A Unified Framework for Lifelong Learning in Deep Neural Networks

Anonymous

Abstract

Humans can learn a variety of concepts and skills incrementally over the course of their lives while exhibiting an array of desirable properties, such as non-forgetting, concept rehearsal, forward transfer and backward transfer of knowledge, few-shot learning, and selective forgetting. Previous approaches to lifelong machine learning can only demonstrate subsets of these properties, often by combining multiple complex mechanisms. In this Perspective, we propose a powerful unified framework that can demonstrate all of the properties by utilizing a small number of weight consolidation parameters in deep neural networks. In addition, we are able to draw many parallels between the behaviours and mechanisms of our proposed framework and those surrounding human learning, such as memory loss or sleep deprivation. This Perspective serves as a conduit for two-way inspiration to further understand lifelong learning in machines and humans.

Keywords Lifelong learning · Human-inspired learning · Weight consolidation · Neural networks

1 Introduction

Humans have a sustained ability to acquire knowledge and skills, refine them on the basis of novel experiences, and transfer them across domains over a lifespan [1–3]. It is no surprise that the learning abilities of humans have been inspiring machine learning approaches for decades. Further extending the influence of human learning on machine learning, continual or lifelong learning (LLL) in particular [4], is the goal of this work.

To mimic human continual concept learning scenarios, in this paper we propose a new learning setting in which one concept-learning task is presented at a time. Each classification task and its corresponding labeled data are presented one at a time in sequence. As an example, assume we are given the task sequence of learning to classify hand-written characters of "1", "2", "3", and so on, by providing training examples of "1", "2", "3" in sequence (such as those in Figure 1). When learning "1", only data of "1" is available. When learning of "2", data of "2" becomes available, and so on.

Preprint. Under review.

We assume that these concepts are mutually exclusive (i.e. any sample can be a positive sample for at most one task). We also assume a set of "background negative" examples are given – they are presumably the previously learned concepts. This setting is in stark contrast to the standard setup in multi-class machine learning, where it is assumed that all training data of all classes is readily available, as a "batch" mode.

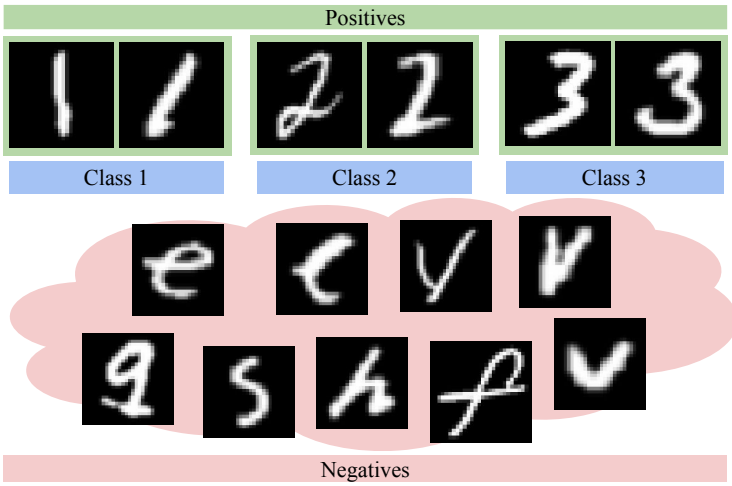

Figure 1: For the example task sequence in this section, we can consider the four classes to be "1", "2", "3", and "I". The negative samples used when learning to identify the three positive classes can come from a domain-appropriate non-overlapping "background" set (in this case, lower-case letters). After a model is trained up to class i, it will not have access to those training samples unless absolutely necessary (see later). The samples shown are from the EMNIST dataset [5].

Under this continual learning setting, we hope than a LLL approach would exhibit the following properties:

Non-forgetting: This is the ability to avoid catastrophic forgetting [6] of old tasks upon learning new tasks. For example, when learning the second task "2", with only training data on "2" (and without using the data of the task 1), "2" should be learned without forgetting how to classify task 1 learned earlier. Due to the tendency towards catastrophic forgetting, non-lifelong learning approaches would require retraining on data for "1" and "2" together to avoid forgetting. A skill opposite to non-forgetting is **selective forgetting**. As we will describe further, learning new tasks may require expansion of the neural network, and when this is not possible, the model can perform selective forgetting to free up capacity for new tasks.

Forward transfer: This is the ability to learn new tasks easier and better following earlier learned tasks. For example, after learning the task of classifying "1", it would be easier (requiring less training data for the same or higher predictive accuracy) to learn to classify "I". Achieving sufficient forward transfer opens the door to **few-shot learning** of later tasks.

Non-confusion: Machine learning algorithms need to find discriminating features for classification only as robust as they need to be to minimize a loss, thus, when more tasks emerge for learning,

earlier learned features may not be sufficient, leading to confusion between classes. For example, after learning "1" and "2" as the first two tasks, the learned model may say "with only straight stroke" is "1" and "with curved stroke" is "2". But when learning "I" as a later new task, the model may rely only on the presence of a straight stroke again, leading to confusion between "1" and "I" when the model is finally tested. To resolve such confusion between "1" and "I", samples of both "1" and "I" are needed to be seen together during training so that discriminating features may be found. In humans, this type of confusion may be seen when we start learning to recognize animals for example. To distinguish between common distinct animals such as birds and dogs, only features such as size or presence of wings is sufficient, ignoring finer features such as facial shape. However, when we next learn to identify cats, we must use the previous data on dogs and new data on cats to identify finer features (such as facial shape) to distinguish them.

Backward transfer: This is knowledge transfer in the opposite direction as forward transfer. When new tasks are learned, they may, in turn, help to improve the performance of old tasks. This is analogous to an "overall review" before a final exam, after materials of all chapters have been taught and learned. Later materials can often help better understand earlier materials.

Past works on LLL have only focused on subsets of the aforementioned properties. For example, an approach inspiring our own, Elastic Weight Consolidation (EWC) [7], focuses only on non-forgetting. The approach of [8] considers non-forgetting as well as forward and backward transfer and confusion reduction, but does not allow for selective forgetting. Figure 2 illustrates the scope of our framework compared with related previous approaches. Section 4 contains a more detailed comparison.

In this paper, we provide a general framework of LLL in deep neural networks where all of these these abilities can be demonstrated. Deep neural networks, which have become popular in recent years, are an attractive type of machine learning model due to their ability to automatically learn abstract features from data. Weights (strengths of links between neurons) of a network can be modified by the back propagation algorithms to minimize the total error of the desired output and the actual output in the output layer.

In our study, we consider fully-connected neural networks with two hidden layers to illustrate our LLL approach. The basic idea of our unified framework, similar to EWC [7], is to utilize "stiffness" parameters of weights during training phases to achieve the various LLL properties such as non-forgetting, forward transfer, etc. For each lifelong learning property, a subset of its weights may be "frozen", another subset of weights may be "free" to be changed, and yet another subset of weights may be "easily changed", depending on the type of lifelong learning properties we are aiming to facilitate at the time.

EWC and its conceptual successors [11–13] are lifelong learning approaches which estimate the importance of each weight in a network for maintaining task performance. By preventing already important weights from changing to accommodate new tasks (i.e. *consolidating* weights), catastrophic forgetting can be reduced. Generally speaking, each network weight, θ_i, is associated with a

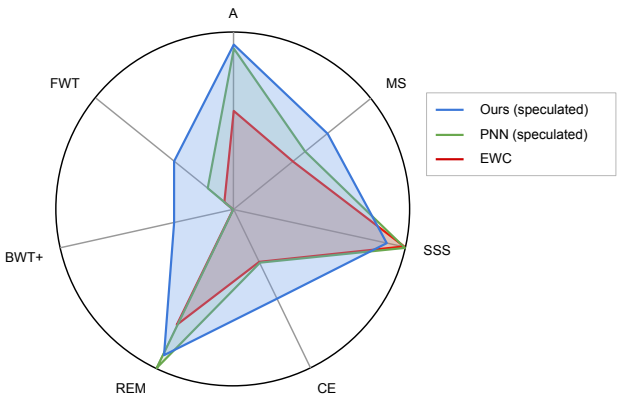

Figure 2: Spider chart: CL metrics per strategy proposed by [9] (larger area is better). A: accuracy, MS: model size efficiency, SSS: samples storage size efficiency, CE: computational efficiency, REM: remembering, BWT+: positive backward transfer, FWT: forward transfer. We expect our approach to be outperform the related approaches of EWC [7] and PNN [10] on a majority of the metrics.

consolidation value, b_i, which can be set or tuned for each stage of learning as we will soon discuss. When training a model with EWC, we combine the original loss L_t with weight consolidation as follows:

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

Here, θ_i^{target} is the consolidation target value for a weight, θ_i^t is the weight value being updated during training on task t, and λ is used to balance the importance of the two loss components. Clearly, a large b value indicates that changing the weight is strongly penalized during training, whereas a value of 0 indicates that the weight is free to change.

In our approach, we use three values for b to control the flexibility of different sets of network weights:

- b_{nf} for non-forgetting (ideally a very large value),
- b_{tr} for forward transfer (ideally very small or zero),
- and b_{free} for freely tunable weights (ideally very small).

While the individual weights of the network are learned via back-propagation, these consolidation hyperparameters are set by several heuristic strategies. We will illustrate how changing these hyperparameters to control the stiffness of weights in different parts of the deep neural networks, our approach can achieve all of the LLL abilities mentioned above (Section 2).

As we mentioned, these hyperparameters are determined during LLL by heuristic strategies, but one might wonder if these heuristics can be learned. Our comparison between lifelong learning of machines and humans suggests that our model hyperparameters are probably intrinsic to the

physiology of the brain – a product of natural evolution. A person can consciously perform meta-learning, such as during memory training and explicit rehearsal, whereas these heuristics may be explicitly learned or fine-turned. This would be our future study.

2 A Unified Framework for Lifelong Learning

To provide an intuitive understanding of our approach, we will describe how it would be applied to learning a sequence of hand-written numbers (such as "1", "2", "3") The training data at each step consists of positive examples for the class as well as samples from a pool of background negative examples, such as hand-written small letters. These background classes are different from the new classes to be learned and may be previously learned by the network. Figure 1) displays samples from these discussed classes.

For a neural network architecture, we will consider for illustrative purposes a simple feed-forward network with two fully-connected hidden layers trained with gradient descent. This approach conceptually extends to more complex architectures including Convolutional Neural Networks (CNNs), whereby it can be applied to improving the training and inference costs associated with many tasks that traditionally require large networks and long training times, common in computer vision tasks [14, 15].

As mentioned earlier, several heuristic strategies are used to set the hyperparameters in the deep neural network so that it can exhibit various LLL properties. However, the purpose of this section is not about those heuristics – they will be described in detail with experimental results in a follow-up technical paper [16]. Instead, we will describe the setting (or sizes) of those hyperparameters to illustrate their ability to demonstrate the desired LLL properties. Nevertheless, we list those heuristics briefly below:

- When a new task is being trained with data, its associated weight consolidation b values would be set to 0 or small to allow easy learning of the task;
- After a new task is learned, its associated b values should be set to be very large so that it will not to be "forgotten" or affected when other tasks are being learned;
- If a new task is similar (by some task similarity measure) to some old tasks, the b values of the forward transfer links from the old tasks to the new one should be set to 0 or very small, to encourage forward transfer of the representation and knowledge, realizing few-shot learning of the new task;
- If two tasks are confused, the b values of their associated weights should be set to be small so that confusion can be resolved by rehearsing on their training data. The b values of other tasks will be set to be very large so that other tasks will not be affected.
- If a certain task is not often used (which can be measured by the number of times it is tested or used in deployment), its associated b values will be gradually reduced. This amounts to

the controlled forgetting of the task while other new tasks are being learned. This is useful when the network reaches its size or computational capacity when it learns new tasks.

In Section 3 we will draw parallel in the setting of these hyperparameters in our model with human learning phenomena, such as memory loss and sleep deprivation.

2.1 Training Class 1

We assume that we start with a small or medium sized network for task 1. As discussed in the next step, this network size does not need to be fixed, and will increase as more tasks are learned. Alternatively, we can start with a very large network, and after learning each task, prune the network while maintaining performance [17]. For this simplified example, will use network expansion to illustrate our framework.

Training the network for class 1 is performed in the standard way: all weights of the neural network are free to modify. In Figure 3a, this is reflected as all the weights connecting *groups* of neurons in sequential layers being blue.

Training data consists of class 1 and background samples.

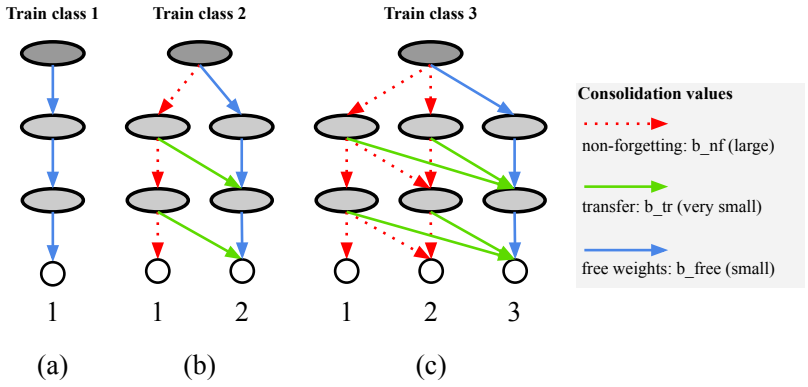

Figure 3: Weight consolidation values used during the first three training steps. In each step, weights for old classes are consolidated with b_{nf} (red), any applicable forward transfer weights are added and have a consolidation of b_{tr} (green), and weights among newly added nodes and the input and output have a consolidation of b_{free} (blue). When b_{nf} is large, the dashed red sets of weights are essentially frozen during training. Note that weights not shown have values fixed to be 0 and its b set to very large.

2.2 Training Class 2

In preparation for training class 2, to achieve non-forgetting of class 1, we increase the network capacity by introducing a set of free neurons for task 2. The number of new neurons is determined by the difficulty of task 2 as well as how similar task 2 is to the previous tasks. The more difficulty

the task 2, or the more dissimilar it is from the previous tasks, the more new neurons. This will be further discussed in our forthcoming paper [16].

We set the consolidation strength of all weights used for task 1 to be very large (as b_{nf}), while for task 2 to be small. This would translate into little influence on class 1's performance when training with back propagation on task 2, without using the task 1 data. This achieves non-forgetting for task 1. The strongly consolidated weights are reflected in Figure 3b with the red weights. The b values of weights between these newly added nodes and the input and output layers are set to be small so weights can updated via back-propagation (blue in Figure 3b). To encourage the forward transfer of skills from task 1 to task 2 if they are similar, we set the b of weights pointing from old nodes to new nodes to be very small (or 0), as b_{tr} (green).

By allowing forward transfer to occur from every previous task to the newest task, few-shot learning can occur to increasing extents, allowing new tasks to be learned with less data than previous tasks. Through leveraging curriculum learning [18] at both the level of task samples and perhaps individual tasks, the degree of forward transfer and few-shot learning capacity may be further improved.

During training of class 2, only the loss from the class 2 output needs to be back-propagated. As task 1 has been given and learned, we can mix in a small amount of class 1 samples with the background samples to be negative examples of class 2, to allow better discrimination of class 1 and 2. This would reduce the amount of confusions between tasks 1 and 2 that need to be resolved (see Section 2.4).

2.3 Training Class 3

To prepare for class 3 training, again apply a consolidation strength of b_{nf} to all weights used by previous classes and extend each layer with new nodes for learning class 3. This time, forward transfer connections are added from both class 1 and class 2 node groups to the new class 3 nodes, as pictured in Figure 3c. Again, during training of class 3, only the loss from the class 3 output needs to be back-propagated.

2.4 Confusion Reduction

By this point, the network may have learned how to minimally discriminate "1" and "2" by identifying if there is only straight stoke or not. As per the previous step, tasks 3 has also been individually learned well. If all classes were visually distinct, the network may perform well when being tested on all classes together, however, if we assume that task 3 is "I", making it visually similar to class 1, confusion may occur between these two classes.

That is, when "I" is presented in the input, both the output for "1" and output for "I" will be activated, indicating a classification of "1" or class "I", and thus, the prediction can be incorrect or unreliable. This is when confusion has happened. In this case, the data for class "1" and class "I" will be used to train the network to resolve confusion. We will use the network with its consolidation values as

in Figure 4a. If the existing network weights are insufficient to reduce the confusion, then additional weights can be added to the network.

Note that back-propagation of errors only need to be applied to outputs for the confused classes (classes "1" and "I" in this case). As consolidation of weights important to class "2" is set to be very large with b_{nf}, its performance should not be significantly affected while confusion is resolved between "1" and "I".

In practice, when individual tasks are being learned in sequence as described earlier, confusion can happen between any pair of tasks. To handle this, pair-wise confusion reduction as described here can be applied to resolve all pair-wise confusions, starting with the pairs with the highest confusion, and continuing until some stopping criterion is reached.

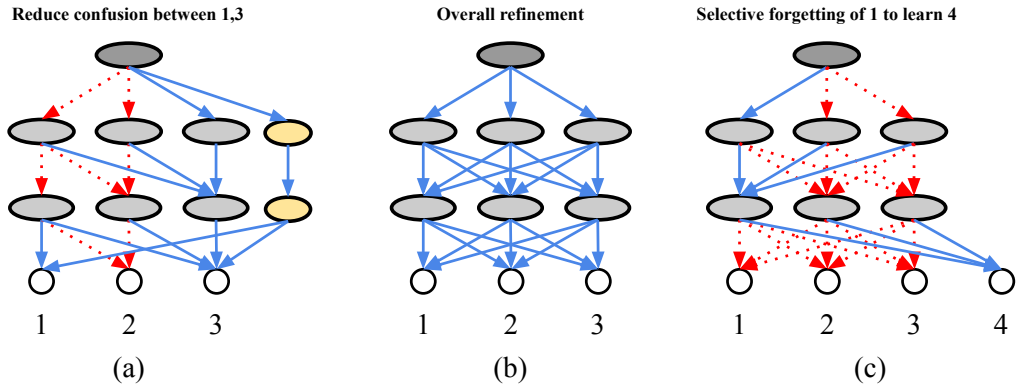

Figure 4: In (a): consolidation of weights during confusion reduction between e.g. classes 1 and 3. Note that weights important to preserving performance on class 2 are frozen with b_{nf}. To provide any additional capacity needed for resolving confusion, the column of yellow nodes can be added (and later grouped with the class 1 node column). In (b): consolidation of weights during backward transfer for overall refinement. Note the addition of backward transfer weights from 3 to 1&2, and 2 to 1. All weights are free to change while rehearsal is done on a fraction of training samples for all classes seen so far. This step facilitates further non-forgetting, forward transfer, confusion reduction, as well as backward transfer. In (c), selective forgetting of task 1 is performed so that an additional task 4 can be learned without the addition of many new weights. This will cause forgetting of task 1 gradually and may slightly affect those tasks relying on forward transfer from task 1 skills.

2.5 Backward Transfer

By this point in training: skills learned for earlier classes have been utilized by later classes (ex. 2 benefits from 1, 3 benefits from 1 and 2), but later skills have not been able to transfer to earlier classes.

To achieve such backward transfer of skills as well as overall refinement of the model, we can first initialize the backward transfer weights pointing from nodes originally added for each class to those

nodes for classes before it (i.e. from 3 to 1 and 2, from 2 to 1), and give them all a consolidation value of b_{free} as demonstrated in Figure 4b. Tuning is then performed on a small fraction of training samples for each class at the same time. A smaller b_{free} value at this stage is expected to allow refinement to occur more easily.

Note that even though this backward transfer looks similar to batch learning of all tasks, it is a final fine-tuning process which does not usually incur a significant computational cost, as the network has been "pre-trained" on each task individually.

2.6 Selective Forgetting

If we have reached a size limit for the network, but still want to learn additional tasks, we can perform selective forgetting to free up network capacity from previous tasks.

To perform selective forgetting, we determine which tasks are the least important. This can be done using a heuristic method such as keeping track of usage frequency. Forgetting unused tasks would be similar to how we may forget a face or name if the associated person has not been seen for a long time.

In our running example, let us assume the least important task is task 1. Before we start training on a new task (task 4), we again apply b_{nf} to weights used by previous classes, however, for those nodes used primarily by task 1, the weights between them are unfrozen with a consolidation of b_{free}. Doing so sacrifices performance on task 1 and possibly a small amount of performance on tasks obtaining forward transference from task 1 in exchange for the ability to continue learning new tasks. The consolidation settings of weights when learning some further task 4 can be seen in Figure 4c.

Note that when the network capacity is reached but we wish to learn additional tasks, network capacity for previous tasks must be sacrificed for new tasks. With b values set to be small for task 1, task 1 is naturally and gradually forgotten when the new tasks are learned. There is not necessarily a clear point at which task 1 is forgotten. This is also similar to human memory when one can only recall the face of an old friend vaguely.

2.7 Consideration of Computational Requirements

Assume that there are a total of T tasks. Training T tasks individually and incrementally (as has been described earlier in this section) needs $\mathcal{O}(T)$ time complexity in total. Additional computation is needed for confusion reduction and backward transfer. However, if we assume tasks are highly distinctive, then confusion would happen infrequently. The final backward transfer can resolve any leftover errors in all tasks. As the network has been pre-trained by all tasks, hopefully the final backward transfer would not require too much computation. On the other hand, with proper forward transfer and backward transfer of knowledge, the overall training data needed would be smaller, thus requiring less learning time. With regard to the suite of LLL metrics proposed by [9] and displayed

in Figure 2, we expect that by intelligently encouraging forward and backward transfer and reducing rehearsal and sample storage to minimally necessary levels, we can outperform related work in a majority of metrics. Detailed empirical and theoretical analysis will however be left to our future manuscript [16].

3 Speculations on Parallels with Human Learning

behaviour	**# new nodes/class**	b_{tr}	**rehearsal**	b_{nf}
Resourceful and versatile	large	small	yes	large
Memory loss	-	-	-	medium/small
Sleep deprived	-	-	no	-
"Rain man"	-	large	-	large
Alzheimer's disease	small	-	-	large

Table 1: Correspondences between settings that can be used in our approach and behavioural descriptions. For example, being able to transfer knowledge across tasks and from old tasks to a newly encountered one allows us to be versatile. This ability to transfer knowledge is analogous to a small b_{tr} value in our approach.

As our approach works at the level of controlling the flexibility of individual network weights, it is natural to ask how our approach lines up with the mammalian brain whose lifelong learning behaviours we are attempting to capture. In this section we consider the parallels between our model and human learning with respect to several behavioural patterns listed in Table 1. For each of these, we will discuss how a similar behaviour can be exhibited by our model with specific hyperparameter settings, and how it may translate to what happens in the human brain.

Resourceful and versatile. Ideally, human learning allows us to acquire a lot of knowledge, yet be flexible enough to adapt to new experiences and make meaningful connections between experiences. The ability to make connections between different events and skills is roughly analogous to forward and backward transference explicitly considered by our approach, which is modulated though the b_{tr} hyperparameter and rehearsal. Being able to remember important information for long spans of time corresponds to a large b_{nf} value, while also being able to adapt to new tasks is facilitated by adding new nodes to the network for each task.

Memory loss. In our approach, a large b_{nf} consolidation value is applied to weights when memories need to remain unchanged. Thus, when the value for b_{nf} is not chosen properly, the model will lack the ability to remember tasks. A small or medium b_{nf} value, where memories are easily replaced with new ones, may appear similar to gradual memory loss in humans, with smaller b_{nf} values corresponding to faster loss.

Sleep deprived. The human brain is suspected of performing important memory-related processes during sleep, and sleep deprivation has been observed to be detrimental to cognitive performance

related to memory [19, 20]. Confusion reduction and backward transfer are important stages of our proposed approach which utilize rehearsal (functionally similar to memory replay) – where the model is exposed to samples from past tasks in order to perform fine-tuning to achieve various properties. Without these rehearsal-utilizing steps, the model may be less able to distinguish between samples of similar classes. Additionally, the ability to identify connections between newer tasks and older ones will be lost, so that potentially useful newly acquired skills cannot benefit older tasks.

"Rain man". By setting b_{tr} to be a large value and not performing the forward and backward transfer step, skill transfer between tasks will not occur. If b_{nf} is still large, this leads to a model which is very good at rote learning – remembering individual tasks, but unable to generalize knowledge to new or old tasks. This is reminiscent of Kim Peek [21], who was able to remember vast amounts of information, but performed poorly at abstraction-related tasks.

Alzheimer's disease. Usually an early stage of Alzheimer's disease is characterized by good memory on events years ago but poor on recent events [22]. This can be modeled in our framework by a small number of new neurons for new tasks and a very large b_{nf} for old ones.

4 Related Work

Much recent work on LLL has been done, with a thorough review of this work available in [23]. However, it seems that no existing approaches simultaneously address all of the LLL abilities of our proposed approach. LLL approaches can be grouped by their primary mechanism. Namely, we consider the approach groups most relevant to the present work: parameter-based, rehearsal-based, and dynamic network-based.

Parameter-based approaches. Parameter-based LLL approaches aim to identify the important weights of a neural network and prevent their modification in order to avoid catastrophic forgetting. An early influential approach of this type is Elastic Weight Consolidation [7], which uses Fisher information to estimate weight importance and employs a quadratic loss to penalize weight changes during training. Aiming to bring some of the biological complexity of real synapses to neural networks, [11] proposes an approach which allows each weight to estimate its own importance during the training process. Theoretical improvements and generalization of [7, 11] can be found in [12, 13]. Uniquely, [24] presents an unsupervised approach to calculate weight importance, which works by estimating the sensitivity of the learned function with respect to each weight (as opposed to the sensitivity with regard to the loss function, as in [7, 11]). Orthogonal Weights Modification [25] can be viewed as a type of parameter-based approach. While the technique does not entirely prevent the modification of important weights, it aims to restrict their movement to directions where forgetting will not occur. Similar to existing parameter-based approaches, the present approach uses regularization to prevent the modification of important weights, and similar to [12], we design our approach to not only combat catastrophic forgetting, but also reduce intransigence and confusion. However, unlike existing approaches, we leverage regularization for weight consolidation in a more flexible manner by modulating the consolidation strengths of subsets of weights as necessary for

maintaining some task skills while allowing other tasks to benefit from transference and refinement. Similar to [24], we also incorporate the ability to selectively forget tasks, but implement it in a more controlled fashion so that forgetting of memories can be done according to any desired policy (not just slowly fading as task skills go unused).

Rehearsal-based approaches. Rehearsal is one of the earliest strategies employed to combat catastrophic forgetting [26]. These approaches generally work by storing some amount of past task data to be incorporated into further training, so that information about old task data distributions is never fully lost (as can happen in parameter-based approaches). When learning new tasks, many rehearsal-utilizing approaches require samples from all previous tasks. For example, Gradient Episodic Memory (GEM) [27] and its more efficient successor Averaged GEM [28] require samples from all previous tasks in order to to restrict the new-task loss so that the loss on old tasks does not increase. Meta-Experience Replay (MER) [8] similarly requires old task samples to perform gradient alignment and to mix in with new samples during training. Old task samples are also required by iCaRL [29] to compute a distillation loss [30] during new task training. In contrast, our unified approach does not require old task data while learning new tasks, only using this data sparingly to reduce pairwise class confusions after the new task learning, and when performing backward transfer. Similar to our approach, [31] does not necessarily require old samples when learning a new task. This is done through the observation that the last fully-connected layer of a lifelong learning neural network has a bias towards newer classes. The authors correct this bias with a two-parameter linear model per task, which is tuned using both previous and new task samples.

Dynamic network-based approaches. While consolidation-based and rehearsal-based LLL approaches aim to condense information about several tasks into the same set of weights, dynamic network-based approaches generally take the route of extending the network for new tasks. Progressive Neural Networks (PNNs) are a prime example of this type of approach [10]. Similar to our approach, PNNs start out with a single column (one set of neural network layers) to use for learning the first task, and for each subsequent task, a new column is added while all previous column weights are frozen during training (making PNNs immune to forgetting, but not necessarily confusion). Lateral connections from every past column to the newest column are also added to facilitate forward transfer. However, unlike our approach, we add lateral connections from new columns to older ones to facilitate backward transfer and propose a method to solve the confusion that comes with single-head evaluation (while [10] considers only multi-head evaluation). A primary criticism of PNNs is the quickly growing network size that comes with added a fixed, possibly redundant set of nodes for each new task. By supporting selective forgetting and variable-size additional columns, our proposed approach partially deals with this increasing network size by allowing parameters of less important tasks to effectively be recycled, and by adding less capacity for easier tasks. Improving upon PNNs, several approaches with more efficient resource allocation have been proposed [32–35]. Dynamically Expandable Networks (DENs) [32] extend each layer of the network only as much as necessary, and abstractly works by identifying which existing parameters can be used as-is, and when a parameter would be substantially changed to adapt to the new task, a duplicate of the

associated neuron is retrained so that the network has a version of the neurons tuned for previous tasks, and a version tuned for the new task. After training on each task, fine-tuning on all tasks is performed, making DEN also fall under the rehearsal-utilizing group of approaches. Naturally, DEN is effective at facilitating forward transfer, however, it does not consider backward transfer and modifying the approach for selective forgetting may pose a challenge. Reinforced Continual Learning (RCL) [35] aims to optimize the expansion amount of each layer to accommodate each new task using reinforcement learning, and aims to balance accuracy on new task with added network complexity. Unlike DEN, RCL keeps the learned parameters for previous tasks fixed and only updates the added parameters. Similar to both DEN and PNNs, RCL does not account for either backward transfer, confusion reduction, or selective forgetting.

5 Conclusions

In this work, we presented a unified approach for lifelong learning. This approach tackles a difficult problem that captures many important aspects of human learning, namely non-forgetting, forward transfer, confusion reduction, backward transfer, few-shot learning, and selective forgetting. Progress in this area is critical for the development of computationally efficient and flexible machine learning algorithms. The success at this problem reduces the demand for training data while a single model learns to solve more and more tasks. While previous works have focused on a subset of these lifelong learning skills, our proposed approach utilizes a single mechanism, controlling weight consolidation, to address all of the considered skills. We define only a small number of consolidation hyperparameters which are dynamically applied to groups of weights. In addition to describing the novel approach, we examine its parallels with human learning. We note several similarities in the response of our model to hyperparameter settings and the effects on human learning to analogous changes in the brain.

References

[1] Spence C. & Stein B Calvert, G. The handbook of multisensory processes. *Cambridge,MA: The MIT Press*, 2004.

[2] Lewkowicz D. & Spence C. Bremner, A. Multisensory development. *Oxford University Press*, 2012.

[3] S. Barnett and S. Ceci. When and where do we apply what we learn? a taxonomy for fartransfer. *Psychological Bulletin128*, page 612–637, 2002.

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

[5] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. Emnist: an extension of mnist to handwritten letters. *CoRR*, abs/1702.05373, 2017.

[6] 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.

[7] 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. cite arxiv:1612.00796.

[8] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.

[9] Natalia Díaz-Rodríguez, Vincenzo Lomonaco, David Filliat, and Davide Maltoni. Don't forget, there is more than forgetting: new metrics for continual learning. *arXiv preprint arXiv:1810.13166*, 2018.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] Robert J Wang, Xiang Li, and Charles X Ling. Pelee: A real-time object detection system on mobile devices. In *Advances in Neural Information Processing Systems*, pages 1963–1972, 2018.

[15] Chuanming Wang, Huiyuan Fu, Charles X Ling, Peilun Du, and Huadong Ma. Region-based global reasoning networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, page TBD, 2020.

[16] Authors not yet determined. Unified lifelong learning: Heuristics and empirical results. 2020.

[17] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[18] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[19] Matthew P Walker. Sleep, memory and emotion. In *Progress in brain research*, volume 185, pages 49–68. Elsevier, 2010.

[20] William DS Killgore. Effects of sleep deprivation on cognition. In *Progress in brain research*, volume 185, pages 105–129. Elsevier, 2010.

[21] Darold A Treffert and Daniel D Christensen. Inside the mind of a savant. *Scientific American*, 293(6):108–113, 2005.

[22] MC Tierney, JP Szalai, WG Snow, RH Fisher, A Nores, G Nadon, E Dunn, and PH St George-Hyslop. Prediction of probable alzheimer's disease in memory-impaired patients: A prospective longitudinal study. *Neurology*, 46(3):661–665, 1996.

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

[24] 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.

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

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

[27] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *NIPS*, pages 6467–6476, 2017.

[28] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018.

[29] 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.

[30] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

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

[32] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *ICLR (Poster)*. OpenReview.net, 2018.

[33] Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11321–11329, 2019.

[34] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. *arXiv preprint arXiv:1904.00310*, 2019.

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