Journal of Machine Learning Research 148:221-235, 2021 NeurIPS 2020 Preregistration Workshop

Towards a Unified Lifelong Learning Framework

Tanner Bohn TBOHN@UWO.CA
Xinyu Yun XYUN@UWO.CA
Charles X. Ling CHARLES.LING@UWO.CA
Department of Computer Science
Western University
London, ON, Canada

Editor: Editor's name

Abstract

Humans can learn a variety of concepts and skills incrementally over the course of their lives while exhibiting many desirable properties, such as continual learning without forgetting, forward transfer of knowledge, and learning a new concept with few examples. However, most previous approaches to efficient lifelong learning demonstrate only subsets of these properties, often by different complex mechanisms. In this preregistration submission, we propose to study the effectiveness of a unified lifelong learning framework designed to achieve many of these properties through *one* central mechanism. We describe this consolidation-based approach and propose experimental protocols to benchmark it on several skills, using grid searches over hyperparameters to better understand the framework. The results of these experiments demonstrate that many properties can indeed be achieved by the framework, which also illuminating directions for improving the framework and evaluation.

Keywords: Lifelong learning, Human-inspired learning, Weight consolidation, Neural networks

1. Introduction

The past decade has seen significant growth in the capabilities of artificial intelligence. Deep learning in particular has archived great successes in medical image recognition and diagnostics (Litjens et al., 2017; Shen et al., 2017), tasks on natural language processing (Radford et al., 2019; Devlin et al., 2019), difficult games (Silver et al., 2017), and even farming (Kamilaris and Prenafeta-Boldú, 2018). However, deep learning models almost always need thousands or millions of training samples to perform well. This is in a sharp contrast with human learning, which normally learns a new concept with a small number of samples. Other major weaknesses in current deep learning, when compared to human learning, include difficulty in leveraging previous learned knowledge to better learn new ones (and vice versa), learn many tasks sequentially without forgetting previous ones, and so on.

Lifelong learning (LLL) (Thrun, 1998; Thrun., 1995), also known as continual (Parisi et al., 2019) or sequential learning (McCloskey and Cohen, 1989), is one research area concerned with flexible and efficient learning and the transfer of skills across long sequences of tasks. In this work, we consider the LLL setting of *task-incremental* classification, where batches of data for new tasks arrive sequentially. That is, a sequence of $(T_1, D_1), (T_2, D_2), ...$ are given, where D_i is the labeled training data of task T_i (from the space of tasks $\mathcal{T}$), and an individual task consists of a set of classes to be learned. Classification models for $(T_1, T_2, ..., T_k)$ must be functional before (T_{k+1}, D_{k+1}) arrives. This models the incremental process of human lifelong learning. The particular set of desirable LLL properties we are concerned with include the following:

- **Continual learning and testing**: Before starting to learn a new task T_j, a LLL approach should be able to perform well on all $T_{i<j}$. While learning the new task T_j, LLL should minimize the use of data $D_{<j}$. This is in contrast to standard multi-task (batch) learning, where all data of all tasks are used for training at the same time. This continual learning condition ensures that the model is 1) useful, since each task must be learned to an acceptable performance level, 2) flexible, in that new tasks can be continually accommodated, and 3) efficient, in that tasks are learned with high computational and data efficiency. For example, if T_1 requires learning to classify images of "0" vs. not "0" (see Section 3), acceptable performance on this task should be reached before moving onto T_2 of "1" vs. not "1", which should also be learned to an acceptable level.
- **Non-forgetting**: This is the ability to avoid catastrophic forgetting (McCloskey and Cohen, 1989), where learning T_j causes a dramatic loss in performance on $T_{i<j}$. Ideally, learning T_j when using only the data of T_j would not affect $T_{i<j}$. For example, learning T_2 of "1" vs. not "1" should not cause performance on the previous task, "0" vs. not "0", to degrade. Due to the tendency towards catastrophic forgetting, non-lifelong learning approaches would require retraining on data for all tasks together to avoid forgetting. This may reduce computational and data efficiency.
- **Forward transfer**: This is the ability to learn new tasks, $T_{\geq i}$, easier and better following earlier learned tasks, $T_{<i}$, also known as knowledge transfer (Pan and Yang, 2009). Achieving sufficient forward transfer also enables **few-shot learning** of later concepts. For example, first learning to classify "0" vs not "0" should allow the later task of "O" vs. not "O" to be learned faster.
- **Non-confusion**: Machine learning algorithms often find the minimal set of discriminating features necessary for classification. Thus, when more tasks emerge for learning in our LLL setting, earlier learned features may not be sufficient, leading to confusion between classes. For example, to distinguish between "1" and "0", the learned model may identify straight stroke for class "1" and curved stroke for "0". The same features may then be used to classify "I" vs "O". However, if the model is tested on all tasks so far, the model may be confused between "1" and "I" as well as "0" and "O".

Most previous approaches can only demonstrate subsets of these human-like properties, often by different complex mechanisms. For example, existing lifelong learning techniques tend to use one or more of three types of mechanisms, each of which comes with their own drawbacks and hurdles De Lange et al. (2019). These mechanisms are based on replay, regularization, and dynamic architecture respectively. See Section 2 for reviews and comparisons of these mechanisms.

In this paper, we describe a unified framework with *one central mechanism* that meshes with additional mechanisms to seamlessly demonstrate many human-like lifelong learning properties. The central mechanism, weight regularization, controls the flexibility of network weights to direct the transfer of skills across tasks as well as prevents the forgetting of skills. It is also intended to support network expansion in efficiently accommodating new tasks. We primarily consider our framework as applied to deep neural networks, which have become popular in recent years, and are an attractive type of machine learning model due to their ability to automatically learn abstract features from data.

The questions to be answered by our empirical analysis as well as our hypotheses are as follows:

1. How well does task-difficulty-based network expansion, as described in Section 3.2, work to accommodate new tasks? We hypothesize that this type of expansion allows for learning new tasks to the same accuracy level as less efficient methods.

2. How well does controlling the flexibility of task-specific weights, as described in Section 3.3, work to reduce forgetting? We hypothesize that forgetting can be almost completely removed with high enough regularization.

3. How well does task-similarity-based skill transfer, as described in Section 3.4, work for enabling forward transfer? We hypothesize that this type of skill transfer mechanism will work better than when no transfer is allowed and when transfer is not controlled at all.

4. How well does pairwise confusion reduction, as described in Section 3.5, reduce confusion? We hypothesize that this mechanism can reduce confusion by the same amount as comparable methods while being less resource intensive.

2. Related Work

The mechanisms used to perform LLL tend to fall into three categories and often only demonstrate subsets of LLL properties previously discussed. The first mechanism, replay, commonly works by storing previous task data and training on it alongside new task data (Rebuffi et al., 2017; Isele and Cosgun, 2018; Chaudhry et al., 2019; Wu et al., 2019). As a result of its data and computation inefficiency, we consider it not to be a very human-like learning mechanism.

The second mechanism is regularization. This mechanism works by restricting weight changes (making them less "flexible") via a loss function so that learning new tasks does not significantly affect previous task performance (Kirkpatrick et al., 2016; Zenke et al., 2017; Chaudhry et al., 2018; Ritter et al., 2018; Li and Hoiem, 2017; Zhang et al., 2020). We use this mechanism as the basis for our unified framework. Compared to previous approaches, we propose to use regularization more flexibly and strategically. Instead of simply controlling weight flexibility to *retain* previous task performance, we leverage it to also encourage forward transfer (Section 3.4).

The third mechanism, dynamic architecture, commonly works by adding new weights for each task and only allowing those to be tuned (Rusu et al., 2016; Yoon et al., 2018; Xu and Zhu, 2018). This is often done without requiring previous task data and stops forgetting while also allowing previous task knowledge to speed up learning of the new task. While this mechanism is necessary for LLL of an arbitrarily long sequence of tasks (any fixed-size network will eventually reach maximum capacity), it should be used sparingly to avoid unnecessary computational costs. In Section 3.2 we describe how a dynamic architecture can be efficiently used to help achieve multiple LLL properties when combined with our central mechanism.

3. Methodology and Experimental Design

In this section we describe our unified framework. We start by introducing the central mechanism and in the rest of the section, discuss how to use the central mechanism and combine it with additional mechanisms to achieve the several desirable LLL properties described in Section 1. For each mechanism, we also describe the experimental protocol to evaluate it. Shared among the experimental protocols are the following settings:

Task. We will use the following binary classification task sequence with samples taken from the balanced EMNIST dataset (Cohen et al., 2017): $T_1 = (0 \text{ vs. not } 0)$, $T_2 = (1 \text{ vs. not } 1)$,..., $T_5 = (4 \text{ vs. not } 4)$, $T_6 = (A \text{ vs. not } A)$, $T_7 = (I \text{ vs. not } I)$, $T_8 = (O \text{ vs. not } O)$, $T_9 = (Z \text{ vs. not } Z)$. For tasks 1 to 5, "not x" means $\{0, 1, 2, 3, 4\} \setminus x$. For tasks 6 to 9, "not x" means $\{A, I, O, Z\} \setminus x$. This task sequence is a minimal case allowing for proof-of-concept experiments where we can be sure that there is a) clear room for forward transfer (e.g. from "0" to "O" or "1" to "I") and b) clear cases of confusion (e.g. between "0" and "O"). In a more complex task sequence it would be harder to verify whether the proposed mechanisms work as intended. For each character, we will use 50 training samples. Additionally, all results will be averaged across 20 random seeds.

Architecture and training. We will use a network architecture with two hidden layers with ReLU activation. The width of the layers for the first task is N_{max}. The Adam optimizer (Kingma and Ba, 2014) will be used, with the default hyperparameters provided by Keras (Chollet et al., 2015). We will use a a batch size of 64 and training for 5 epochs for each task.

3.1. A Central Consolidation Mechanism

We propose a LLL framework which situates a consolidation policy as the central mechanism. The consolidation policy works through a high-dimensional dynamic hyperparameter, $\boldsymbol{b}$, which separately controls the flexibility of *all* network weights. Each network weight thus has its own consolidation value specifying how easy (or hard) it is to modify the weight. Depending on the specific $\boldsymbol{b}$-setting policy used during training, we hypothesize that several desirable learning properties can be achieved. While the network weights are learned via back-propagation, $\boldsymbol{b}$ is set by a consolidation policy.

The consolidation mechanism ultimately works through dynamically modifying the loss function. If each network weight, θ_i, is associated with a consolidation value of $\boldsymbol{b}_i \geq 0$, the loss for the new task by itself, L_t, is combined with weight consolidation as follows:

$$L(\theta) = L_t(\theta) + \sum_i \boldsymbol{b}_i (\theta_i^t - \theta_i^{target})^2 \tag{1}$$

Here, θ_i^{target} is the target value for a weight to be changed to. This can be either its value before training of the new task, or zero, in the case where we explicitly want to prevent certain weights from being used. θ_i^t is the weight value being updated during training on task t. This loss has the following behaviour: a large $\boldsymbol{b}_i$ causes changing θ_i away from θ_i^{target} to be strongly penalized during training. When $\boldsymbol{b}_i$ is arbitrarily large, we refer to these weights as **"frozen"**, and simply fix them during training. In contrast, $\boldsymbol{b}_i = 0$ indicates that the weight is free to change, i.e. it is **"unfrozen"**. If $\boldsymbol{b}_i$ is arbitrarily large, we can consider θ_i to be masked during backpropagation and completely prevented from changing to improve efficiency.

As elaborated in the rest of this section, we distinguish between three types of weights, which have consolidation values and initialization methods corresponding to each. High-level details of these weight groups are in Figure 1. There are group B weights (blue), which are intended to be free to tune. There are group R weights (red), which contain previous task knowledge. Finally, there are group G weights (green), which can facilitate the transfer of knowledge between tasks.

3.2. Continual Learning of Classification Tasks

In LLL and human learning, we desire to learn new tasks after learning previous tasks. In real brains, this is supported by continually growing new neurons and connections between them (Winocur et al., 2012; Nelson and Alkon, 2015). Our framework similarly considers learning new tasks with the strategic use of network expansion.

To accommodate a new task, T_j, we propose to extend the width each layer of the neural network by N_j, an amount proportional to the estimated difficulty of the task.

To compute N_j, we first compute the maximum similarity to previous tasks. To compute the similarity between two tasks, $sim(T_i, T_j)$, we feed positive samples of the new task, T_j, into the network, and average the probabilities output by model for T_i. When the similarity between T_j and any previous task is high (i.e. the new samples are similar to those of a previous task), proportionally fewer nodes are added. That is, $N_j = N_{max}\,(1 - \max_{i=1,\ldots j-1} sim(T_i, T_j))$. In the extreme case where a new task is identical (or very similar) to a previous one, no new nodes (aside from the output) may need to be added.

When extending each layer of the network, the new column of nodes is connected as shown in Figure 1 (b) and (c). These group B weights are randomly initialized and have $\boldsymbol{b}$ values of $B_{\boldsymbol{b}} = 0$, so that they are free to tune. As outlined in the pseudo-code in Algorithm 1, after extending the network and performing steps corresponding to components of our proposed framework to be discussed next, training can be performed using only samples of the new task.

Experimental design. We need to demonstrate that tasks are learned to the same competency as though they were trained with a larger number of nodes added. The evaluation metric consists of computing the AUC for each task, and averaging across tasks after all tasks have been learned. We will try a range of values for $N_{max} : \{0, 10, 50, 100\}$. Baselines consist of adding the following constant widths: $\{0, 10, 50, 100\}$. For these tasks, we will use $R_{\boldsymbol{b}} = inf$ (i.e. frozen) and disable the forward transfer mechanism (introduced in Section 3.4) so that all transfer links are initialized to non-zero values, but no weight-copying is done. In addition to task performance, we will also report the model size, as a fraction

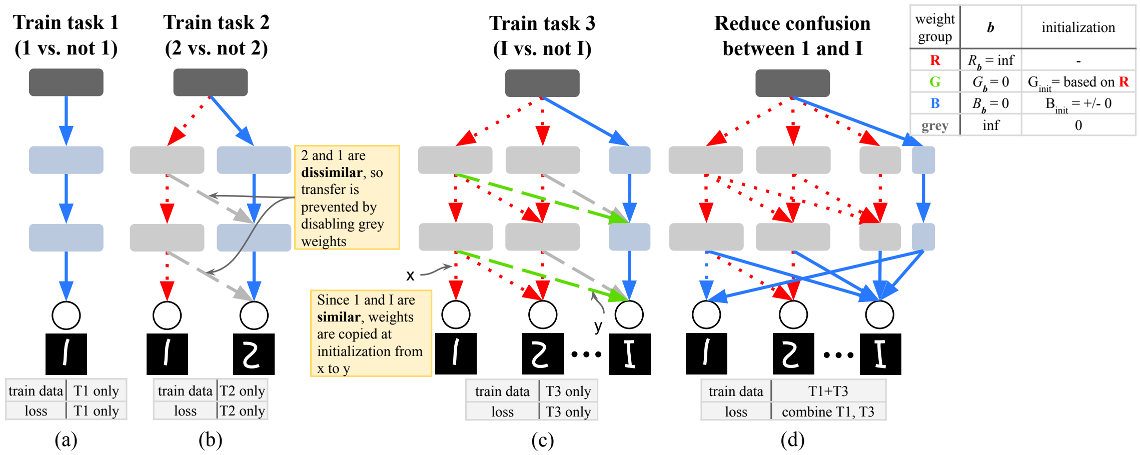

Figure 1: An example of applying the several mechanisms of our LLL framework. In step (a), task 1 is being trained. All weight here are in group B (randomly initialized and free to tune). In step (b) task 2 is being learned without forgetting task 1. Group R (red) weights can be frozen to prevent forgetting of T_1. Since 2 is dissimilar from 1, transfer can be prevented by disabling the transfer weights (grey). In (c), task I is learned while adding fewer nodes due to similarity with task 1. High dissimilarity from task 2 means that the corresponding forward transfer links are disabled. In (d), confusion is being reduced when 1 and I are confused, with additional nodes added when necessary.

Algorithm 1: Combining Framework Skills

```
// Given that tasks T_1, ..., T_{k-1} have been learned
Extend width of network proportional to task difficulty as described in Section 3.2;
Set consolidation values for non-forgetting of previous tasks as described in Section 3.3
    // see group R weights in Figure 1
Initialize weights from earlier units to newly recruited units as described in Section 3.4
    // see group G weights in Figure 1
Train the new task T_k to minimize Eq. 1 // only on the data of new task T_k
Perform confusion reduction as described in Section 3.5;
```

of the model size when the same N_{max} value with constant expansion is used. We expect that the performance will be roughly the same for both dynamic expansion and constant expansion for a given maximum expansion amount.

3.3. Non-Forgetting

Maintaining performance on previous tasks while learning new ones is the primary difficulty of LLL. In our framework, we can design consolidation policies to ensure that this is achieved while the new task is learned with the data for the new task only.

An intuitive way to prevent forgetting is by using a larger $\boldsymbol{b}$ value for weights which most influence the loss of a trained model (Kirkpatrick et al., 2016). To ensure non-forgetting during new-task training, we thus propose to set $R_{\boldsymbol{b}}$ such that the previous-task weights are frozen/near-frozen.

Experimental design. We need to demonstrate that the *learned AUC* (the AUC on a task when it is first learned) is close to the *retained AUC* (the AUC after all tasks have been learned). We can subtract the average learned task AUC from the average retained task AUC. Larger negative values indicate greater forgetting. This experiment will be run with a fixed width extension value of 50. We will try the following range of $R_{\boldsymbol{b}}$: $\{0, 1, 10, 100, 1000, inf\}$. We will also leave the forward transfer mechanism disabled. We expect the larger consolidation values to provide better non-forgetting in comparison to the baseline value of 0 when no consolidation is applied.

3.4. Forward Transfer

While non-forgetting ensures task performance is maintained over time, previous tasks do not "help" learning new tasks, a concept prominent in multi-task and transfer learning (Pan and Yang, 2009; Zhang and Yang, 2017), and appears in LLL as "forward transfer".

We propose to achieve positive forward transfer with our framework by controlling the transfer of skills between tasks. This skill transfer is mediated by the dashed links in Figure 1. When these weights are "disabled" (initialized to 0 and frozen – see grey links in Figure 1(b), (c)), the past task is unable to influence the new task. When they are "enabled" (unfrozen and initialized to non-zero values – see group G weights in Figure 1), features learned by past tasks can be quickly reused when learning a new task. When a task might lead to negative transfer for the new task, the transfer weights would be disabled, and when positive forward transfer is expected, they are enabled. Group G weights have a consolidation of $G_{\boldsymbol{b}} = 0$, allowing them to be freely tuned.

In an attempt to more directly leverage previous knowledge, we identify the most similar task to the new one, and copy the output layer weights from that task to the new task weights, as shown in Figure 1(c). All other G weights are randomly initialized. We use a simple technique to decide when to allow transfer: if the similarity, $sim(T_i, T_j)$ (discussed in Section 3.2) is above a certain threshold, $\alpha \in [0, 1]$, then enable the transfer links and copy the similar-task weights, otherwise disable them.

This idea of selectively sharing knowledge between tasks is conceptually shared by GO-MTL (Kumar and Daume III, 2012), which learns in a non-continual fashion. GO-MTL computes task similarity by first learning a separate model for each task and then looking at the similarity between learned weights. A sparse matrix representing transferability of knowledge between tasks is computed. A LLL approach by (Raghavan et al., 2020) selectively transfers skills from "canonical tasks" to a new task. The amount of transfer is based on the likelihood that new task samples would be generated by a generative network learned by each canonical task.

Experimental design. We need to demonstrate that tasks are learned to a greater competency with the selective forward transfer mechanism enabled than without. We can thus subtract the learned AUC when the mechanism is disabled (all transfer weights randomly initialized) from the learned AUC when it is enabled. Larger positive values indicating greater forward transfer. The experiment will use $R_{\boldsymbol{b}} = inf$, fixed width expansion, and a grid search over $N_{max} = \{0, 10, 50, 100\}$ and $\alpha = \{0, 0.25, 0.5, 0.75, 1.0\}$. We expect that the forward transfer mechanism will have a greater contribution when N_{max} is smaller (with larger networks, the effect of forward transfer is likely smaller) and when $\alpha = 0.5$ (this is a guess at the threshold below which negative transfer would occur).

3.5. Non-Confusion

When a new task such as "O vs not O" is similar to a previous one, "0 vs. not 0", we leverage this fact to learn the new task faster, as described in the previous subsection. However, since both tasks are learned without observing samples of the other, confusion may occur when we present an O or 0 to the model.

We propose to resolve such confusion in a pairwise manner, as step 5 in Algorithm 1. This process uses stored prior task samples (Mem each) to compute entries of the confusion matrix corresponding to confusion with the new task, T_j. Whenever confusion occurs between T_i and T_j at a rate greater than some threshold, $\gamma \in [0, 1]$, we can simultaneously fine-tune the last-layer weights of T_i and T_j on samples of the confused tasks. This is done by adding a temporary softmax output and minimizing the categorical cross-entropy loss when classifying positive samples of both classes. When this is insufficient to reduce confusion to below γ, we can expand the model by a constant amount, $N_{confused}$, with type B weights and repeat. This step is reflected in Figure 1(d).

Experimental setup. We can determine effectiveness by observing the confusion, as measured by the average task recall error when the task ID is not provided. When this value is larger, it indicates greater confusion. This experiment will be run with the following fixed values: $N_{max} = 100$, $R_{\boldsymbol{b}} = inf$, $\alpha = 0.5$, and $Mem = 20$. We will run a grid search over $\gamma = \{0.1, 0.2, 0.3\}$, and $N_{confused} = \{0, 10, 50\}$. For baselines, we can consider when no confusion reduction is performed ($\gamma = 1$) and when no expansion for confusion is performed ($N_{confused} = 0$). We expect that smaller values for γ will reduce confusion, especially with larger values of $N_{confused}$.

4. Experimental Results

Through the four sets of experiments described previously, we found that in general the results agree with the predictions. Several details of the results differ from our expectations however, which may serve to design a more effective framework and more insightful experiments in future work. The results for continual learning will be described in Section 4.1. Non-forgetting results will be described in Section 4.2. Forward transfer results will be described in Section 4.3. Non-confusion results will be described in Section 4.4.

Final experimental setup The following changes and edits were made to the proposed experimental setup. First, instead of averaging across 20 random seeds, we found that some results were noisier than expected, motivating an increase to 30 random seeds for all experiments. Second, for the continual learning experiments, we realized that it does not make sense to use $N_{max} = 0$ (since you cannot have a neural network of width zero), so the minimum value we consider is 10. Third, for N_{max} during the non-confusion experiments, we used 50 rather than 100, so as to agree with N_{max} for the non-forgetting experiments. Fourth, the algorithm used for deciding which weights to freeze and unfreeze for confusion reduction differs slightly from Figure 1d: when reducing confusions between previous tasks and the newest task, *all* weights added for the newest task should still be unfrozen (since changing these weights does not affect previous tasks). In addition to the analyses originally described, we include closer examinations of some of the results in Sections 4.3 and 4.4.

4.1. Continual Learning

The goal of this set of experiments is to determine whether difficulty-based expansion is an effective way of reducing total model size without sacrificing performance. From Table 1 we can see that difficulty-based expansion (DBE) reduces the number of parameters to between 50% and 69%, depending on the value of N_{max}. While LA is lower when using DBE, the difference is only around 0.7 when $N_{max} = 100$.

N_max	LA (const. expand)	LA (DBE)	DBE param. pct.
10	72.1	65.0	50%
50	95.9	92.7	69%
100	98.3	97.6	63%

Table 1: Continual learning results. DBE param. pct. is the percentage of parameters that are use when applying DBE vs. when using constant expansion. We can see that DBE does indeed reduce total network size, but at the cost of small decreases in performance.

These results suggest two questions for future research. First, how does the estimated difficulty of tasks depend on the number of weights used for previous tasks? Second, what is the relationship between task difficulty and performance with a given number of nodes? Answering these questions would allow for designing a better DBE strategy.

4.2. Non-Forgetting

Addressing the question of how well larger $R_{\boldsymbol{b}}$ values prevent forgetting, Table 2 shows that when $R_{\boldsymbol{b}} = 0$, forgetting is indeed the highest, and quickly drops as the value increases. As expected, the amount of forgetting is exactly zero when weights are fully frozen (masked during training).

4.3. Forward Transfer

The goal of the forward transfer experiments is to determine to what extent allowing knowledge transfer only from similar tasks helps performance.

From Table 3 we can see that for both DBE and constant expansion and all values for N_{max}, a similarity threshold of $\alpha = 0.75$ is optimal or near-optimal (in contrast to our initial guess of $\alpha = 0.5$ being optimal). Interestingly, we find a greater amount of forward transfer when using DBE than constant expansion. This may be a result of the

$R_{\boldsymbol{b}}$	RA - LA
0 (unfrozen)	-0.89±0.21
1	-0.03±0.03
10	0.00±0.02
100	0.02±0.03
1000	0.03±0.03
infinity (frozen)	0± 0

Table 2: Non-forgetting results. We found the results to vary considerably with $R_{\boldsymbol{b}} = 0$ and thus include 95% confidence intervals, estimated over the 30 trials.

fact that greater forward transfer is seen with smaller networks than larger ones (DBE leads to smaller networks).

	const. expansion			DBE		
α	$N_{max} = 10$	$N_{max} = 50$	$N_{max} = 100$	$N_{max} = 10$	$N_{max} = 50$	$N_{max} = 100$
0	6.4	0.8	0.4	9.3	1.3	0.4
0.25	14.2	1.7	0.5	18.4	3.4	0.5
0.5	17.0	1.8	0.5	23.1	3.8	0.6
0.75	17.6	1.8	0.5	23.3	3.7	0.6
1.0	16.2	1.7	0.4	21.5	3.4	0.4

Table 3: Forward transfer results. Matching our expectations, the benefit is greater for smaller values of N_{max}. The ideal value for α differs slightly from our prediction (0.75 instead of 0.5).

Figure 2 shows the estimated similarities of previous (source) tasks to new (target) tasks computed just before learning the target task. We find that, justifying the construction of our task sequence, the most similar task pairs are indeed "0" and "O", "1" and "I", and "2" and "Z". While these similarity values are below 0.5 on average, this value varies between trials (for example the "0"-"O" similarity is above 0.5 roughly 23% of the time). From Figure 3 we can see that the forward transfer mechanism (using $\alpha = 0.5$, $N_{max} = 10$, and constant expansion) has a positive effect on all tasks after the first one. Given that for many tasks, no previous task will have a similarity > 0.5, these results suggest that avoiding transfer from dissimilar tasks is very effective at improving task performance.

4.4. Non-Confusion

The final experiment aims to determine the effectiveness of the described two-step pairwise confusion reduction mechanism.

From Figure 4 we can see that, in line with our expectation, small confusion threshold (γ) values lead to smaller overall confusion levels, with the average task recall error reduced from

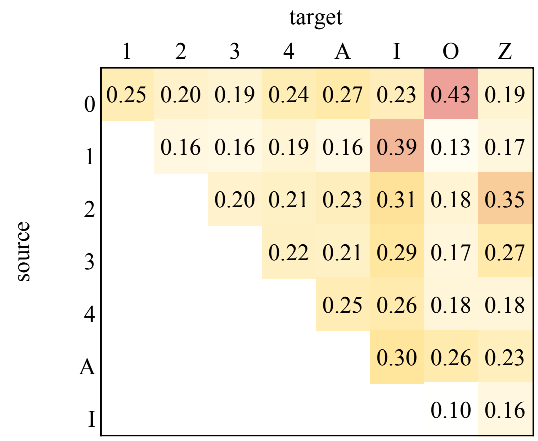

Figure 2: Similarities between previous (source) tasks and the task to be learned next (target). These values are estimated with target task training data. These particular values were computed when using $\alpha = 0.5$, $N_{max} = 10$, and using constant expansion.

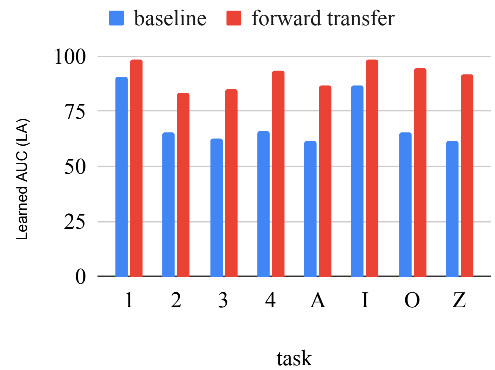

Figure 3: The LA of individual tasks with and without the selective forward transfer mechanisms enabled (using the same settings as for Figure 2).

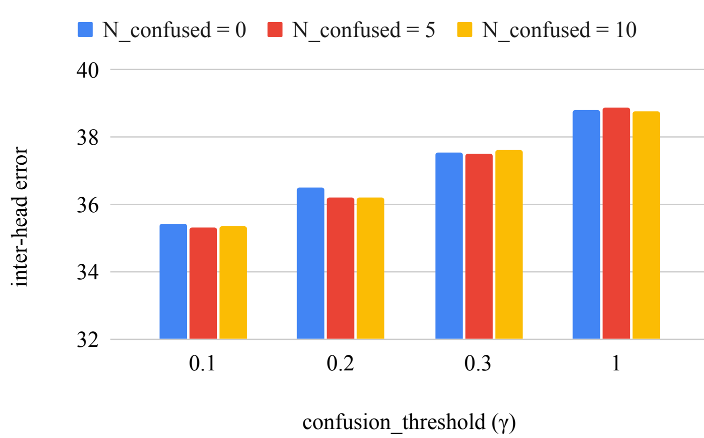

Figure 4: Non-confusion results. We can see that the threshold for confusion reduction has the largest effect on average task confusion, while the amount of expansion when trying to reduce more difficult confusions has no clear effect.

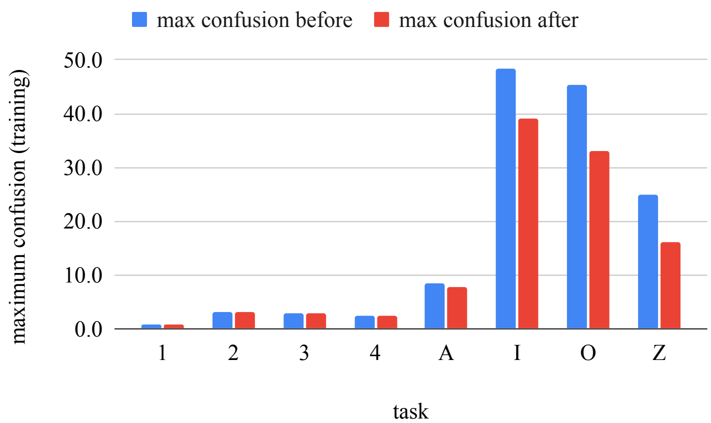

Figure 5: The maximum confusion values during training between each task and those before it (when using $\gamma = 0.1$ and $N_{confused} = 10$).

around 39 points to 35. From the confusion reduction only averaging around 4 points, we can guess that some of the task confusions are especially difficult to reduce (e.g. samples of "0" and "O" are often indistinguishable even to humans). Additionally, while we predicted that greater expansion for confusion-reduction ($N_{confused}$) would improve results, we do not see any clear pattern. This may suggest that either the present network capacity is sufficient to reduce confusion as much as it possibly can be, that the considered expansion amounts are insufficient, or that the remaining confusions are unable to be reduced at all due to overly-similar tasks.

When we examine the maximum amount of confusion experienced during learning by each task (shown in Figure 5), we see that the first several tasks are not subject to much confusion (as also suggested by Figure 2). However, for the last three tasks (which have high similarities with previous tasks), there is significant confusion which is able to be reduced.

5. Conclusions

In this work, we introduced a lifelong learning framework based on varying consolidation of different groups of weights over the course of learning. This framework was designed to demonstrate several properties: the ability to continually learn tasks to high performance while reducing parameter count, not forgetting previous tasks, leveraging previous task knowledge to improve new task learning, and reduce confusion between similar tasks.

Our proof-of-concept experiments demonstrated that DBE was able to reduce total model size without significantly affecting performance. Results on non-forgetting also confirmed that increasing consolidation strength of previous task weights (and at the extreme, freezing them) prevented forgetting. We also found that the forward-transfer mechanism based on selective weight copying and initialization performed surprisingly well, especially with smaller networks. Finally, we saw that while confusion between our chosen tasks could be reduced on average, the total confusion reduction was rather small, and the amount of network expansion during confusion reduction did not have a clear effect.

Reflecting on these results, there are several promising directions for improving the design and evaluation of the framework. This includes developing a more refined understanding of the relationship between task difficulty and network capacity required to reach a given performance, designing experiments to understand the contributions of the individual aspects of the forward transfer experiment, and performing more extensive testing on confusion reduction to understand what is required to reduce difficult confusions.

References

Arslan Chaudhry, Puneet Kumar Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. *CoRR*, abs/1801.10112, 2018. URL http://dblp.uni-trier.de/db/journals/corr/corr1801.html#abs-1801-10112.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. Continual learning with tiny episodic memories. *arXiv preprint arXiv:1902.10486*, 2019.

François Chollet et al. Keras, 2015.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. Emnist: an extension of mnist to handwritten letters. *CoRR*, abs/1702.05373, 2017. URL http://dblp.uni-trier.de/db/journals/corr/corr1702.html#CohenATS17.

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. *arXiv preprint arXiv:1909.08383*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Andreas Kamilaris and Francesc X Prenafeta-Boldú. Deep learning in agriculture: A survey. *Computers and electronics in agriculture*, 147:70–90, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

James Kirkpatrick, Razvan Pascanu, Neil 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, 2016. URL http://arxiv.org/abs/1612.00796. cite arxiv:1612.00796.

Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning. *arXiv preprint arXiv:1206.6417*, 2012.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *The psychology of learning and motivation*, pages 109–165, 1989.

Thomas J Nelson and Daniel L Alkon. Molecular regulation of synaptogenesis during associative learning and memory. *Brain research*, 1621:239–251, 2015.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

Aswin Raghavan, Jesse Hostetler, Indranil Sur, Abrar Rahman, and Ajay Divakaran. Lifelong learning using eigentasks: Task separation, skill acquisition, and selective transfer. *arXiv preprint arXiv:2007.06918*, 2020.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 5533–5542. IEEE Computer Society, 2017. ISBN 978-1-5386-0457-1. URL http://dblp.uni-trier.de/db/conf/cvpr/cvpr2017.html#RebuffiKSL17.

Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems*, pages 3738–3748, 2018.

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016. URL http://dblp.uni-trier.de/db/journals/corr/corr1606.html#RusuRDSKKPH16.

Dinggang Shen, Guorong Wu, and Heung-Il Suk. Deep learning in medical image analysis. *Annual review of biomedical engineering*, 19:221–248, 2017.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

S. Thrun. A lifelong learning perspective for mobile robot control. *Intelligent Robots and Systems*, 1995.

Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998.

Gordon Winocur, Suzanna Becker, Paul Luu, Shira Rosenzweig, and J Martin Wojtowicz. Adult hippocampal neurogenesis and memory interference. *Behavioural brain research*, 227(2):464–469, 2012.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning, 2019.

Ju Xu and Zhanxing Zhu. Reinforced continual learning. In *Advances in Neural Information Processing Systems*, pages 899–908, 2018.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *ICLR (Poster)*. OpenReview.net, 2018. URL http://dblp.uni-trier.de/db/conf/iclr/iclr2018.html#YoonYLH18.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In Doina Precup and Yee Whye Teh, editors, *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 3987–3995. PMLR, 2017. URL http://dblp.uni-trier.de/db/conf/icml/icml2017.html#ZenkePG17.

Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1131–1140, 2020.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*, 2017.