Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)

Complementary Learning for Overcoming Catastrophic Forgetting Using Experience Replay

Mohammad Rostami[1] , **Soheil Kolouri**[2] and **Praveen K. Pilly**[2]
[1]University of Pennsylvania
[2]HRL Laboratories, LLC
mrostami@seas.upenn.edu, skolouri@hrl.com, pkpilly@hrl.com

Abstract

Despite huge success, deep networks are unable to learn effectively in sequential multitask learning settings as they forget the past learned tasks after learning new tasks. Inspired from complementary learning systems theory, we address this challenge by learning a generative model that couples the current task to the past learned tasks through a discriminative embedding space. We learn an abstract generative distribution in the embedding that allows generation of data points to represent past experience. We sample from this distribution and utilize experience replay to avoid forgetting and simultaneously accumulate new knowledge to the abstract distribution in order to couple the current task with past experience. We demonstrate theoretically and empirically that our framework learns a distribution in the embedding, which is shared across all tasks, and as a result tackles catastrophic forgetting.

1 Introduction

Recent breakthrough of deep learning has led to algorithms with human-level performance for many machine learning applications. This may seem natural as these networks are the best accessible tools that mimic the human nervous system [Morgenstern *et al.*, 2014]. However, this success is highly limited to single task learning, and retaining learned knowledge in a continual learning setting remains a major challenge. That is, when a deep network is trained on multiple sequential tasks with diverse data distributions, the new obtained knowledge usually interferes with past learned knowledge. As a result, the network is often unable to accumulate the new learned knowledge in a manner consistent with the past experience and forgets past learned tasks by the time the new task is learned. This phenomenon is called "catastrophic forgetting" in the literature [French, 1999]. It is in contrast with continual learning ability of humans over their lifetime.

To mitigate catastrophic forgetting, one of the main approaches is to replay data points from past tasks that are stored selectively in a memory buffer [Robins, 1995]. This is consistent with the Complementary Learning Systems (CLS) theory [McClelland *et al.*, 1995]. CLS theory hypothesizes that a dual long-term and short-term memory system, involving the neocortex and the hippocampus, is necessary for the continual, lifelong learning ability of humans. In particular, the hippocampus rapidly encodes recent experiences as a short-term memory that is used to consolidate the knowledge in the slower neocortex as long-term memory through experience replays during sleep [Diekelmann and Born, 2010]. Similarly, if we selectively store samples from past tasks in a buffer, like in the neocortex, they can be replayed to the deep network in an interleaved manner with current task samples from recent-memory hippocampal storage to train the deep network jointly on past and current experiences. In other words, the online sequential learning problem is recast as an offline multitask learning problem that supports performance on all tasks. A major issue with this approach is that the memory size for storing data points grows as more tasks are learned. Building upon recent successes of generative models, this challenge has been addressed by amending the network structure such that it can generate pseudo-data points for the past learned tasks explicitly [Shin *et al.*, 2017].

In this paper, our goal is to address catastrophic forgetting via coupling sequential tasks in a latent embedding space. We model this space as the output of a deep encoder, which is between the input and the output layers of a deep classifier. Representations in this embedding space can be thought of neocortex representations in the brain, which capture learned knowledge. To consolidate knowledge, we minimize the discrepancy between the distributions of all tasks in the embedding space. In order to mimic the offline memory replay process in the sleeping brain [Rasch and Born, 2013], we amend the deep encoder with a decoder network to make the classifier network generative. The resulting autoencoding pathways can be thought of neocortical areas, which encodes and remembers past experiences. We fit a parametric distribution to the empirical distribution of data representations in the embedding space. This distribution can be used to generate pseudo-data points through sampling, followed by passing the samples into the decoder network. The pseudo-data points can then be used for experience replay of the previous tasks towards incorporation of new knowledge. This would enforce the embedding to be invariant with respect to the tasks as more tasks are learned; i.e., the network would retain the past learned knowledge as more tasks are learned.

2 Related Work

Past works have addressed catastrophic forgetting using two main approaches: model consolidation [Kirkpatrick *et al.*, 2017] and experience replay [Robins, 1995]. Both approaches implement a notion of memory to enable a network to remember the distributions of past learned tasks.

The idea of model consolidation is based upon separating the information pathway for different tasks in the network such that new experiences do not interfere with past learned knowledge. This is inspired from the notion of structural plasticity [Lamprecht and LeDoux, 2004]. During the learning of a task, important weight parameters for that task are identified and are consolidated when future tasks are learned. As a result, the new tasks are learned through free pathways in the network; i.e., the weights that are important to retain knowledge about distributions of past tasks mostly remain unchanged. Several methods exist for identifying important weight parameters. Elastic Weight Consolidation (EWC) models posterior distribution of weights of a given network as a Gaussian distribution that is centered around the weight values from past learned tasks and a precision matrix, defined as the Fisher information matrix of all network weights. The weights are then consolidated according to their importance; i.e., the value of Fisher coefficient [Kirkpatrick *et al.*, 2017]. In contrast to EWC, Zenke et al. [Zenke *et al.*, 2017] consolidate weights in an online scheme during task learning. If a network weight contributes considerably to changes in the network loss, it is identified as an important weight. More recently, Aljundi et al. [Aljundi *et al.*, 2018] use a semi-Hebbian learning procedure to compute the importance of the weight parameters in both an unsupervised and online scheme. The issue with the methods based on structural plasticity is that the network learning capacity is compromised to avoid catastrophic forgetting. As a result, the learning ability of the network decreases as more tasks are learned.

Methods that use experience replay retain the past tasks' distributions via replaying selected representative samples of past tasks continuously. Prior works have mostly investigated on how to store a subset of past experiences to reduce dependence on memory. These samples can be selected in different ways. Schaul et al. select samples such that the effect of uncommon samples in the experience is maximized [Schaul *et al.*, 2016]. Isele and Cosgun explore four potential strategies to select more helpful samples in a buffer for replay [Isele and Cosgun, 2018]. The downside is that storing samples requires memory, and selection becomes more complex as more tasks are learned. To reduce dependence on a memory buffer, similar to humans [French, 1999], Shin et al. [Shin *et al.*, 2017] developed a more efficient alternative by considering a generative model that can produce pseudo-data points of past tasks to avoid storing real data points. They use a generative adversarial structure to learn the tasks' distributions to allow for generating pseudo-data points without storing data. However, adversarial learning is known to require deliberate architecture design and selection of hyper-parameters [Roth *et al.*, 2017], and can suffer from mode collapse [Srivastava *et al.*, 2017]. Alternatively, we demonstrate that a simple autoencoder structure can be used as the base generative model.

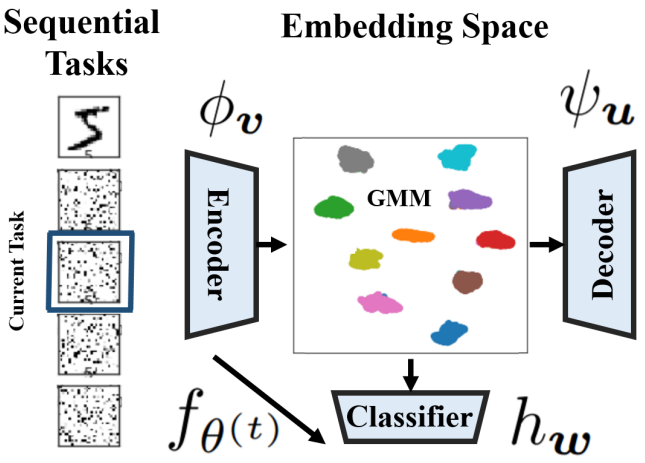


Figure 1: Architecture of the proposed framework.

Our contribution is to match the distributions of the tasks in the embedding layer of the autoencoder and learn a shared distribution across the tasks to couple them. The shared distribution is then used to generate samples for experience replay to avoid forgetting. We demonstrate the effectiveness of our approach theoretically and empirically validate our method on benchmark tasks that have been used in the literature.

3 Generative Continual Learning

We consider a lifelong learning setting [Chen and Liu, 2016], where a learning agent faces multiple, consecutive tasks $\{\mathcal{Z}^{(t)}\}_{t=1}^{T_{\text{Max}}}$ in a sequence $t = 1, \ldots, T_{\text{Max}}$. The agent learns a new task at each time step and proceeds to learn the next task. Each task is learned based upon the experiences gained from learning past tasks. Additionally, the agent may encounter the learned tasks in future and hence must optimize its performance across all tasks; i.e., not to forget learned tasks when future tasks are learned. The agent also does not know *a priori* the total number of tasks, which potentially might not be finite, the distributions of the tasks, and the order of tasks.

Let at time t, the current task $\mathcal{Z}^{(t)}$ with training dataset $\mathcal{Z}^{(t)} = \langle \boldsymbol{X}^{(t)}, \boldsymbol{Y}^{(t)} \rangle$ arrives. We consider classification tasks where the training data points are drawn i.i.d. in pairs from the joint probability distribution, i.e., $(\boldsymbol{x}_i^{(t)}, \boldsymbol{y}_i^{(t)}) \sim p^{(t)}(\boldsymbol{x}, \boldsymbol{y})$, which has the marginal distribution $q^{(t)}$ over $\boldsymbol{x}$. We assume that the lifelong learning agent trains a deep neural network $f_\theta : \mathbb{R}^d \to \mathbb{R}^k$ with learnable weight parameters θ to map the data points $\boldsymbol{X}^{(t)} = [\boldsymbol{x}_1^{(t)}, \ldots, \boldsymbol{x}_{n_t}^{(t)}] \in \mathbb{R}^{d \times n_t}$ to the corresponding one-hot labels $\boldsymbol{Y}^{(t)} = [\boldsymbol{y}_1^{(t)}, \ldots, \boldsymbol{y}_n^{(t)}] \in \mathbb{R}^{k \times n_t}$. Learning a single task in isolation is a standard classical learning problem. The agent can solve for the optimal network weight parameters using standard empirical risk minimization (ERM), $\hat{\theta}^{(t)} = \arg\min_\theta \hat{e}_\theta = \arg\min_\theta \sum_i \mathcal{L}_d(f_\theta(\boldsymbol{x}_i^{(t)}), \boldsymbol{y}_i^{(t)})$, where $\mathcal{L}_d(\cdot)$ is a proper loss function, e.g., cross entropy. Given large enough number of labeled data points n_t, the model trained on a single task $\mathcal{Z}^{(t)}$ will generalize well on the task test samples, as the empirical risk would be a suitable surrogate

for the real risk function (Bayes optimal solution), $e = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim p^{(t)}(\boldsymbol{x},\boldsymbol{y})}(\mathcal{L}_d(f_{\theta^{(t)}}(\boldsymbol{x}),\boldsymbol{y}))$ [Shalev-Shwartz and Ben-David, 2014]. The agent then can advance to learn the next task, but the challenge is that ERM is unable to tackle catastrophic forgetting as the model parameters are learned using solely the current task data, which can potentially have a very different distribution. Catastrophic forgetting can be considered as the result of considerable deviations of $\theta^{(T)}$ from past values over $\{\theta^{(t)}\}_{t=1}^{T-1}$ time as a result of drift in tasks' distributions $p^{(t)}(\boldsymbol{x},\boldsymbol{y})$. As a result, the updated $\theta^{(t)}$ can potentially be highly non-optimal for previous tasks. Our idea is to prevent catastrophic forgetting through mapping all tasks' data into an embedding space, where the tasks share a common distribution. We represent this space by the output of a deep network mid-layer, and we condition updating $\theta^{(t)}$ to what has been learned before in this discriminative embedding space. In other words, we want to train the deep network such the tasks are coupled in the embedding space by updating the parameters $\theta^{(T)}$ conditioned on $\{\theta^{(t)}\}_{t=1}^{T-1}$.

High performance of deep networks stems from learning data-driven and task-dependent high-quality features [Krizhevsky *et al.*, 2012]. In other words, a deep network maps data points into a discriminative embedding space, captured by network layers, where classification can be performed easily, e.g., classes become separable in the embedding. Following this intuition, we consider the deep network f_θ to be combined of an encoder $\phi_{\boldsymbol{v}}(\cdot)$ with learnable parameters $\boldsymbol{v}$, i.e., early layers of the network, and a classifier network $h_{\boldsymbol{w}}(\cdot)$ with learnable parameters $\boldsymbol{w}$, i.e., higher layers of the network. The encoder sub-network $\phi_{\boldsymbol{v}} : \mathcal{X} \to \mathcal{Z}$ maps the data points into the embedding space $\mathcal{Z} \subset \mathbb{R}^f$, which describes the input in terms of abstract discriminative features. Note that after training, the encoder network changes the input task data distribution.

If the embedding space is discriminative, this distribution can be modeled as a multi-modal distribution for a given task, e.g., using a Gaussian mixture model (GMM). Catastrophic forgetting occurs because this distribution is not stationery with respect to different tasks. The idea that we want to explore is based on training $\phi_{\boldsymbol{v}}$ such that all tasks share a similar distribution in the embedding; i.e., the new tasks are learned such that their distribution in the embedding matches the past experience. Doing so, the embedding space becomes invariant with respect to any learned input task, which in turn mitigates catastrophic forgetting.

The key question is how to adapt the standard supervised learning model $f_\theta(\cdot)$ such that the embedding space, captured in the deep network, becomes task-invariant. Following prior discussion, we use experience replay as the main strategy. We expand the base network $f_\theta(\cdot)$ into a generative model by amending the model with a decoder $\psi_{\boldsymbol{u}} : \mathcal{Z} \to \mathcal{X}$, with learnable parameters $\boldsymbol{u}$. The decoder maps the data representation back to the input space $\mathcal{X}$ and effectively makes the pair $(\phi_{\boldsymbol{u}}, \psi_{\boldsymbol{u}})$ an autoencoder. If implemented properly, we would learn a discriminative data distribution in the embedding space, which can be approximated by a GMM. This distribution captures our knowledge about past learned tasks. When a new task arrives, pseudo-data points for past tasks can be generated by sampling from this distribution and feeding the samples to the decoder network. These pseudo-data points can be used for experience replay in order to tackle catastrophic forgetting. Additionally, we need to learn the new task such that its distribution matches the past shared distribution. As a result, future pseudo-data points would represent the current task as well. Figure 1 presents a high-level block-diagram visualization of our framework.

4 Optimization Method

Following the above framework, learning the first task ($t = 1$) reduces to minimizing discrimination loss for classification and reconstruction loss for the autoencoder to solve for optimal parameters $\hat{\boldsymbol{v}}^{(1)}, \hat{\boldsymbol{w}^{(1)}}$ and $\hat{\boldsymbol{u}^{(1)}}$:

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

where $\mathcal{L}_r$ is the reconstruction loss and γ is a trade-off parameter between the two loss terms.

Upon learning the first task, as well as subsequent future tasks, we can fit a GMM distribution with k components to the empirical distribution represented by data samples $\{(\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(t)}), \boldsymbol{y}_i^{(t)})_{i=1}^{n_t}\}_{t=1}^{T}$ in the embedding space. The intuition behind this possibility is that as the embedding space is discriminative, we expect data points of each class to form a cluster in the embedding. Let $\hat{p}_J^{(0)}(\boldsymbol{z})$ denote this parametric distribution. We update this distribution after learning each task to accumulate what has been learned from the new task to the distribution. As a result, this distribution captures knowledge about past experiences. Upon learning this distribution, experience replay is feasible without saving data points. One can generate pseudo-data points in future through random sampling from $\hat{p}_J^{(T-1)}(\boldsymbol{z})$ at $t = T$ and then passing the samples through the decoder sub-network. It is also crucial to learn the current task such that its distribution in the embedding matches $\hat{p}_J^{(T-1)}(\boldsymbol{z})$. Doing so ensures suitability of GMM to model the empirical distribution.

Let $\mathcal{Z}_{ER}^{(T)} = \langle \boldsymbol{X}_{ER}^{(T)}, \boldsymbol{Y}_{ER}^{(T)} \rangle$ denote the pseudo-dataset generated at $t = T$. Following our framework, learning subsequent tasks reduces to solving the following problem:

$$\begin{aligned}
\min_{\boldsymbol{v},\boldsymbol{w},\boldsymbol{u}} & \sum_{i=1}^{n_t} \Big[\mathcal{L}_d\Big(h_{\boldsymbol{w}}(\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(T)})), \boldsymbol{y}_i^{(T)}\Big) \\
& \qquad + \gamma\mathcal{L}_r\Big(\psi_{\boldsymbol{u}}(\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(T)})), \boldsymbol{x}_i^{(T)}\Big)\Big] \\
& + \sum_{i=1}^{n_{er}} \Big[\mathcal{L}_d\Big(h_{\boldsymbol{w}}(\phi_{\boldsymbol{v}}(\boldsymbol{x}_{er,i}^{(T)})), \boldsymbol{y}_{er,i}^{(T)}\Big) \\
& \qquad + \gamma\mathcal{L}_r\Big(\psi_{\boldsymbol{u}}(\phi_{\boldsymbol{v}}(\boldsymbol{x}_{er,i}^{(T)})), \boldsymbol{x}_{er,i}^{(T)}\Big)\Big] \\
& + \lambda \sum_{j=1}^{k} D\Big(\phi_{\boldsymbol{v}}(q^{(T)}(\boldsymbol{X}^{(T)}|C_j)), \hat{p}_J^{(T-1)}(\boldsymbol{Z}_{ER}^{(T)}|C_j)\Big),
\end{aligned} \tag{2}$$

where $D(\cdot,\cdot)$ is a discrepancy measure (metric) between two probability distributions and λ is a trade-off parameter. The first four terms in Eq. (2) are empirical classification risk and autoencoder reconstruction loss terms for the current task and the generated pseudo-dataset. The third and the fourth terms enforce learning the current task such that the past learned knowledge is not forgotten. The fifth term is added to enforce the learned embedding distribution for the current task to be similar to what has been learned in the past, i.e., task-invariant. Note that we have conditioned the distance between the two distributions on classes to avoid the class matching challenge, i.e., when wrong classes across two tasks are matched in the embedding, as well as to prevent mode collapse from happening. Class-conditional matching is feasible because we have labels for both distributions. Adding this term guarantees that we can continually use GMM to fit the shared distribution in the embedding space.

The main remaining question is selecting the metric $D(\cdot,\cdot)$ such that it fits our problem. Since we are computing the distance between empirical distributions through drawn samples, we need a metric that can measure distances between distributions using the drawn samples. Additionally, we must select a metric that has non-vanishing gradients as deep learning optimization techniques are gradient-based methods. For these reasons, common distribution distance measures such as KL divergence and Jensen–Shannon divergence are not suitable [Kolouri *et al.*, 2018]. We rely on Wasserstein Distance (WD) metric [Bonnotte, 2013], which has been used extensively in deep learning applications. Since computing WD is computationally expensive, we use Sliced Wasserstein Distance (SWD) [Rabin and Peyré, 2011], which approximates WD, but can be computed efficiently.

SWD is computed through slicing a high-dimensional distribution. The d-dimensional distribution is decomposed into one-dimensional marginal distributions by projecting the distribution into one-dimensional spaces that cover the high-dimensional space. For a given distribution p, a one-dimensional slice of the distribution is defined as:

$$\mathcal{R}p(t;\gamma) = \int_{\mathcal{S}} p(\boldsymbol{x})\delta(t - \langle\gamma, \boldsymbol{x}\rangle)d\boldsymbol{x}, \tag{3}$$

where $\delta(\cdot)$ denotes the Kronecker delta function, $\langle\cdot,\cdot\rangle$ denotes the vector dot product, $\mathbb{S}^{d-1}$ is the d-dimensional unit sphere and γ is the projection direction. In other words, $\mathcal{R}p(\cdot;\gamma)$ is a marginal distribution of p obtained from integrating p over the hyperplanes orthogonal to γ.

SWD approximates the Wasserstein distance between two distributions p and q by integrating the Wasserstein distances between the resulting sliced marginal distributions of the two distributions over all γ:

$$SW(p,q) = \int_{\mathbb{S}^{d-1}} W(\mathcal{R}p(\cdot;\gamma), \mathcal{R}q(\cdot;\gamma))d\gamma, \tag{4}$$

where $W(\cdot)$ denotes the Wasserstein distance. The main advantage of using SWD is that it can be computed efficiently as the Wasserstein distance between one-dimensional distributions has a closed-form solution and is equal to the ℓ_p-distance between the inverse of their cumulative distribution functions. On the other hand, the ℓ_p-distance between cumulative distributions can be approximated as the ℓ_p-distance between the empirical cumulative distributions, which makes SWD suitable in our framework. Finally, to approximate the integral in Eq. (4), we rely on a Monte Carlo style integration and approximate the SWD between f-dimensional samples $\{\phi_{\boldsymbol{v}}(\mathbf{x}_i^{(t)} \in \mathbb{R}^f \sim q^{(t)}\}_{i=1}^{n_t}$ and $\{\phi_{\boldsymbol{v}}(\mathbf{x}_{er,i}^{(t)}) \in \mathbb{R}^f \sim \hat{p}_J^{(t)}\}_{j=1}^{n_t}$ in the embedding space as the following sum:

$$SW^2(\phi_{\boldsymbol{v}}(q^{(t)}), \hat{p}_J^{(t)}) \approx \frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{n_t} |\langle\gamma_l, \phi_{\boldsymbol{v}}(\mathbf{x}_{t_l[i]}^{(t)})\rangle - \langle\gamma_l, \phi_{\boldsymbol{v}}(\mathbf{x}_{t_l[er,i]}^{(t)})\rangle|^2 \tag{5}$$

where $\gamma_l \in \mathbb{S}^{f-1}$ denote random samples that are drawn from the unit f-dimensional ball $\mathbb{S}^{f-1}$, and $s_l[i]$ and $t_l[i]$ are the sorted indices of $\{\gamma_l \cdot \phi_{\boldsymbol{v}}(\mathbf{x}_i)\}_{i=1}^{n_t}$ for the two one-dimensional distributions. We utilize the SWD as the discrepancy measure between the distributions in Eq. (2) to learn each task. We tackle catastrophic forgetting using the proposed procedure. Our algorithm, named Continual Learning using Encoded Experience Replay (CLEER), is summarized in Algorithm 1.

Algorithm 1 CLEER (L, λ)

Input: data $\mathcal{D}^{(t)} = (\boldsymbol{X}^{(t)}, \boldsymbol{Y}^{(t)})_{t=1}^{T_{\text{Max}}}$.
Pre-training: learning the first task ($t = 1$)
$\hat{\theta}^{(1)} = (\boldsymbol{u}^{(1)}, \boldsymbol{v}^{(1)}, \boldsymbol{w}^{(1)}) = \arg\min_\theta \sum_i [\mathcal{L}_d(f_\theta(\boldsymbol{x}_i^{(t)}), \boldsymbol{y}_i^{(t)}) +$
$\gamma\mathcal{L}_r(\psi_{\boldsymbol{u}}(\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(1)})), \boldsymbol{x}_i^{(1)})]$
Estimate $\hat{p}_J^{(0)}(\cdot)$ using $\{\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(1)}))\}_{i=1}^{n_t}$
for $t = 2, \ldots, T_{\text{Max}}$ **do**
  Generate pseudo-dataset:
    $\mathcal{D}_{\text{ER}} = \{(\boldsymbol{x}_{er,i}^{(t)} = \psi(\boldsymbol{z}_{er,i}^{(t)}), \boldsymbol{y}_{er,i}^{(t)}) \sim \hat{p}_J^{(t-1)}(\cdot)\}_{i=1}^{n_{er}}$
  Update learnable parameters using pseudo-dataset: Eq. (2)
  Estimate: $\hat{p}_J^{(t)}(\cdot)$
    use $\{\phi_{\boldsymbol{v}}(\boldsymbol{x}_i^{(t)})), \phi_{\boldsymbol{v}}(\boldsymbol{x}_{er,i}^{(t)}))\}_{i=1}^{n_t}$
end for

5 Theoretical Justification

We use existing theoretical results about using optimal transport within domain adaptation [Redko *et al.*, 2017], to justify why our algorithm can tackle catastrophic forgetting. Note that the hypothesis class in our learning problem is the set of all functions represented by the network $f_\theta(\cdot)$ parameterized by θ. For a given model in this class, let e_t denote the observed risk for a particular task $\mathcal{Z}^{(t)}$ and e_t^J denote the observed risk for learning the network on samples of the distribution $\hat{p}_J^{(t-1)}$. We rely on the following theorem.

Theorem 1 [Redko *et al.*, 2017]: Consider two tasks $\mathcal{Z}^{(t)}$ and $\mathcal{Z}^{(t')}$, and a model $f_{\theta^{(t')}}$ trained for $\mathcal{Z}^{(t')}$, 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 $\min(n_t, n_{t'}) \geq \max(\xi^{-(d'+2),1})$ with probability at least $1 - \xi$ for all $f_{\theta^{(t')}}$, the following holds:

$$e_t \leq e_{t'} + W(\hat{p}^{(t)}, \hat{p}^{(t')}) + e_{\mathcal{C}}(\theta^*) + \sqrt{(2\log(\frac{1}{\xi})/\zeta)}\left(\sqrt{\frac{1}{n_t}} + \sqrt{\frac{1}{n_{t'}}}\right), \tag{6}$$

where $W(\cdot)$ denotes the Wasserstein distance between empirical distributions of the two tasks and θ^* denotes the optimal parameter for training the model on tasks jointly, i.e., $\theta^* = \arg\min_\theta e_{\mathcal{C}}(\theta) = \arg\min_\theta\{e_t + e_{t'}\}$.

We observe from Theorem 1 that performance, i.e., real risk, of a model learned for task $\mathcal{Z}^{(t')}$ on another task $\mathcal{Z}^{(t)}$ is upper-bounded by four terms: i) model performance on task $\mathcal{Z}^{(t')}$, ii) the distance between the two distributions, iii) performance of the jointly learned model f_{θ^*}, and iv) a constant term that depends on the number of data points for each task. Note that we do not have a notion of time in this Theorem; i.e., the roles of $\mathcal{Z}^{(t)}$ and $\mathcal{Z}^{(t')}$ can be shuffled and the theorem would still hold. In our framework, we consider the task $\mathcal{Z}^{(t')}$ to be the pseudo-task, i.e., the task derived by drawing samples from $\hat{p}_J^{t'}$ and then feeding the samples to the decoder sub-network. We use this result to conclude the following.

Lemma 1 : Consider CLEER algorithm for lifelong learning after $\mathcal{Z}^{(T)}$ is learned at time $t = T$. Then all tasks $t < T$ and under the conditions of Theorem 1, we can conclude the following inequality:

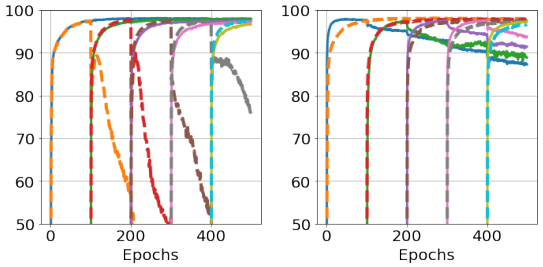
$$e_t \leq e_{T-1}^J + W(\hat{q}^{(t)}, \psi(\hat{p}_J^{(t)})) + \sum_{s=t}^{T-2} W(\psi(\hat{p}_J^{(s)}), \psi(\hat{p}_J^{(s+1)})) + e_{\mathcal{C}}(\theta^*) + \sqrt{(2\log(\frac{1}{\xi})/\zeta)}(\sqrt{\frac{1}{n_t}} + \sqrt{\frac{1}{n_{er,t-1}}}), \tag{7}$$

Proof: We consider $\mathcal{Z}^{(t)}$ with empirical distribution $\hat{q}^{(t)}$ and the pseudo-task with the distribution $\psi(\hat{p}_J^{(T-1)})$ in the network input space, in Theorem 1. Using the triangular inequality on the term $W(\hat{q}^{(t)}, \psi(\hat{p}_J^{(T-1)}))$ recursively, i.e., $W(\hat{q}^{(t)}, \psi(\hat{p}_J^{(s)})) \leq W(\hat{p}^{(t)}, \psi(\hat{p}_J^{(s-1)})) + W(\psi(\hat{p}_J^{(s)}), \psi(\hat{p}_J^{(s-1)}))$ for all $t \leq s < T$, Lemma 1 can be derived.

Lemma 1 explains that When future tasks are learned, our algorithms updates the model parameters conditioned on minimizing the upper bound of e_t in Eq. 7. Given suitable network structure and in the presence of enough labeled data points, the terms e_{t-1}^J and $e_{\mathcal{C}}(\theta^*)$ are minimized using ERM, and the last constant term would be small. The term $W(\hat{q}^{(t)}, \psi(\hat{p}_J^{(t)}))$ is minimal because we deliberately fit the distribution $\hat{p}_J^{(t)}$ to the distribution $\phi(\hat{q}^{(t)})$ in the embedding space and ideally learn ϕ and ψ such that $\psi \approx \phi^{-1}$. This term demonstrates that minimizing the discrimination loss is critical as only then can we fit a GMM distribution on $\phi(\hat{p}^{(t)})$ with high accuracy. Similarly, the sum terms in Eq. 7 are minimized because at $t = s$ we draw samples from $\hat{p}_J^{(s-1)}$ and enforce indirectly $\hat{p}_J^{(s-1)} \approx \phi(\psi(\hat{p}_J^{(s-1)}))$. Since the upper bound of e_t in Eq. 7 is minimized and conditioned on its tightness, the task $\mathcal{Z}^{(t)}$ will not be forgotten.

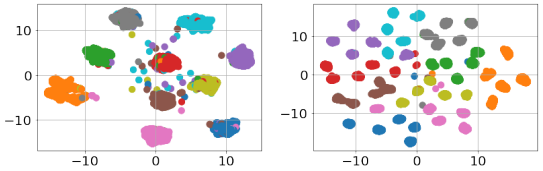
6 Experimental Validation

We validate our method on learning two sets of sequential tasks: permuted MNIST tasks and related digit classification tasks. Our implementation code is available on GitHub.



(a) BP vs. EWC (b) CLEER vs. FR

Figure 2: Performance results for permuted MNIST tasks. (Best viewed in color.)



(a) CLEER (b) FR

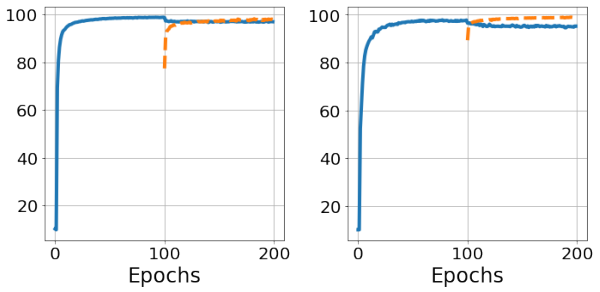
Figure 3: UMAP visualization of CLEER versus FR for permuted MNIST tasks. (Best viewed in color.)

6.1 Learning Sequential Independent Tasks

Following the literature, we use permuted MNIST tasks to validate our framework. The sequential tasks involve classification of handwritten images of MNIST ($\mathcal{M}$) dataset [LeCun *et al.*, 1990], where pixel values for each data point are shuffled randomly by a fixed permutation order for each task. As a result, the tasks are independent and quite different from each other. Since knowledge transfer across tasks is less likely to happen, these tasks are a suitable benchmark to investigate the effect of an algorithm on mitigating catastrophic forgetting as past learned tasks are not similar to the current task. We compare our method against: a) normal back propagation (BP) as a lower bound, b) full experience replay (FR) of data for all the previous tasks as an upper bound, and c) EWC as a competing model consolidation framework.

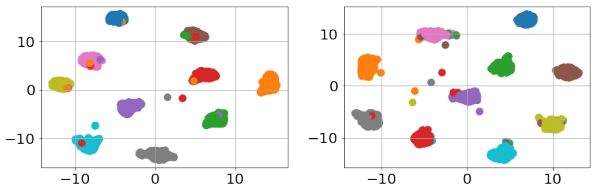
We learn permuted MNIST tasks using a simple multilayer perceptron (MLP) network trained via standard stochastic gradient descent and compute the performance of the network on the testing split of each task data at each iteration. Figure 2 presents results on five permuted MNIST tasks. Figure 2a presents learning curves for BP (dotted curves) and EWC (solid curves) [1]. We observe that EWC is able to address catastrophic forgetting quite well. But a close inspection reveals that as more tasks are learned, the asymptotic performance on subsequent tasks is less than the single task learning performance (roughly 4% less for the fifth task). This can be understood as a side effect of model consolidation, which limits the learning capacity of the network. This in an inherent limitation for techniques that regularize network parameters to prevent catastrophic forgetting. Figure 2b presents learning curves for our method (solid curves) ver-

[1]We have used PyTorch implementation of EWC [Hataya, 2019].



(a) $\mathcal{M} \rightarrow \mathcal{U}$ (b) $\mathcal{U} \rightarrow \mathcal{M}$

Figure 4: Performance results on MNIST and USPS digit recognition tasks. (Best viewed in color.)



(a) $\mathcal{M} \rightarrow \mathcal{U}$ (b) $\mathcal{U} \rightarrow \mathcal{M}$

Figure 5: UMAP visualization for $\mathcal{M} \rightarrow \mathcal{U}$ and $\mathcal{U} \rightarrow \mathcal{M}$ tasks. (Best viewed in color.)

sus FR (dotted curves). As expected, FR can prevent catastrophic forgetting perfectly but as we discussed the downside is the memory growth challenge. FR result in Figure 2b demonstrates that the network learning capacity is sufficient for learning these tasks, and that if we have a perfect generative model, we can prevent catastrophic forgetting without compromising the network learning capacity. Despite more forgetting in our approach compared to EWC, the asymptotic performance after learning each task, just before advancing to learn the next task, has been improved. We also observe that our algorithm suffers an initial drop in performance of previous tasks, when we proceed to learn a new task. Forgetting beyond this initial forgetting is negligible. This can be understood as the existing distance between $\hat{p}_J^{(T-1)}$ and $\phi(q^{(t)})$ at $t = T$. In other words, our method can be improved if more advanced autoencoder structures are used. These results suggest that catastrophic forgetting may be tackled better if both model consolidation and experience replay are combined.

To provide a better intuitive understating, we have also included the representations of the testing data for all tasks in the embedding space of the MLP in Figures 3. We have used UMAP [McInnes *et al.*, 2018] to reduce the dimensions for visualization purpose. In these figures, each color corresponds to a specific class of digits. We can see that although FR is able to learn all tasks and form distinct clusters for each digit class for each task, five different clusters are formed for each class in the embedding space. This suggests that FR is unable to learn the concept of the same class across different tasks in the embedding space. In comparison, we observe that CLEER is able to match the same class across the different tasks; i.e., we have exactly ten clusters for the ten digits. This empirical observation demonstrates that we can model the data distribution in the embedding using a multi-modal distribution such as a GMM [Heinen *et al.*, 2012].

6.2 Learning Sequential Tasks in Related Domains

We performed a second set of experiments on related tasks to investigate the ability of the algorithm to learn new domains. We consider two digit classification datasets for this purpose: MNIST ($\mathcal{M}$) and USPS ($\mathcal{U}$) datasets. Despite being similar, USPS dataset is a more challenging task as the size of the training set is smaller.We also resized the USPS images to 28×28 pixels. We consider the two possible sequential learning scenarios: $\mathcal{M} \rightarrow \mathcal{U}$ and $\mathcal{M} \rightarrow \mathcal{U}$. The experiments can be considered as a special case of domain adaptation as both tasks are digit recognition tasks in different domains. we use a CNN to capture cross-tasks relations.

Figure 4 presents learning curves for these two tasks. We observe that the network retains the knowledge about the first domain, after learning the second domain. We also see that forgetting is negligible compared to unrelated tasks and there is a jump-start in performance. These observations suggest relations between the tasks help to avoid forgetting. As a result of task similarities, the empirical distribution can capture the task distribution more accurately. As expected from the theoretical justification, this empirical result suggests the performance of our algorithm depends on the closeness of the distribution $\psi(\hat{p}_J^{(t)})$ to the distributions of previous tasks. And improving probability estimation will increase the performance of our approach. We have also presented UMAP visualization of all tasks' data in the embedding in Figure 5. As expected the distributions are matched in the embedding.

7 Conclusions

Inspired from CLS theory, we addressed the challenge of catastrophic forgetting for sequential learning of multiple tasks using experience replay. We amend a base learning model with a generative pathway that encodes experiences meaningfully as a parametric distribution in an embedding space. This idea makes experience replay feasible without requiring a memory buffer to store task data. The algorithm is able to accumulate new knowledge in a manner consistent with past learned knowledge, as the parametric distribution in the embedding space is enforced to be shared across all tasks. Compared to model-based approaches that regularize the network to consolidate the important weights for past tasks, our approach is able to address catastrophic forgetting without limiting the learning capacity of the network. Future works for our approach may extend to learning new tasks and/or classes with limited labeled data points.

Acknowledgments

This material is based upon work supported by the United States Air Force and DARPA under Contract No. FA8750-18-C-0103. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force and DARPA.

References

[Aljundi *et al.*, 2018] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[Bonnotte, 2013] Nicolas Bonnotte. *Unidimensional and evolution methods for optimal transportation*. PhD thesis, Paris 11, 2013.

[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.

[Diekelmann and Born, 2010] S. Diekelmann and J. Born. The memory function of sleep. *Nat Rev Neurosci*, 11(114), 2010.

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

[Hataya, 2019] Ryuichiro Hataya. EWC PyTorch. https://github.com/moskomule/ewc.pytorch, 2019. [Online; accessed 14-June-2019].

[Heinen *et al.*, 2012] Milton Roberto Heinen, Paulo Martins Engel, and Rafael C Pinto. Using a gaussian mixture neural network for incremental learning and robotics. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2012.

[Isele and Cosgun, 2018] David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[Kirkpatrick *et al.*, 2017] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[Kolouri *et al.*, 2018] Soheil Kolouri, Gustavo K Rohde, and Heiko Hoffmann. Sliced wasserstein distance for learning gaussian mixture models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3427–3436, 2018.

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

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

[LeCun *et al.*, 1990] Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems*, pages 396–404, 1990.

[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.

[Rabin and Peyré, 2011] Julien Rabin and Gabriel Peyré. Wasserstein regularization of imaging problem. In *2011 18th IEEE International Conference on Image Processing*, pages 1541–1544. IEEE, 2011.

[Rasch and Born, 2013] B. Rasch and J. Born. About sleep's role in memory. *Physiol Rev*, 93:681–766, 2013.

[Redko *et al.*, 2017] Ievgen Redko, Amaury Habrard, and Marc Sebban. Theoretical analysis of domain adaptation with optimal transport. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 737–753. Springer, 2017.

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

[Roth *et al.*, 2017] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. In *Advances in Neural Information Processing Systems*, pages 2018–2028, 2017.

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

[Shalev-Shwartz and Ben-David, 2014] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

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

[Srivastava *et al.*, 2017] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems*, pages 3308–3318, 2017.

[Zenke *et al.*, 2017] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3987–3995. JMLR. org, 2017.