is learned, also any future task, according to the PDP hypothesis, a multi-modal distribution $p^{(1)}(\boldsymbol{z}) = \sum_{j=1}^{k_1} \alpha_j \mathcal{N}(\boldsymbol{Z}|\mu_j, \Sigma_j)$ with k_1 components is formed in the embedding space. We assume that this distribution can be modeled with a GMM. Since the labels for the input task data samples are known, we use MAP estimation to recover the GMM parameters (see Appendix for details). Let $\hat{p}^{(1)}(\boldsymbol{z})$ denotes the estimated distribution.

As subsequent tasks are learned, the internal distribution should be updated continually to accumulate the new acquired knowledge. Let $k_t = k_{old}^t + k_{new}^t$, where k_{old}^t denotes the number of the previously learned concepts that exist in the current task and k_{new}^t denotes the number of the new observed classes. Hence, the total number of learned concepts until $t = T$ is $k_{Tot}^T = \sum_{t=1}^T k_{new}^t$. Also, let the index set $\mathbb{N}_{Tot}^T = \{1, \ldots, k_{Tot}^T\}$ denotes an order on the classes C_j, with $j \in \mathbb{N}_{Tot}^T$, that are observed until $t = T$. Let $\mathbb{N}_T = \mathbb{N}_{old}^T \cup \mathbb{N}_{new}^T = \{i_1, \ldots, i_{k_T}\} \subset \mathbb{N}_{Tot}^T$ contains the k_T indices of the existing concepts in $\mathcal{Z}^{(T)}$. To update the internal distribution after learning $\mathcal{Z}^{(T)}$, the number of distribution modes should be updated to k_{Tot}^T. Additionally, catastrophic forgetting must be mitigated using experience replay. We can draw random samples from the GMM distribution $\boldsymbol{z}_i \sim \hat{p}^{(T-1)}(\boldsymbol{z})$ and then pass each sample through the decoder $\psi(\boldsymbol{z}_i)$ to generate pseudo-data points for pseudo-rehearsal. Since each particular concept is represented by exactly one mode of the internal GMM distribution, the corresponding pseudo-labels for the generated pseudo-data points are known. Moreover, the confidence levels for these labels are also known from the classifier softmax layer. To generate a clean pseudo-dataset, we can set a threshold τ and only pick the pseudo-data points for which the model confidence level is more than τ. We also generate a balanced pseudo-dataset with respect to the learned classes. Doing so, we ensure suitability of a GMM with k_{Tot}^T components to estimate the empirical distribution accurately after learning the next tasks.

Let $\tilde{\mathcal{D}}^{(t)} = \langle \psi(\tilde{\boldsymbol{Z}}^{(t)}), \tilde{\boldsymbol{Y}}^{(t)} \rangle$ denotes the pseudo-dataset, generated at time t after learning the tasks $\{\mathcal{Z}^{(s)}\}_{s=1}^{t-1}$. We form the following objective to learn the task $\mathcal{Z}^{(t)}, \forall t \geq 2$:

$$\begin{aligned} &\min_{\boldsymbol{v},\boldsymbol{w},\boldsymbol{u}} \mathcal{L}_c(\boldsymbol{X}^{(t)}, \boldsymbol{Y}^{(t)}) + \mathcal{L}_c(\tilde{\boldsymbol{X}}^{(t)}, \tilde{\boldsymbol{Y}}^{(t)}) + \\ &\lambda \sum_{j \in \mathbb{N}_{old}^t} D\Big(\phi_{\boldsymbol{v}}(q^{(t)}(\boldsymbol{X}^{(t)})|C_j), \hat{p}^{(t-1)}(\tilde{\boldsymbol{Z}}^{(t)})|C_j)\Big), \end{aligned} \tag{2}$$

where $D(\cdot,\cdot)$ is a probability metric and λ is a parameter.

The first and the second terms in Eq. (2) are combined loss terms for the current task training dataset and the generated pseudo-dataset that represent the past tasks, defined similar to Eq. (1). The second term in Eq. (2) mitigates catastrophic forgetting through pseudo-rehearsal process. The third term is a crucial term to guarantee that our method will work in a lifelong learning setting. This term enforces that each concept is encoded in one mode of the internal distribution across all tasks. This term is computed on the subset of the concept classes that are shared between the current task and the pseudo-dataset, i.e, $\mathbb{N}_{old}^t$, to enforce consistent knowledge accumulation. Minimizing the probability metric $D(\cdot,\cdot)$ enforces that the internal conditional distribution for the current task $\phi_{\boldsymbol{v}}(q^{(t)}(\cdot|C_j))$, conditioned on a particular shared concept C_j, to be close to the conditional shared distribution $p^{(t-1)}(\cdot|C_j)$. Hence, both form a single mode of the internal distribution and concept drifting is mitigated. Conditional

Algorithm 1 ICLA (λ, γ, τ)

Input: labeled training datasets in a sequence
  $\mathcal{D}^{(t)} = (\{\boldsymbol{X}^{(t)}, \boldsymbol{X}^{(t)})\}$ for $t = \geq 1$
Initial Learning: learn the first task via Eq. (1)
Fitting GMM:
  estimate $\hat{p}_J^{(1)}(\cdot)$ using $\{\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(1)}), \boldsymbol{y}_i^{(1)}\}_{i=1}^{n_t}$
For $t \geq 2$
  Generate the pseudo dataset:
    $\tilde{\mathcal{D}}^{(t)} = \{(\tilde{\boldsymbol{x}}_i^{(t)} = \psi(\tilde{\boldsymbol{z}}_i^{(t)}), \tilde{\boldsymbol{y}}_i^{(t)})\}$
    $(\tilde{\boldsymbol{z}}_i^{(t)}, \tilde{\boldsymbol{y}}_i^{(t)}) \sim \hat{p}^{(t-1)}(\cdot)$
  Task learning:
    learnable parameters are updated via Eq. (2)
  Estimating the internal distribution:
    update $\hat{p}^{(t)}(\cdot)$ with $k_{Tot}^{(t)}$ components via the
    combined samples $\{\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(t)}), \phi_{\boldsymbol{v}}(\tilde{\boldsymbol{x}}_i^{(t)})\}_{i=1}^{n_t}$
EndFor

matching of the two distributions is feasible as we have access to pseudo-labels. Adding this term guarantees that we can continually use a GMM with exactly $k_{Tot}^{(t)}$ components to capture the internal distribution in this lifelong learning setting. The remaining task is to select a suitable probability metric $D(\cdot,\cdot)$ for solving Eq. (2). Wasserstein Distance (WD) metric has been found to be an effective choice for deep learning due to its applicability for gradient-based optimization [Courty *et al.*, 2017]. To reduce the computational burden of computing WD, we use the Sliced Wasserstein Distance (SWD) [Bonneel *et al.*, 2015]. (for details on the SWD, refer to the Appendix). Our Incremental Concept Learning Algorithm (ICLA) method is summarized in Algorithm 1.

5 Theoretical Analysis

We demonstrate that ICLA minimizes an upperbound for the expected risk of the learned concept classes across all the previous tasks for all t. We perform our analysis in the embedding space as an input space and consider the hypothesis class $\mathcal{H} = \{h_{\boldsymbol{w}}(\cdot) | h_{\boldsymbol{w}}(\cdot) : \mathcal{Z} \to \mathbb{R}_t^k, \boldsymbol{w} \in \mathbb{R}^H\}$. Let $e_t(\boldsymbol{w})$ denote the real risk for a given function $h_{\boldsymbol{w}^{(t)}}(\cdot) \in \mathcal{H}$ when used on task $\mathcal{Z}^{(t)}$ data representations in the embedding space. Similarly, $\tilde{e}_t(\boldsymbol{w})$ denotes the observed risk for the function $h_{\boldsymbol{w}^{(t)}}(\cdot)$ when used on the pseudo-task, generated by sampling the learned GMM distribution $\hat{p}^{(t-1)}$. Finally, let $e_{t,s}(\boldsymbol{w})$ denote the risk of the model $h_{bmw^{(t)}}(\cdot)$ when used only on the concept classes in the set $\mathbb{N}_s \subset \mathbb{N}_{Tot}^t$, for $\forall s \leq t$, i.e., task specific classes, after learning the task $\mathcal{Z}^{(t)}$.

Theorem 1 : Consider two tasks $\mathcal{Z}^{(t)}$ and $\mathcal{Z}^{(s)}$, where $s \leq t$. Let $h_{\boldsymbol{w}^{(t)}}$ be an optimal classifier trained for the $\mathcal{Z}^{(t)}$ using the ICLA algorithm. Then for any $d' > d$ and $\zeta < \sqrt{2}$, there exists a constant number N_0 depending on d' such that for any $\xi > 0$ and $\min(\tilde{n}_{t|\mathbb{N}_s}, n_s) \geq \max(\xi^{-(d'+2),1})$ with probability at least $1 - \xi$ for $h_{\boldsymbol{w}^{(t)}} \in \mathcal{H}$, then:

$$\begin{aligned} e_s(\boldsymbol{w}) \leq & e_{t-1,s}(\boldsymbol{w}) + W(\hat{p}_s^{(t-1)}, \phi(\hat{q}^{(s)})) + e_{\mathcal{C}}(\boldsymbol{w}^*) + \\ & \sqrt{\big(2\log(\frac{1}{\xi})/\zeta\big)}\Big(\sqrt{\frac{1}{\tilde{n}_{t|\mathbb{N}_s}}} + \sqrt{\frac{1}{n_s}}\Big), \end{aligned} \tag{3}$$

where $W(\cdot,\cdot)$ denotes the WD metric, $\tilde{n}_{t|\mathbb{N}_s}$ denotes the pseudo-task samples that belong to the classes in $\mathbb{N}_s$, $\phi(\hat{q}^{(s)}(\cdot))$ denotes the empirical marginal distribution for $\mathcal{Z}^{(s)}$ in the embedding, $\hat{p}_s^{(t-1)}$ is the conditional empirical shared distribution when the distribution $\hat{p}^{(t-1)}(\cdot)$ is conditioned to the classes in $\mathbb{N}_s$, and $e_{\mathcal{C}}(\boldsymbol{w}^*)$ denotes the optimal model learned for the combined risk of the tasks on the shared classes in $\mathbb{N}_s$, i.e., $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} e_{\mathcal{C}}(\theta) = \arg\min_{\boldsymbol{w}}\{e_{t,s}(\boldsymbol{w}) + e_s(\boldsymbol{w})\}$. This is a model with the best performance if the tasks could be learned simultaneously.

Proof: included in the Appendix due to page limit.

We then use Theorem 1 to conclude the following lemma:

Lemma 1 : Consider the ICLA algorithm after learning $\mathcal{Z}^{(T)}$. Then all tasks $t < T$ and under the conditions of Theorem 1, we can conclude the following inequality:

$$e_t(\boldsymbol{w}) \leq e_{T-1,t}(\boldsymbol{w}) + W(\phi(\hat{q}^{(t)}), \hat{p}_t^{(t)}) + e_{\mathcal{C}}(\boldsymbol{w}^*) + \sum_{s=t}^{T-2} W(\hat{p}_t^{(s)}, \hat{p}_t^{(s+1)}) + \sqrt{\big(2\log(\frac{1}{\xi})/\zeta\big)}\big(\sqrt{\frac{1}{n_t}} + \sqrt{\frac{1}{\tilde{n}_{t|\mathbb{N}_t}}}\big), \tag{4}$$

Proof: included in the Appendix due to page limit.

Lemma 1 concludes that when a new task is learned at time $t = T$, ICLA updates the model parameters conditioned on minimizing the upper bound of e_t for all $t < T$ in Eq. 4. The last term in Eq. 4 is a small constant term when the number of training data points is large. If the network is complex enough so that the PDP hypothesis holds, then the classes would be separable in the embedding space and in the presence of enough labeled samples, the terms $e_{T-1,t}(\boldsymbol{w})$ would be small because $e_{T-1}(\boldsymbol{w})$ is minimized using ERM. The term $W(\phi(\hat{q}^{(t)}), \hat{p}_t^{(t)})$ would be small because we deliberately fit the GMM distribution $\hat{p}^{(t)}$ to the distribution $\phi(\hat{q}^{(t)})$ in the embedding space when learning the task $\mathcal{Z}^{(t)}$. Existence of this term indicates that our algorithm requires that internal distribution can be fit with a GMM distribution with high accuracy and this limits applicability of our algorithm. Note however, all parametric learning algorithms face this limitation. The term $e_{\mathcal{C}}(\boldsymbol{w}^*)$ is small because we continually match the distributions in the embedding space class-conditionally. Hence, if the model is trained on task $\mathcal{Z}^{(t)}$ and the pseudo-task at $t - T$, it will perform well on both tasks. Note that this is not trivial because if the wrong classes are matched across the domains in the embedding space, the term $e_{\mathcal{C}}(\boldsymbol{w}^*)$ will not be minimal. Finally, the sum term in Eq. 4 indicates the effect of experience replay. Each term in this sum is minimized at $s = t+1$ because we draw random samples from $\hat{p}_t^{(t)}$ and then train the autoencoder to enforce $\hat{p}_t^{(t)} \approx \psi(\phi(\hat{p}_t^{(t)}))$. Since all the terms in the upperbound of $e_t(\boldsymbol{w})$ in Eq. 4 are minimized when a new task is learned, catastrophic forgetting of the previous tasks will be mitigated. Another important intuition from Eq. 4 is that as more tasks are learned after learning a task, the upperbound becomes looser as more terms are accumulated in the sum which enhances forgetting. This observation accords with our intuition about forgetting as more time passes after initial learning time of a task or concept.

6 Experimental Validation

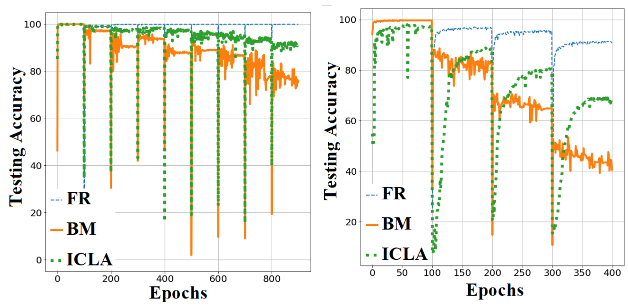
We validate our method on two sequential task learning settings: incremental learning and continual incremental learning. Incremental learning is a special case of our learning setting when each concept class is observed only in one task and concept drift does not exist. We use this special case to compare our method against existing incremental learning approaches.Our implementation is available as a supplement.

Evaluation Methodology : We use the same network structure for all the methods for fair comparison. To visualize the results, we generate learning curves by plotting the model performance on the testing split of datasets versus the training epochs, i.e, to model time. We report the average performance of five runs. Visualizing learning curves allows studying temporal aspects of learning. For comparison, we provide learning curves for: (a) full experience replay (FR) which stores the whole training data for all the previous tasks and (b) experience replay using a memory buffer (MB) with a fixed size, similar to Li et. al ([Li and Hoiem, 2018]). At each time-step, the buffer stores an equal number of samples per concept from the previous tasks. When a new task is learned, a portion of old stored samples are discarded and replaced with samples from the new task to keep the buffer size fixed. FR serves as a best achievable upperbound to measure the effectiveness of our method against the upperbound. For more details about the experimental setup and all parameteric values, please refer to the Appendix and the provided code.

6.1 Incremental Learning

The classes are encountered only at one task in incremental learning. We design two incremental learning experiments using the MNIST and the Fashion-MNIST datasets. Both datasets are classification datasets with ten classes. MNIST dataset consists of gray scale images of handwritten digits and Fashion-MNIST consists of images of common fashion products. We consider an incremental learning setting with nine tasks for the MNIST dataset. The first task is a binary classification of digits 0 and 1 and each subsequent task involves learning a new digit. The setup for Fashion-MNIST dataset is similar, but we considered four tasks and each task involves learning two fashion classes. We use a memory buffer with the fixed size of 100 for MB. We build an autoencoder by expanding a VGG-based classifier by mirroring the layers.

Figure 2 presents results for the designed experiments. For simplicity, we have provided condensed results for all tasks in a single curve. Each task is learned in 100 epochs and at each epoch, the model performance is computed as the average classification rate over all the classes, observed before. We report performance on the standard testing split of each dataset for the observed classes. Figure 2a and present the learning curves for the MNIST experiments. Similarly, Figure 2b present learning curves for the Fashion-MNIST experiments. We can see in both figures that FR (dashed blue curves) leads to superior performance. This is according to expectation but as we discussed, the challenge is the requirement for a memory buffer with an unlimited size. The buffer cannot have a fixed size as the number of data points grows when more tasks are learned. MB (solid yellow curves) is



(a) MNIST (b) FMNIST

Figure 2: Learning curves for the incremental learning experiments (a) MNIST and (b) Fashion-MNIST (FMNIST) datasets; (c) MNIST performance comparison (Best viewed in color on screen. See Appendix for enlarged versions.)

Method	2T	5T
CAB [He and Jaeger, 2018]	94.9±0.3	-
IMM [Lee *et al.*, 2017]	94.1±0.3	-
OWM [Zeng *et al.*, 2019]	96.3±0.1	-
GEM [Lopez-Paz and Ranzato, 2017]	-	78.0
iCarl [Rebuffi *et al.*, 2017]	-	81.0
GSS [Aljundi *et al.*, 2019]	-	61.0
DGR [Shin *et al.*, 2017]	88.7±2.6	-
MeRGAN [Wu *et al.*, 2018]	97.0	-
ICLA	97.2±0.2	91.6±0.4

Table 1: Classification accuracy for MNIST.

initially somewhat effective and comparable with ICLA, but as more tasks are learned, forgetting effect becomes more severe. This is because fewer data points per task can be stored in the buffer with fixed size as more tasks are learned. As a result, the stored samples would not be sufficiently representative of the past learned tasks. In comparison, we can generate as many pseudo-data points as desired.

We can also see in Figure 2a and Figure 2b that ICLA (dotted green curves) is able to mitigate catastrophic forgetting considerably better than MB and the performance difference between ICLA and MB increases as more tasks are learned. We also observe that ICLA is more effective for MNIST dataset. This is because FMNIST data points are more diverse. As a result, generating pseudo-data points that look more similar to the original data points is easier for the MNIST dataset given that we are using the same network structure for both tasks. Another observation is that the major performance degradation for ICLA occurs each time the network starts to learn a new concept class as initial sudden drops. This degradation occurs due to the existing distance between the distributions $\hat{p}_{J,k}^{(T-1)}$ and $\phi(q^{(s)})$ at $t = T$ for $s < T$. Although ICLA minimizes this distance, the autoencoder is not ideal and this distance is non-zero in practice.

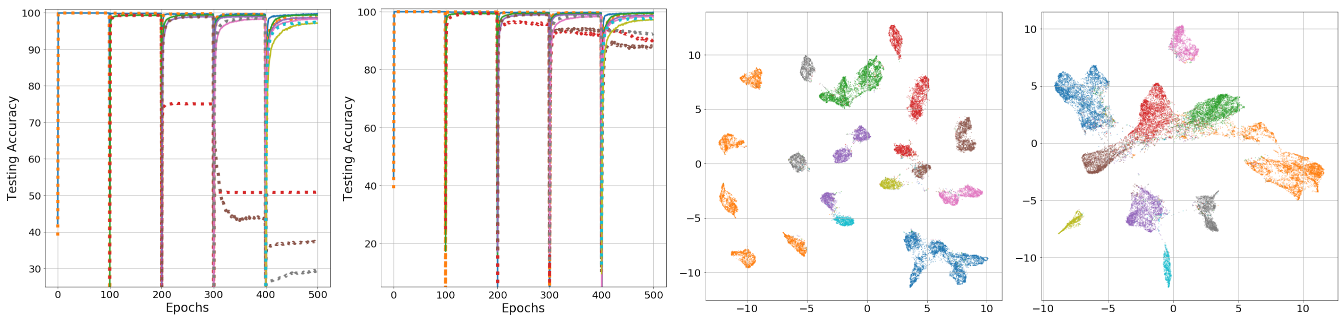
For comparison purpose, we have listed our performance and a number of methods for incremental learning on MNIST in Table 1. Two sets of incremental learning tasks have been designed using MNIST in the literature: 5 tasks (5T) setting and 2 tasks (2T) setting. In the 2T setting, two tasks are define involving digits $(0-4)$ and $(5-9)$. In the 5T setting, five binary classification tasks are defined involving digits $(0, 1)$ to $(8, 9)$. We have compared our performance against several methods, representative of prior works: CAB [He and Jaeger, 2018], IMM [Lee *et al.*, 2017], OWM [Zeng *et al.*, 2019], GEM [Lopez-Paz and Ranzato, 2017], iCarl [Rebuffi *et al.*, 2017], GSS [Aljundi *et al.*, 2019], DGR [Shin *et al.*, 2017], and MeRGAN [Wu *et al.*, 2018]. The CAB, IMM, and OWM methods are based on regularizing the network weights. The GEM, iCarl, and GSS methods use a memory buffer to store selected samples. Finally, DGR and MeRGAN methods are based on generative replay similar to ICLA but use adversarial learning. We have reported the classification accuracy on the ten digit classes after learning the last task in Table 1. A memory buffer with a fixed size of 100 is used for GEM, iCarl, and GSS. Following these works, an MLP with two layers is used as the base model for fair comparison.

We observe in Table 1 that when the buffer size is small, buffer-based methods perform poorly. Methods based on weight regularization perform quite well but note that these methods limit the network learning capacity. As a result, when the number of tasks grow, the network cannot be used to learn new tasks. Generative methods, including ICLA, perform better compared to buffer-based methods and at the same time do not limit the network learning capacity because the network weights can change after generating the pseudo-dataset. Although ICLA has the state-of-the-art performance for these tasks, there is no superior method for all conditions, because by changing the experimental setup, e.g., network structure, dataset, hyper-parameters such as memory buffer, etc, a different method may have the best performance result. However, we can conclude that ICLA has a superior performance when the network size is small and using a memory buffer is not possible, i.e., we have limited learning resources.

6.2 Continual Incremental Learning

Permuted MNIST task is a common supervised learning benchmark for sequential task learning [Kirkpatrick *et al.*, 2017]. The sequential tasks are generated using the MNIST dataset. Each task $\mathcal{Z}^{(t)}$ is generated by rearranging the pixels of all images in the dataset using a fixed random predetermined permutation transform and keeping the labels as their original value. As a result, we can generate many tasks that are diverse, yet equally difficult. As a result, these tasks are suitable for performing controlled experiments. Since no prior work has addressed incremental learning of drifting concepts, we should design a suitable set of tasks.

We design continual incremental learning tasks that share common concepts using five permuted MNSIT tasks. The first task is a binary classification of digits 0 and 1 for the MNIST dataset. For each subsequent task, we generate a permuted MNIST task but include only the previously seen digits plus two new digits in the natural number order, e.g., the third task includes permuted versions of digit $0-5$. This means that at each task, new forms of all the previously learned concepts are encountered, i.e, we need to learn drifting concepts, in additional to new tasks. Hence, the model needs to expand its knowledge about the previously learned concepts while learning new concepts. We use a memory buffer with size of 30000



(a) FR vs. MB (b) FR vs. ICLA (c) FR (d) ICLA

Figure 3: Learning curves for the five continual incremental learning tasks, designed using the permuted MNIST tasks (a) FR (solid) vs. MB (dotted), (b) FR (solid) vs. ICLA (dotted); UMAP visualization of (c) FR and (d) ICLA in the embedding space. (Best viewed in color.)

for MB. Due to the nature of these tasks, we use a multi-layer perceptron (MLP) network.Figure 3 presents learning curves for the five designed permuted MNIST tasks. In this figure, the learning curve for each task is illustrated with a different color and different line styles are used to distinguish the different methods (for enlarged versions, please refer to the Appendix). At each epoch time-step, model performance is computed as the average classification rate on the standard testing split of the current and all the past learned tasks.

Figure 3a presents learning curves for MB (dotted curves) and FR (solid curves). Unsurprisingly, FR leads to almost perfect performance. We also observe MB is less effective in this setting and catastrophic forgetting is severe for MB beyond the second task. The reason is that the concepts are more diverse in these tasks. As a result, it is more challenging to estimate the input distribution using a fixed number of stored samples that also decrease due to a fixed buffer size. We can conclude that as tasks become more complex, a larger memory buffer will be necessary which poses a challenge for MB. Figure 3b presents learning curves for FR (solid curves) and MB (dotted curve). As can be seen, ICLA is able to learn drifting concepts incrementally. Again, major forgetting effect for ICLA occurs as a sudden performance drop when learning a new task starts. This observation demonstrates that an important vulnerability for ICLA is the structure of the autoencoder that we build. This can be deduced from our theoretical result because an important condition for tightness of the provided bound in Lemma 1 is that we have: $\psi \approx \phi^{-1}$. Both our theoretical and experimental results suggest that if can build auto-encoders that can generate pseudo-data points with high quality, incremental learning can be performed using ICLA. In other words, learning quality depends on the generative power the base network structure. Finally, we also observe that as more tasks are learned after learning a particular task, model performance on that particular task degrades more. This observation is compatible with the nervous system as memories fade out when time passes.

In addition to requiring a memory buffer with an unlimited size, we also demonstrate that an issue for FR is inability to identify concepts across the tasks in the embedding space. We use the UMAP [McInnes *et al.*, 2018] tool to reduce the dimensionality of the data representations in the embedding space to two for 2D data visualization. We illustrated the testing split of data for all the tasks in the embedding space $\mathcal{Z}$ in Figure 3c for FR and Figure 3d for ICLA when the final task is learned. In these figures, each color corresponds to one of the digits $\{0, 1, \dots, 9\}$. As expected from the learning curves, data points for digits form separable clusters for both methods. This result verifies that the PDP hypothesis holds in these experiments and hence the internal distribution can be modeled using a GMM. The important distinction between FR and ICLA is that FR has led to the generation of distinct clusters for each concept class per task. This means that each concept class has not been learned internally as one concept and FR learns each concepts as several distinct concepts across the domains. This observation also serves as an ablative study for our method because it demonstrates that matching distributions class-conditionally in the embedding space is necessary, as justified by the theoretical analysis.

In figure 3d, we observe that ten clusters for the ten observed concepts are formed when ICLA is used. This observation demonstrates that ICLA is able to track modes of the GMM successfully as more tasks are learned. ICLA is also able to build concept classes that are semantically meaningful across all tasks based on the labels. This is the reason that we can learn new classes incrementally in a continual lifelong learning scenario. In other words, as opposed to FR, ICLA encodes each cross-task concept in a single mode of the internal GMM distribution. This allows for expanding concepts for cross-domain abstraction similar to humans.

7 Conclusions

We developed an algorithm for continual incremental learning of concepts based on modeling the internal distribution of input data as a GMM and then updating the GMM as new experiences are acquired. We track this distribution to accumulate the new learned knowledge to the past learned knowledge consistently. We expand the base classifier model to make a generative model to allow for generating a pseudo-dataset for pseudo-rehearsal and experience replay. We provided theoretical and empirical result to validate our algorithm.

Ethical Statement

We foresee no significant ethical issues for our work.

References

[Aljundi *et al.*, 2019] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, pages 11816–11825, 2019.

[Bonneel *et al.*, 2015] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Math. Imag. and Vision*, 51(1):22–45, 2015.

[Chen and Liu, 2016] Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 10(3):1–145, 2016.

[Courty *et al.*, 2017] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE TPAMI*, 39(9):1853–1865, 2017.

[French, 1991] Robert M French. Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks. In *Proceedings of the 13th annual cognitive science society conference*, volume 1, pages 173–178, 1991.

[French, 1999] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in Cog. Sciences*, 3(4):128–135, 1999.

[Gama *et al.*, 2014] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *CSUR*, 46(4):1–37, 2014.

[Gennari *et al.*, 1989] John H Gennari, Pat Langley, and Doug Fisher. Models of incremental concept formation. *Artificial intelligence*, 40(1-3):11–61, 1989.

[He and Jaeger, 2018] Xu He and Herbert Jaeger. Overcoming catastrophic interference using conceptor-aided backpropagation. In *International Conference on Learning Representations*, 2018.

[Jiang and Conrath, 1997] Jay J Jiang and David W Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the 10th Research on Computational Linguistics International Conference*, pages 19––33, 1997.

[Kirkpatrick *et al.*, 2017] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, and Others. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[Lake *et al.*, 2015] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[Lamprecht and LeDoux, 2004] Raphael Lamprecht and Joseph LeDoux. Structural plasticity and memory. *Nature Reviews Neuroscience*, 5(1):45, 2004.

[Lee *et al.*, 2017] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in neural information processing systems*, pages 4652–4662, 2017.

[Li and Hoiem, 2018] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018.

[Lopez-Paz and Ranzato, 2017] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476, 2017.

[McClelland and Rogers, 2003] James L McClelland and Timothy T Rogers. The parallel distributed processing approach to semantic cognition. *Nature reviews Neuro.*, 4(4):310–322, 2003.

[McClelland *et al.*, 1986] James L McClelland, David E Rumelhart, PDP Research Group, et al. Parallel distributed processing. *Explorations in the Microstructure of Cognition*, 2:216–271, 1986.

[McClelland *et al.*, 1995] James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3):419, 1995.

[McInnes *et al.*, 2018] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[Morgenstern *et al.*, 2014] Yaniv Morgenstern, Mohammad Rostami, and Dale Purves. Properties of artificial networks evolved to contend with natural spectra. *Proceedings of the National Academy of Sciences*, 111(Supplement 3):10868–10872, 2014.

[Rannen *et al.*, 2017] Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1320–1328, 2017.

[Rasch and Born, 2013] Björn Rasch and Jan Born. About sleep's role in memory. *Physiological Reviews*, 93(2):681–766, 2013.

[Rebuffi *et al.*, 2017] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[Robins, 1995] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

[Rostami and Galstyan, 2023] Mohammad Rostami and Aram Galstyan. Overcoming concept shift in domainaware settings through consolidated internal distributions. In *Proceedings of the AAAI conference on artificial intelligence*, volume 1, 2023.

[Rostami *et al.*, 2018] Mohammad Rostami, David Huber, and Tsai-Ching Lu. A crowdsourcing triage algorithm for geopolitical event forecasting. In *Proceedings of the 12th*

ACM Conference on Recommender Systems, pages 377–381, 2018.

[Rostami *et al.*, 2019] Mohammad Rostami, Soheil Kolouri, and Praveen Pilly. Complementary learning for overcoming catastrophic forgetting using experience replay. In *IJCAI*, 2019.

[Rostami *et al.*, 2020a] Mohammad Rostami, David Isele, and Eric Eaton. Using task descriptions in lifelong machine learning for improved performance and zero-shot transfer. *Journal of Artificial Intelligence Research*, 67:673–704, 2020.

[Rostami *et al.*, 2020b] Mohammad Rostami, Soheil Kolouri, Praveen Pilly, and James McClelland. Generative continual concept learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5545–5552, 2020.

[Rostami, 2021a] Mohammad Rostami. Lifelong domain adaptation via consolidated internal distribution. In *Proceedings of the 2021 NeurIPS Conference*, 2021.

[Rostami, 2021b] Mohammad Rostami. *Transfer Learning Through Embedding Spaces*. CRC Press, 2021.

[Roy *et al.*, 2020] Deboleena Roy, Priyadarshini Panda, and Kaushik Roy. Tree-cnn: a hierarchical deep cnn for incremental learning. *Neural Networks*, 121:148–160, 2020.

[Sarwar *et al.*, 2019] Syed Shakib Sarwar, Aayush Ankit, and Kaushik Roy. Incremental learning in deep convolutional neural networks using partial network sharing. *IEEE Access*, 2019.

[Saxe *et al.*, 2019] Andrew M Saxe, James L McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, page 201820226, 2019.

[Schaul *et al.*, 2016] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *IJCLR*, 2016.

[Shin *et al.*, 2017] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *NeurIPS*, pages 2990–2999, 2017.

[Stan and Rostami, 2021] Serban Stan and Mohammad Rostami. Unsupervised model adaptation for continual semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2593–2601, 2021.

[Widmer and Kubat, 1996] Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine learning*, 23(1):69–101, 1996.

[Wu *et al.*, 2018] Chenshen Wu, Luis Herranz, Xialei Liu, Joost van de Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. In *NeurIPS*, pages 5962–5972, 2018.

[Zeng *et al.*, 2019] Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372, 2019.

[Zenke *et al.*, 2017] Friedemann Zenke, Wulfram Gerstner, and Surya Ganguli. The temporal paradox of hebbian learning and homeostatic plasticity. *Curr. opinion in neuro.*, 43:166–176, 2017.