Generating Accurate Pseudo Examples for Continual Learning

Daniel L. Silver
Jodrey School of Computer Science
Acadia University
danny.silver@acadiau.ca

Sazia Mahfuz
School of Computing
Queens Uninversity
sazia.mahfuz@queensu.ca

Abstract

Continual learning (CL) is concerned with the persistent and cumulative nature of learning. This requires a method of successfully consolidating new knowledge into long-term memory without the loss of prior knowledge. Prior research has addressed this CL retention problem through the efficient rehearsal of prior examples while learning the examples of a new task within a long-term Multiple Task Learning (MTL) network. The approach maintains or improves prior knowledge while allowing its representation to remain plastic for the integration of new task examples. Preferably, rehearsal is done using pseudo examples synthesized by the MTL network; eliminating the need to retain prior task training examples or a generate them with an additional model. Previous work has shown that to properly retain knowledge the pseudo examples must adhere to the input probability distribution of those original examples. Two approaches are investigated for creating appropriate pseudo examples from a Restricted Boltzmann Machine (RBM) autoencoder, which can reside in the lowest layers of the long-term MTL Deep Belief network. We show that appropriate pseudo examples can be reconstructed by passing uniform random examples to a generative RBM model and selecting only those with reconstruction error less than the mean training error. These pseudo examples are shown to adhere to the probability distribution of the input variables of the original training examples and retain prior task knowledge during rehearsal as well as those examples. As part of the research, we develop and test a new metric called the Autoencoder Divergence Measure for comparing the probability distributions of two datasets given to a generative RBM network based on their reconstruction mean squared error.

1. Introduction

A human learns new knowledge throughout her life while retaining prior knowledge. Similarly, a Continual Learning (CL) system should be able to retain prior task knowledge over a long time while integrating, or consolidating new task's knowledge periodically [21]. The challenge in consolidation lies in the process of retaining the old information while integrating the new information. This is known as the stability-plasticity dilemma [6]. Stability refers to the learning system's need to retain prior knowledge effectively (accurately) and efficiently (minimal memory and time). Plasticity refers to the learning system's need to accommodate new knowledge effectively and efficiently. A solution to the stability-plasticity problem has previously been proposed and tested using back-propagation multiple task learning (MTL) neural networks and a technique called *sweep task rehearsal* [20]. The approach rehearses examples of prior tasks while learning the examples of a new task. Recently, this has been referred to as replaying prior training examples [23, 19], but its approach dates back to work from the 1990s [16, 17] using the better psychological term of *rehearsal*. Sweep task rehearsal has been shown to maintaining functional knowledge of prior tasks within a neural network while changing its representation to accommodate new knowledge. In this way the networks provide *functional stability* as well as *representational plasticity*.

Sweep task rehearsal works well, but it requires training examples of prior tasks that preserve the probability distribution over the input variables [20]. One approach is to store a set of examples from the training set for each task. The problem with this approach is that it's complexity grows linearly with the number of tasks, each time lengthening the number of retained examples and network training time. A more efficient approach is to (1) generate *pseudo examples (PE)* using the consolidated MTL model of prior tasks and knowledge of the input variable distribution, and (2) weight the derivatives of the backward propagated error for the PEs to give them equal opportunity to effect the representation of the network along with the new examples. This paper seeks to advance the sweep task rehearsal consolidation approach by finding a method that does not require keeping explicit knowledge such as prior training examples or the probability distribution over those examples.

The contributions of this paper are: (1) an approach to generating accurate PEs of prior tasks, using generative components of a MTL network, such that those PEs adhere to the probability distribution of the input variables of the original training examples; (2) a demonstration using PEs generated in this manner to rehearse a prior task within a MTL network while successfully consolidating a new task; and (3) the development of a simple measure for comparing the similarity of the probability distributions of two sets of examples, called the Autoencoder Divergence Measure.

This paper has the following sections. Section 2 provides background knowledge on CL, Restricted Boltzmann Machines (RBM), consolidation, rehearsal, and the metrics used for selecting and evaluating pseudo examples. Section 3 discusses recent related work. Section 4 presents our theory for generating pseudo examples such that they adhere to the probability distribution of the input variables of training examples. Sections 5 and 6 present empirical studies of the approaches for generating pseudo examples on a synthetic and a real-world domain of tasks. Section 7 demonstrates the value of using the more accurate PEs when consolidating one task with a prior task. Section 8 presents the findings of this research and potential future work.

2. Background

This section provides the necessary background for the development of our theory and approach.

Continual Learning: CL deals with systems that can retain knowledge of many tasks from a domain over time and can selectively transfer that knowledge when learning a new task and improve upon prior knowledge as new tasks are learned [21]. CL can be characterized by being composed of two tightly integrated phases: *transfer learning* and *consolidation*.

Transfer Learning: Transfer learning refers to using the existing information of the learning system to facilitate new learning [17]. Formally, let D be the domain of source and target tasks, t_s be the source task (the model from which the knowledge transfer occurs), t_t be the target task (the model to which transfer is received) and $t_s \neq t_t$. Then transfer learning occurs when the performance of a model for task t_t with transfer from t_s improves over a model for task t_t without transfer [12].

Consolidation: Consolidation of information is a principle requirement of a CL system that enables integration of new information into the representation of the system. The main challenge with consolidation is preventing the *catastrophic forgetting* of prior learned knowledge and overcoming the *stability-plasticity* dilemma. Catastrophic forgetting can be defined as the disruption or loss of the prior training information while integrating new information to a trained model [16]. The stability-plasticity dilemma deals with how to eliminate or reduce the affect of catastrophic forgetting.

Multiple Task Learning: MTL is the learning of two or more tasks within the same machine learning system, typically a neural network with the intention of developing shared internal representation that is beneficial to the development of both tasks, in this way *transferring* knowledge from one task to the other [3].

Consolidation via Rehearsal and Pseudorehearsal: To reduce the affect of catastrophic forgetting, Robins introduces *rehearsal* of a subset of the prior training examples while training on new examples [16]. The addition of the prior examples forces the learning system to maintain functional accuracy for both old and new training examples. This is a powerful concept supported by cognitive and neural science [18]. Robins also introduced *pseudo-rehearsal* which uses pseudo examples, or PEs, created by passing random input vectors through the learning system and then recording the output generated as the label for that example. This prevents having to retain real examples; the PE can be generated by the current model and added to the real examples of the new tasks as needed. Robins also discusses different pseudo-rehearsal approaches, and amongst them *sweep pseudo-rehearsal* is the most efficient and effective. Sweep pseudo-rehearsal is the process where each time new examples are to be learned by the system, a set of PE are randomly created and for each training iteration a subset of these example are randomly chosen for rehearsal.

The rehearsal and pseudo-rehearsal approaches have been used with CL MTL networks to learn a sequence of tasks with transfer from prior knowledge. In [20] up to twenty tasks are trained in a row with little or no loss of prior tasks accuracy, and at times the accuracy of prior tasks is shown to improve as new task examples are consolidated. This work also shows that it very important to create PEs that adhere to the probability distribution of the training data; only then can one be guaranteed that rehearsal will maintain prior task knowledge.

Restricted Boltzmann Machine (RBM): A Restricted Boltzmann Machine (RBM) is an unsupervised generative neural network model that is trained to reconstruct whatever is presented at its inputs (visual layer) after passing these inputs to a hidden layer and back. RBM networks form a bipartite graph; where a neuron in one layer is connected to all the neurons in the next layer, however connections between neurons in the same layer are not allowed. A RBM uses a gradient descent algorithm called *contrastive divergence* to minimize the error between the reconstructed values at the visual layer and the training examples [8].

Maximum Mean Discrepancy (MMD): The probability distribution over the input variables of a dataset is represented by the probability density function of a vector $\{x_1, x_2, ..., x_n\}$ from an input feature space X. Comparing probability distributions of different data, we make the assumption that the data have originated from the same feature space X. MMD is a measure for comparing the probability distributions of two given datasets based on their mean and variance statistics. The MMD-based hypothesis test provides the degree of confidence that one set of examples agrees with another set of examples in terms of their variable distributions [5]. The MMD test returns a *Value* and a *Bound*. As long as the Value is less than the Bound, the null hypothesis that the two distributions are the same is accepted. MMD is used in our research for comparing the probability distributions of the generated pseudo examples for a task and it's original training examples.

3. Related Work

Recently, there has been considerable interest in Continual Learning as a function of the success of deep learning and the development of rich feature spaces within deep neural networks [13]. This has brought about renewed interest in the problem of catastrophic forgetting and the stability-plasticity dilemma as the importance of consolidation in CL systems becomes more apparent [10].

Kirkpatrick et al. [9] presents a novel algorithm, *elastic weight wonsolidation* (EWC), which avoids catastrophic forgetting of prior knowledge during the integration of new examples into the model. EWC can selectively decrease the plasticity of weights while training on new examples, thus protecting prior knowledge in the network. Srivastava et al [22] show that interference among patterns leads to catastrophic forgetting when the number of stored patterns exceed a critical limit. They demonstrate an approach that eliminates catastrophic forgetting using Gram-Schmidt orthogonalization combined with a Hebb-Hopfield-type model. He et al [7] propose a variant of the backpropagation algorithm, called *conceptor-aided backprop*, where conceptors are used to protect the gradients of prior trained tasks from degradation.

Rebuffi et al [15] develop a class-incremental learning strategy named iCaRL (incremental classifier and representation learning), which stores exemplars representing the examples of the previously learned classes and the weight vector associated with the classes. iCARL adjusts the exemplar set when it comes across new classes. Though iCARL does not need to store the training data for the learned classes, still it needs the exemplars and the weight vector associated with the new classes, which requires substantial storage. Lopez-paz et al [11] develop a continual learning strategy called Gradient Episodic Memory (GEM) which stores a subset of the examples from the previous tasks, which again requires substantial storage.

De Lange et al [10] provide a comparative study between current CL methods and four baseline methods. The paper deals with task-incremental classification, where different tasks are learned sequentially. Parisi et al [14] present a review of research on CL with neural networks. The authors identify that the current research is still lacking in terms of flexibility, robustness, and scalability shown in biological systems. They also point out that the most research focuses on supervised domains of tasks which requires large amounts of labelled data, and that this does not capture the real challenge faced by the lifelong learning agents; to build a base of low level features of which future learning can take advantage.

Recent research by Atkinson and Robins et al [1] use pseudo-rehearsal with a Generative Adversarial Network (GAN). Every time a new task is presented, pseudo-rehearsal on the GAN model is achieved by generating pseudo-images from the GAN and mixing them with the current task examples. They termed this process pseudo-recursal as it can be repeated recursively. Atkinson and Robins et al [2] develop a Reinforcement-Pseudo-Rehearsal model (RePR) using pseudo-rehearsal and a GAN to achieve iterative learning in reinforcement learning tasks. The RePR model can be used to effectively learn multiple sequential tasks, without increasing model complexity and without having to store training data for prior tasks. This work is closest to the the research reported in this paper, but the method requires the use of GAN model to generate the PEs that are in addition to the consolidated task model.

4. Theory

A method is required to generate PEs having the same probability distribution over the input variables as the training examples of the prior tasks. Preferably, the solution uses the knowledge retained in the long-term consolidated MTL neural network used to model the supervised tasks. We wish to avoid having to use a separate network to generate the PEs as this would add computational time and storage requirements as per some of the related work.

To achieve this, we consider MTL Deep Belief Networks, that use stacked unsupervised RBMs in the lower layers of a network to develop representations that are useful for training supervised back-propagation representations in the upper layers of that network. After a RBM model has been trained to a low reconstruction error, the model has learned the probability distribution of the training data. This is because the algorithm works to develop a model that learns the $p(h|v)$ at each hidden node and $p(v|h)$ at each visible node; where h is the set of hidden nodes, and v is the set of visible input nodes [8]. Such a network configuration is exactly what is needed to develop rich internal representations for a domain, to generate appropriate PEs, and to learn multiple tasks. We propose two approaches for using RBMs to generating appropriate PEs.

4.1. Approaches Considered

Relaxation of a Trained RBM: Ideally, when a uniform random set of examples is fed to a trained RBM model, after each example has settled to equilibrium following several oscillations between visible and hidden units, the reconstructed example should adhere to the training data distribution. We initially considered that any random input vector would converge to be similar to an original example. Unfortunately, we discovered that for even low dimensional data, this approach fails to generate accurate PEs based on tests using the MMD measure. The method generated examples concentrating in the centre of each high probability region of the training data distribution, but do not capture the character of the full distribution. For this reason, the application of this approach will not be reported in this paper.

Reconstruction Error from a Trained RBM: An accurately trained RBM can reconstruct a training example with low MSE. This means, after one oscillation of feeding a uniform random set of examples into an RBM model, the best pseudo examples (those most adhering to the distribution of the training examples) are those with the lowest reconstruction error. The lower the error, the higher the probability that the example is from the original training distribution. However, because not all randomly generate examples adhere to the training data distribution after one oscillation, we must select those examples based on some metric and tolerance level.

4.2. Selection of Metrics and Tolerance Levels

We considered two selection metrics: Euclidean distance, Mean Squared Error (MSE). The Euclidean distance between the uniform random input example and its corresponding reconstruction was calculated and if the distance was less than a defined tolerance, δ, the example was considered a PE. Similarly, the MSE between the uniform random input example and its corresponding reconstruction was calculated and if the error was less than a defined tolerance, ϵ, the example was considered a PE. In this way, a collection of pseudo examples for prior tasks can be generated for rehearsal when consolidating a new task into the long-term network. The mean Euclidean distance and MSE for the RBM training data reconstruction was used as the initial tolerance levels for δ and ϵ.

The probability density functions for the actual training examples and the generated PEs from a RBM trained model are shown in Figure 1. The blue probability density function (pdf) represents the one-dimensional training examples. The red pdf represents the 700 selected pseudo examples from the reconstruction of a uniform random set of 3000 examples using a trained RBM model and the mean Euclidean distance metric. The figure shows that the selected pseudo examples adhere to the input regions in the training data. So we conclude that, the pseudo examples match the training data input variable distribution.

4.3. Autoencoder-based Divergence Measure:

To evaluate the success of the adherence of the PE distribution to the training data distribution, we require some measure of similarity over the sets of examples. The MMD measure can be used but it requires retaining all of the training examples, which would defeat the purpose of generating PEs. Instead, we defined a measure that requires saving only the reconstruction error of the trained RBM model on the training set after each consolidation.

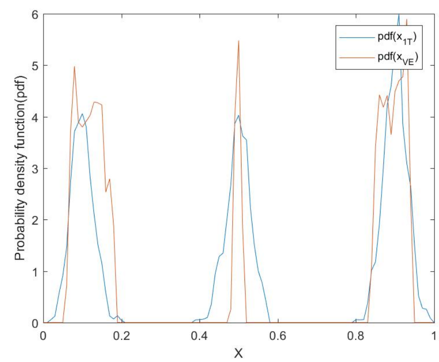

Figure 1: Probability density functions of actual training examples and selected pseudo examples.

We considered an unsupervised autoencoder approach to measure the difference between two probability distributions. If our model has been accurately trained, we can measure the relative degree of similarity between a reconstructed test set and the original training set probability distributions based on the ratio of their reconstruction error. We defined the Autoencoder-based Divergence Measure (ADM) as follows: Let, MSE_{TRN} = the MSE of the consolidation training data on the RBM model and MSE_{TST} = the MSE of the test data given to the trained RBM model, then $ADM = \frac{MSE_{TST}}{MSE_{TRN}}$.

Thus, if the test dataset is the training data set, then ADM = 1. And if the input is from a very different probability distribution compared to the training data distribution, then the MSE_{TST} is higher than MSE_{TRN}. So the value of ADM is much higher than 1.

We verified the ADM measure over multiple experiments using two-dimensional synthetic training and test datasets; including where we made various pathological changes to the test or training dataset in an effort to trick the measure. In all cases, the test data with very different probability distributions compared to the training data had higher ADM values than test data with small variations from the training dataset. To summarize, $0 < ADM < 1$ signifies the same probability distribution as the training data; $1 < ADM < 2$ signifies similar to or containing part of the training data distribution; $ADM > 2$ signifies an increasingly different distribution than the training data.

5. Empirical Study 1: Generating PEs for a Synthetic Domain of Tasks

The purpose of this experiment is to test the Reconstruction Error approach to generating PEs and compare the two metrics for selecting PEs: Euclidean distance and MSE. Three different synthetic tasks were created using a Gaussian generator: a one-variable (one-dimensional) task (see Figure 1), a two-dimensional task, and a four-dimensional task (see Figure 2). These tasks we consider a good starting point for examining the proposed methods. Each task had 1 to 3 Gaussians of random mean and variance spread over the range [0,1]. A one layer RBM autoencoder was trained on each task using 1000 training examples. Following this, uniformly distributed random data was fed into each trained RBM and the PEs were selected based on their reconstruction error. We report on the details of the most challenging four-dimensional task. See[1] for details on the hardware used.

Tolerance Level Measured by MSE Metric: As per Section 4.2, one measure for selecting the initial tolerance level is the sum of squared error between the training example and its corresponding reconstruction after one oscillation. If the squared error falls below the MSE_{TRN} tolerance level the example is selected as a PE.

The four-dimensional synthetic training data set of 1000 examples produced a RBM with a mean reconstruction error of $MSE_{TRN} = 0.000395$. Some 80 PEs were selected from the reconstructed set of 5000 uniform random examples passed through the trained RBM model. This generated a ADM = $\frac{MSE_{VE}}{MSE_{TRN}}$ = $\frac{0.000228}{0.000395}$ = 0.5772. As per Section 4.3, this value signifies that the pseudo examples are from the same probability distribution as the training data.

Tolerance Level Measured by Euclidean Distance Metric: As per Section 4.2, the other measure for selecting the initial tolerance level is the mean Euclidean distance between the training data and its corresponding reconstruction after one oscillation. If the difference falls below the MSE_{ED} tolerance level the example is selected as a PE.

The same training data used for the previous measure is used for this experiment as well. The four-dimensional training set of 1000 examples produced a RBM model with a mean Euclidean distance of $ED_{TRN} = 0.017452$. Some 52 PEs were selected based on the ED_{TRN} tolerance level from a reconstructed set of 5000 uniform random examples passed through the trained RBM model. The resulting ADM = $\frac{0.000165}{0.000395}$ = 0.4177, which signifies that the pseudo examples were from the same probability distribution as the training data. In Figure 2, the blue probability density functions (pdf) represent the training data. The red pdfs represent the PEs selected from the reconstructed set uniform random examples using the trained RBM model.

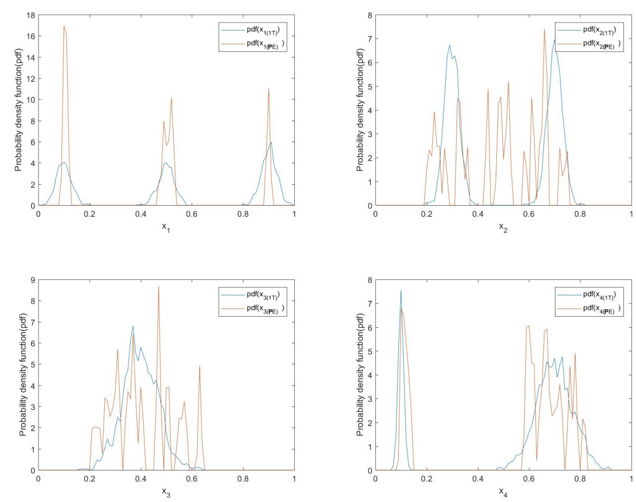

Figure 2: Exp 1 - Probability density functions for the training example variables $x_{1(1T)}$, etc and the selected pseudo example variables $x_{1(PE)}$, etc using the tolerance level measured by average Euclidean distance.

Discussion: For both the tolerance level measured by MSE and the tolerance level measured by Euclidean distance, the ADM showed that the selected PEs adhere to the training data distribution for one, two and four dimensional datasets. The PEs generated by the Euclidean distance-based method were further validated for their similarity to the original training set distribution. This was done by computing the MMD statistics: Value = 0.031646 and Bound = 0.077804. Therefore, the Null hypothesis that the two distributions are the same is accepted. In comparison the PEs generated by the MSE-based method did not quite meet the MMD requirement with Value = 0.042707 and Bound = 0.037726. For this reason we will proceed to favour the Euclidean distance-based method in the remaining studies.

6. Empirical Study 2: Generating PEs for Real-world Tasks

The purpose of this experiment is to test the Euclidean distance-based approach to generating PEs for a real-world task. The real-world dataset deals with the diagnosis of a

[1] All experiments were conducted on a Dell desktop with a 3.4 GHz Intel Core i7 processor and 16 GB of memory running Windows 8.1. The only bottleneck was is memory when the MMD metric was used to check the similarity of the probability distributions of two sets of examples. Fortunately, this is only done at the end of runs to validate our ADM metric.

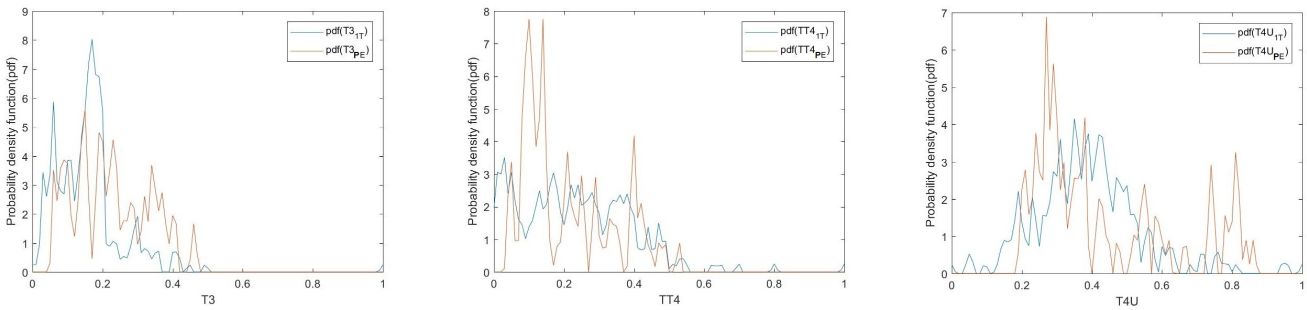

Figure 3: Exp 2 - Probability density functions for the training examples $T3_{1T}$, $TT4_{1T}$, $T4U_{1T}$ and the selected PEs $T3_{PE}$, $TT4_{PE}$, $T4U_{PE}$ using the tolerance level measured by mean Euclidean distance.

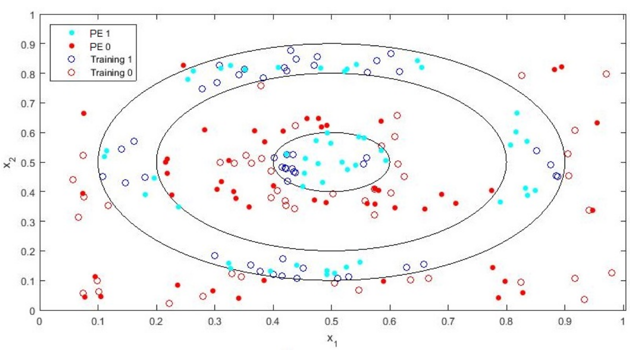

Figure 4: Exp 2 - Task A and its actual training and generated pseudo examples.

thyroid disorder named *hypothyroidism* [4]. The dataset contains 3163 patient examples and 26 attributes including the classification value. All records with missing values were discarded leaving 274 examples. A decision tree was built using the dataset and the following three attributes were considered the most important for model development to predict the binary diagnostic outcome: T3, TT4, and T4U. To make the dataset balanced, we selected all of the examples having positive target values and then randomly selected an equal number of examples having negative examples. We used min-max scaling to normalize the selected attributes to the range [0,1]. The resulting Hypothyroid dataset had 274 balanced examples with 3 input attributes. The target classification had the value of 1 for the presence of hypothyroidism and value of 0 for the absence of the condition.

Tolerance Level Measured by Euclidean Distance: A one layer RBM autoencoder was trained using all 274 training examples. Following this, 2000 uniformly distributed random examples were fed into the trained RBM and the PEs were selected based on their reconstruction error. For this experiment, the tolerance level was set at ED_{TRN} = 0.033149, the mean Euclidean distance between the training data and its corresponding reconstruction after one oscillation. Some 91 PEs with variables T3, TT4, and T4U were selected from the reconstructed set of examples. This generated an ADM = $\frac{0.000504}{0.001876}$ = 0.2687, which signifies that the pseudo examples were from the same probability distribution as the training data.

In the Figure 3, the blue probability density functions (pdf) represent the training data. And the red pdfs represent the 92 pseudo examples selected from the reconstructed set of 2000 uniform random examples after one oscillation using a trained model.

Discussion: As can be seen in Figure 3, the MSE-based ADM measure used to select the PEs from the reconstructed set of uniform random examples passed through the RBM model come close to the probability distribution of the original training examples. This was validated by computing the MMD statistics: Value = 0.002513 and Bound = 0.009934. Because the value is less than the bound, the Null hypothesis that the two distributions are the same is accepted.

7. Empirical Study 3: Consolidation Using PEs for a Synthetic Task Domain

The purpose of this study is to test the usability of PEs for a prior Task A when consolidating a related new Task B into a Multiple Task Learning (MTL) neural network trained on task A. We expected the PEs generated by our method for Task A can be used to maintain the functional accuracy of that task while integrating in the examples for new Task B. Further we expect the level of accuracy for both tasks at the end of consolidation is the same as a model develop using the actual training examples for both tasks from scratch in a MTL network.

The first task, **Task A**, is a 2-variable Boolean concept task whose examples are classified as follows: define $D = sqrt((x_1 - 0.5)^2 + (x_2 - 0.5)^2)$; if $D \leq 0.1\ OR\ (D \geq 0.3\ AND\ D \leq 0.4)$, then the target value is 1, otherwise

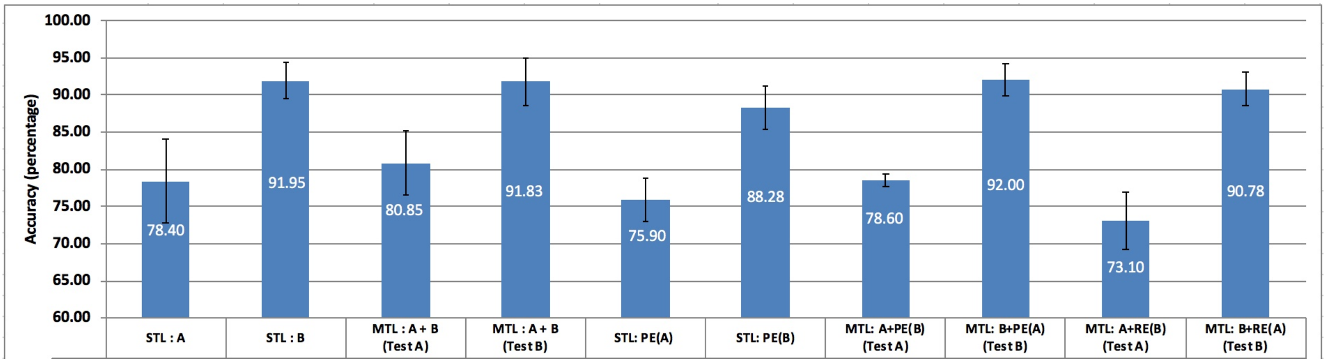

Figure 5: Exp 3 - Average classification accuracy on Task A and Task B.

it is 0. Task A and its training examples and generated PEs are shown in Figure 4.

Similarly, the second task, **Task B**, is Boolean concept task whose examples are classified as follows: define $D = sqrt((x_1 - 0.6)^2 + (x_2 - 0.6)^2)$; if $D \leq 0.05\ OR\ (D \geq 0.2\ AND\ D \leq 0.3)$, then the target value is 1, otherwise 0.

We created and experimented with nine different models for Task A and B. For each of the datasets, we built a MTL network with one hidden layer of 19 nodes. All of the models used learning rate of 0.05, momentum of 0.9 and 500,000 epochs. Each training sets had 80 examples and each validation set had 20 examples. All test sets had 1000 instances which made for good estimates of the generalization accuracy.

Nine Models Trained and Tested: The single task learning (STL) models **STL:A** and **STL:B** were trained using only the actual examples of Task A and B, respectively. **STL:PE(A)** and **STL:PE(B)** were trained using the PEs generated for Task A and Task B, respectively. For all pseudo example models, a dataset PE(A) was developed by passing the generated pseudo examples from a trained RBM model through the Task A target function. The pseudo examples were generated from the trained model using the Euclidean distance measure. The **MTL:A+B** model was trained using an equal number of actual examples for Task A and B. The **MTL:B+PE(A)** model was trained using an equal number of PE(A) examples and actual examples of Task B. Similarly, the **MTL:A+PE(B)** model was trained using an equal number of PE(B) examples and actual examples of Task A. For the Random Example models, a dataset RE(X) was developed by passing a set of uniform random examples through the Task X target function. The **MTL:B+RE(A)** models were trained using an equal number of RE(A) examples and actual examples of Task B. Similarly, **MTL:A+RE(B)** were trained using an equal number of RE(B) examples and actual examples of Task A.

Results: Four runs were completed for each model and a hypothesis t-test comparison was made with the baseline STL models. Figure 5 shows that the STL models using the PEs performed reasonably well compared to the STL models using the actual examples. As expected, the MTL:A+B model classification accuracy on Task A and B was as good as or better than their STL models. The MTL:A+PE(B) and MTL:B+PE(A) models were statistically as accurate as the MTL:A+B models. The MTL:A+PE(B) models were statistically more accurate than MTL:A+RE(B) (p-value = 0.02652 and the MTL:B+PE(A) models showed insignificantly higher accuracy than MTL:B+RE(A). We conclude that using the PEs generated by our method are as good as the actual examples and better than random examples for this problem.

8. Conclusion and Future Work

Summary and Findings: Lifelong Continual Learning is the logical next step for machine learning because it considers learning a variety of tasks over a lifetime, building richer internal representation with each new example. A significant challenge in developing a CL system is retaining prior task knowledge while consolidating in new task knowledge. Prior research has addressed this CL retention problem through the efficient sweep rehearsal of pseudo examples of prior tasks while learning the examples of a new task within a Multiple Task Learning (MTL) network. The approach does a good job at over-coming catastrophic forgetting as long as the pseudo examples adhere to the probability distribution of the input variables of the original training examples. This paper seeks to improve the sweep task rehearsal approach to consolidation by finding a method of generating accurate PEs directly from the long-term consolidated MTL Deep Belief Network.

Our major findings are as follows: (1) Accurate PEs of prior tasks can be created by a generative RBM layer at the beginning of a deep MTL network, such that those PEs adhere to the probability distribution of the input variables of the original training examples. Such PEs can be reconstructed by passing uniform random examples to the visible layer of the RBM model and selecting only those with reconstruction error less than the mean RBM training error. (2) These PEs can work as well as the actual training examples for a prior task B when consolidating a new task A into an MTL network that has been previously trained on task B. (3) The Autoencoder Divergence Measure, for comparing the similarity of the probability distributions of two sets of examples based on there RBM reconstruction MSE, is introduced and shown to work well across several synthetic and real-world domains.

Future Work: We planned to do more extensive experimentation using large sequences of tasks on a variety of task domains to test the sweep task rehearsal approach while generating PEs based on the RBM reconstruction error method we have presented. In particular, we would like to empirically compare our approach against recent work on EWC by [9] and Pseudo-recursal by [1].

Our long-term goal is to develop a CL system that is able to selectively transfer knowledge from prior tasks when learning a new task within a short-term memory network and then, if sufficiently accurate, to consolidate this new task knowledge into a long-term memory network.

References

[1] Craig Atkinson, Brendan McCane, Lech Szymanski, and Anthony V. Robins. Pseudo-recursal: Solving the catastrophic forgetting problem in deep neural networks. *CoRR*, abs/1802.03875, 2018.

[2] Craig Atkinson, Brendan McCane, Lech Szymanski, and Anthony V. Robins. Pseudo-rehearsal: Achieving deep reinforcement learning without catastrophic forgetting. *CoRR*, abs/1812.02464, 2018.

[3] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, Jul 1997.

[4] Dheeru Dua and Casey Graff. Hypothyroid dataset. In *UCI Machine Learning Repository*, 2017. [Online; accessed January-2020].

[5] Arthur Gretton, Karsten M. Borgwardt, Malte J Rasch, Bernhard Schlkopf, and Alex J. Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13:723–733, 2012.

[6] Stephen Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, 11:23–63, 1987.

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

[8] Geoffrey E. Hinton. A practical guide to training restricted boltzmann machines. *Neural Networks: Tricks of the Trade, 2nd Edition, LNCS*, 7700:599–619, 2012.

[9] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Computing Research Repository*, abs/1612.00796, 2017.

[10] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. In *arXiv*, 2019.

[11] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6470–6479, Red Hook, NY, USA, 2017. Curran Associates Inc.

[12] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans- actions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.

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

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

[15] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph Lampert. icarl: Incremental classifier and representation learning. In *arXiv*, pages 5533–5542, 07 2017.

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

[17] Anthony Robins. Consolidation in neural network and in the sleeping brain. In *Connection Science*, volume 8(2), pages 259–276, 1996.

[18] Anthony Robins. Sequential learning in neural networks: A review and a discussion of pseudorehearsal based methods. *Intelligent Data Analysis*, 8:301–, 08 2004.

[19] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *NIPS*, 2017.

[20] Daniel L. Silver, Geoffrey Mason, and Lubna Eljabu. Consolidation using sweep task rehearsal: Overcoming the stability-plasticity problem. In *Advances in Artificial Intelligence: 28th Canadian Conference on Artificial Intelligence Association*, volume LNAI 9091, pages 307–322. Springer-verlag, 2015.

[21] Daniel L. Silver, Qiang Yang, and Li. Lifelong machine learning systems: Beyond learning algorithms. In *Lifelong Machine Learning: Papers from the 2013 AAAI Spring Symposium*, pages 49–55. AAAI, 2013.

[22] V. Srivasatava, S. Sampath, and D.J. Parker. Overcoming catastrophic interference in connectionist networks using gram-schmidt orthogonalization. *PLoS ONE*, 9(9), 2014.

[23] Chenshen Wu, Luis Herranz, Xialei Liu, Yaxing Wang, Joost van de Weijer, and Bogdan Raducanu. Memory replay gans: Learning to generate images from new categories without forgetting. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 5966–5976, USA, 2018. Curran Associates Inc.