Continual Learning, Fast and Slow

Quang Pham, Chenghao Liu, Steven C. H. Hoi, *Fellow, IEEE*

Abstract—According to the Complementary Learning Systems (CLS) theory [1] in neuroscience, humans do effective *continual learning* through two complementary systems: a fast learning system centered on the hippocampus for rapid learning of the specifics, individual experiences; and a slow learning system located in the neocortex for the gradual acquisition of structured knowledge about the environment. Motivated by this theory, we propose *DualNets* (for Dual Networks), a general continual learning framework comprising a fast learning system for supervised learning of pattern-separated representation from specific tasks and a slow learning system for representation learning of task-agnostic general representation via Self-Supervised Learning (SSL). DualNets can seamlessly incorporate both representation types into a holistic framework to facilitate better continual learning in deep neural networks. Via extensive experiments, we demonstrate the promising results of DualNets on a wide range of continual learning protocols, ranging from the standard offline, task-aware setting to the challenging online, task-free scenario. Notably, on the CTrL [2] benchmark that has unrelated tasks with vastly different visual images, DualNets can achieve competitive performance with existing state-of-the-art dynamic architecture strategies [3]. Furthermore, we conduct comprehensive ablation studies to validate DualNets efficacy, robustness, and scalability. Code will be made available at https://github.com/phquang/DualNet.

Index Terms—Continual learning, fast and slow learning.

1 Introduction

Humans have the remarkable ability to learn and accumulate knowledge over their lifetime to perform different cognitive tasks. Interestingly, such a capability is attributed to the complex interactions among different interconnected brain regions [4]. One prominent model is the *Complementary Learning Systems (CLS) theory* [1], [5] which suggests that much of the learning and remembering capabilities are attributed to the interactions between the systems located at the hippocampus and neocrotex regions.[1] Particularly, the hippocampus focuses on fast learning of pattern-separated representation of specific experiences [7], [8]. Via the memory consolidation process, the hippocampus's memories are consolidated to the neocortex over time to form a more general representation that supports long-term retention and generalization to new experiences. The two fast and slow learning systems constantly interact to facilitate fast learning and long-term remembering, which is illustrated in Fig. 1. On the other hand, although deep neural networks have achieved impressive results in many applications [9], they often require having access to a large amount of data while performing poorly when learning on data streams [10], [11]. Moreover, when the stream consists of data from several distributions, deep neural networks *catastrophically forget* past knowledge, which further hinders the overall performance [12], [13], [14]. Therefore, the main focus of this study is exploring how the CLS theory can motivate a general continual learning framework with a better trade-off between alleviating catastrophic forgetting and facilitating knowledge transfer.

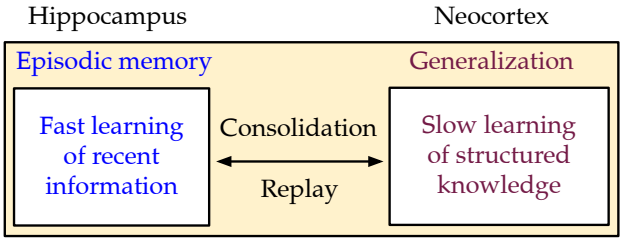

Fig. 1: An illustration of the fast and slow learning according to the CLS theory. Figure is inspired from [15].

In literature, several continual learning strategies are inspired from the CLS theory, from using the episodic memory [14] to improving the representation [16], [17]. However, most of them use supervised learning to model both the hippocampus and neocortex functionalities, which lack a separate slow, general representation learning component. During continual learning, the representation obtained by repeatedly performing supervised learning on a small amount of memory data can be prone to overfitting and may not generalize well across tasks. On the other hand. recent studies in continual learning show that unsupervised representation [18], [19] is often more resisting to forgetting compared to the supervised representation, which yields little improvements [20]. This result motivates us to conceptualize a novel fast-and-slow learning framework for continual learning, which comprises a two separate learning systems. The fast learner focuses on supervised learning while the slow learner focuses on accumulating better representations. As a result, the fast learner can take advantage of the slow representation to learn new tasks more efficiently, while retaining the old tasks' knowledge.

- *Corresponding author: Quang Pham is with Institute for Infocomm Research (I²R), A*Star.*
  E-mail: hqpham.2017@phdcs.smu.edu.sg
- *Chenghao Liu is with Salesforce Research Asia.*
  E-mail: chenghao.liu@salesforce.com
- *Steven C. H. Hoi is with Singapore Management University and Salesforce Research Asia.*
  E-mail: chhoi@smu.edu.sg

1. The brain functionalities involve many components [6] and understanding its mechanism is still an open research topic. For simplicity and brevity, we use hippocampus and neocortex to refer to the "hippocampus-dependent" and "neocortex-dependent" systems.

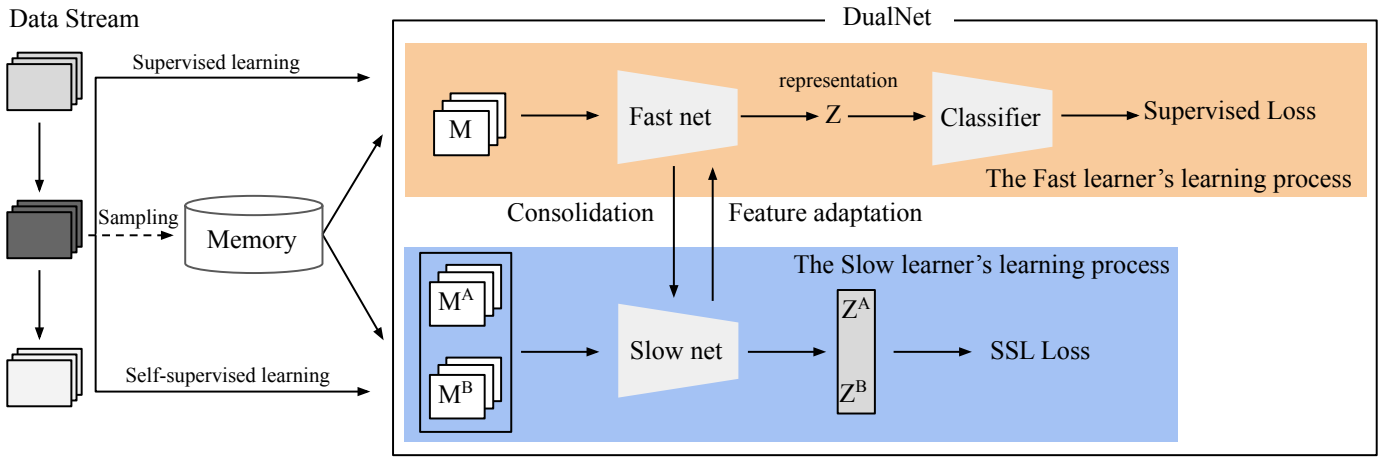

Fig. 2: Overview of the DualNet architecture, which consists of (i) a slow learner (blue) that learns representation by optimizing an SSL loss using samples from the memory, and (ii) a fast learner (orange) that adapts the slow net's representation for quick knowledge acquisition of labeled data. Both learners can be trained synchronously. $\boldsymbol{M}$ denotes a randomly sampled mini-batch and $\boldsymbol{M}^A$, $\boldsymbol{M}^B$ denote two views of $\boldsymbol{M}$ obtained by applying two different data transformations. Best viewed in colors.

From the fast-and-slow learning framework, we propose *DualNets* (for Dual Networks), a novel and practical continual learning paradigm consisting of two separate learning systems. Particularly, DualNets consist of two *complementary* and *parallel* training processes. First, the representation learning phase involves only the *slow network*, which continuously optimizes a Self-Supervised Learning (SSL) loss to model the generic, task-agnostic features [18], [19]. Separating the slow learning phase allows DualNets continuously improve representation even when there are no labeled samples, which is ubiquitous in real-world deployment scenarios where labeled data are delayed [21] or even limited, which we will demonstrate in Sec. 4.1.6. Secondly and simultaneously, the supervised learning phase involves both learners. In this phase, the goal is to train the fast learner, a more lightweight model that can do supervised learning more efficiently on data streams. We also propose a simple feature adaptation mechanism so that the fast learner can incorporate the slow representations into its predictions to ensure good results. Fig. 2 depicts an overview of our DualNets.

Lastly, by design, the original DualNet [22] utilizes all slow features to learn the current sample. We note that this strategy may hinder the performance when the continuum (data stream) contains unrelated tasks. In such scenarios, there might be negative transfer among tasks, resulting in a performance drop when using all slow features. In continual learning literature, this challenge is commonly addressed by a dynamic architecture design, which uses different subnetworks for the unrelated tasks [2] and can perform well where tasks are not related. However, despite strong results, such strategies are often expensive to train, incurs additional complexities overhead, and often perform poorly in the online learning scenario [23]. Since DualNet's slow learning is generic, we propose to enrich DualNet with a more robust fast feature to tackle the complex transfer continual learning scenarios. To this end, we propose DualNet++, which equips DualNet with a simple yet elegant regularization strategy to alleviate the negative knowledge transfer in continual learning. Particularly, DualNet++ inserts a dropout layer between the fast and slow learners' interaction, which prevents its fast learner from co-adapting the slow features. As a result, DualNet++ is robust to the negative knowledge transfer under the presence of unrelated or interference tasks **without the need for modularized representations**. As we will empirically verify in Sec. 4.2, DualNet++ achieves promising performance on the CTrL benchmark [2], specifically designed to test the model's ability to transfer knowledge in different complex scenarios.

In summary, our work makes the following contributions:

1) We propose DualNet, a novel and generalized continual learning framework comprising two key components of fast and slow learning systems, which is motivated by the CLS theory.
2) We develop to practical algorithms of DualNet and DualNet++, which implements the fast and slow learning approaches for continual learning. Notably, DualNet++ is also robust to the negative knowledge transfer.
3) We conduct extensive experiments to demonstrate DualNet's competitive performance compared to state-of-the-art (SOTA) methods. We also provide comprehensive studies of DualNet's efficacy, robustness to the slow learner's objectives, and scalability to the computational resources.

2 Related Work

2.1 Continual learning

Existing continual learning methods [15], [24] can be broadly categorized into *two* groups. First, *dynamic architecture* methods aims at having a separate subnetwork for each task, thus eliminating catastrophic forgetting to a great extend. The task-specific network can be identified simply allocating new parameters [25], [26], [27], finding a configuration of existing blocks or activations in the backbone [28], [29],

or generating the whole network conditioning on the task identifier [30]. While achieving strong performance, they are often expensive to train and do not work well on the online continual learning setting [23] because of the lack of knowledge transfer mechanism across tasks. In the second category of *fixed architecture* methods, learning is regularized by employing a memory to store information of previous tasks. In *regularization-based* methods, the memory stores the previous parameters and their importance estimations [13], [31], [32], [33], which regulates training of newer tasks to avoid changing crucial parameters of older tasks. Recent works have demonstrated that the *experience replay* (ER) principle [34] is an effective approach and its variants [14], [23], [35], [36], [37], [38] have achieved promising results in different domains, from vision [39], language [40], [41], to reinforcement learning [42]. Recent approaches augment the ER strategies with a calibration component also showed encouraging results. Particularly, they introduce a separate controller to calibrate the backbone network to balance between alleviating forgetting and facilitating knowledge transfer [43], [44]. It is worth noting that such strategies requires the task identifiers and are only applicable to the task-aware scenarios.

Neuroscience inspired continual learning Neuroscience and the CLS theory have inspired several recent continual learning strategies [43], [45], [46], [47]. Such studies try to emulate the hippocampus and neocortex by introducing additional learning components [43], [48] or memory units [46], [47]. The most related work to ours is CLSER [48], which approaches fast-and-slow learning as updating the learners at different frequencies, i.e. the fast/slow learner is optimized more/less frequently. Nevertheless, CLSER still only learns the supervised learning patterns. Instead of learning at different frequencies, our work approaches the CLS theory from the representation learning perspective by introducing a slow learning phase to the slow net, which was not explored in previous studies. We will empirically compare DualNets with CLSER in Sec. 4.2.

2.2 Representation Learning for Continual Learning

Representation learning has been an important research field in machine learning and deep learning [49], [50]. Recent works demonstrated that a general representation could transfer well to many downstream tasks [51], or generalize well under limited training samples [52]. For continual learning, extensive efforts have been devoted to learning a generic representation that can alleviate forgetting while facilitating knowledge transfer. The representation can be learned either by supervised learning [53], unsupervised learning [17], [18], [19], or meta (pre-)training [16], [54]. While unsupervised and meta training have shown promising results on simple datasets such as MNIST and Omniglot, they lack the scalability to real-world benchmarks. Concurrent with our work, several studies have explored incorporating SSL into continual learning and achieved promising results [55], [56], [57]. Different from such studies which use one network for both representations, our DualNets decouple the slow learning into the slow learner, which does not require any pre-training steps and is capable of training synchronously. We will explore the benefits of having two learners in Sec. 4.1.5.

2.3 Feature Adaptation

Feature adaptation allows the feature to quickly change and adapt [58]. Existing continual learning methods have explored the use of task identifiers [29], [30], [43] or the memory data [54] as a context to facilitate learning. While the task identifier is powerful since it provides additional information regarding the task of interest, it is limited only to the task-aware setting or require inferring the underlying task, which can be challenging in practice. On the other hand, sample-based context conditioning is useful in incorporating information of similar samples to the current query and has found success beyond continual learning [52], [59], [60], [61]. Several continual learning strategies have adopted this approach by implementing the meta-learning gradient updating rules [54], [62], [63] and have shown promising results. However, we argue that naively adopting this approach is not practical for real-world continual learning because the model always performs the full forward/backward during inference, which suffers from the high computational costs. Moreover, the predictions are not deterministic because of the dependency on the data chosen for finetuning during inference. For DualNets, feature adaptation plays an important role in the interaction between the fast and slow learners. We address the limitations of existing techniques by developing a novel mechanism that allows the fast learner to efficiently utilize the slow representation without additional information about the task identifiers. Notably, our approach is built upon the feature-wise transformation [58], which does not require backpropagation during testing.

3 Method

3.1 Setting and Notations

We consider the continual learning setting [14], [23] over a continuum $\mathcal{D} = \{\boldsymbol{x}_i, t_i, y_i\}_i$, where each instance is a labeled sample $\{\boldsymbol{x}_i, y_i\}$ with an *optional* task identifier t_i. Each labeled sample is drawn from an underlying distribution $P^t(\boldsymbol{X}, \boldsymbol{Y})$ that represents a task and can suddenly change to P^{t+1}, indicating a task switch. In the *task-aware setting*, the task identifier t is provided and only the corresponding task's classifier is selected for inference [14]. When the task identifier is not provided (during both training and evaluation), the model has a shared classifier for all classes observed so far, which follows the *task-free setting* [64], [65]. We consider both scenarios in our experiments. Note that there is a hybrid setting with task identifiers provided during training but not during evaluation [3], which we will consider as *task-aware* in this work.

A common continual learning component is the episodic memory $\mathcal{M}$ to store a subset of observed data and interleave them when learning the current samples [14], [39]. From $\mathcal{M}$, we use $\boldsymbol{M}$ to denote a randomly sampled mini-batch, and $\boldsymbol{M}^A$, $\boldsymbol{M}^B$ to denote two views of $\boldsymbol{M}$ obtained by applying two different data transformations. We also denote $\boldsymbol{\phi}$ as the parameter of the slow network that learns general representation from the input data and $\boldsymbol{\theta}$ as the parameter of the fast network that learns the transformation coefficients.

3.2 The DualNets Paradigm

DualNet learns generic representations to support better generalization capabilities across both old and new tasks in

continual learning. The model consists two main learning modules (Figure 2): (i) the slow learner is responsible for learning a general representation; and (ii) the fast learner learns with labeled data from the continuum to quickly capture the new information and then consolidate the knowledge to the slow learner.

DualNets' learning can be broken down into two *synchronous* phases. First, the self-supervised learning phase in which the slow learner optimizes an SSL objective on incoming samples and samples from the episodic memory. Second, the supervised learning phase, which involves the fast learner using the representation from the slow learner and adapting it for supervised learning. The incurred loss will be backpropagated through both learners for supervised knowledge consolidation. Additionally, the fast learner's adaptation is per-sample-based and does not require additional information such as the task identifiers. While the SSL can make the slow learner's representation generic, backprogating the supervised learning loss end-to-end ensures that the slow learner can learn representations that are useful for the supervised learning. Lastly, DualNet uses the same episodic memory's budget as other methods to store the samples and their labels, but the slow learner only requires the samples while the fast learner uses both samples and their labels.

3.2.1 The Slow Learner

The slow learner is a standard backbone network ϕ trained to optimize an SSL loss, denoted by $\mathcal{L}_{SSL}$. As a result, any SSL objectives can be applied in this step. However, to minimize the additional computational resources while ensuring a general representation, we only consider the SSL loss that (i) does not require additional memory unit (such as the negative queue in MoCo [67]), (ii) does not always maintain an additional copy of the network (such as BYOL [68]), and (iii) does not use handcrafted pretext losses (such as RotNet [49] or JiGEN [69]). Therefore, we consider contrastive SSL losses [51] and implement DualNets using Barlow Twins [70], a common SSL method that achieved promising results with minimal computational overheads. Formally, Barlow Twins requires two views $\boldsymbol{M}^A$ and $\boldsymbol{M}^B$ by applying two different data transformations to a batch of images $\boldsymbol{M}$. By default, $\boldsymbol{M}$ contains incoming samples from the environment and samples in the episodic memory to maximize the number of samples for SSL. The augmented data are then passed to the slow net ϕ to obtain two representations $\boldsymbol{Z}^A$ and $\boldsymbol{Z}^B$. The Barlow Twins loss is defined as:

$$\mathcal{L}_{\mathcal{BT}} \triangleq \sum_i (1 - \mathcal{C}_{ii})^2 + \lambda_{\mathcal{BT}} \sum_i \sum_{j \neq i} \mathcal{C}_{ij}^2, \tag{1}$$

where $\lambda_{\mathcal{BT}}$ is a trade-off factor, and $\mathcal{C}$ is the cross-correlation matrix between $\boldsymbol{Z}^A$ and $\boldsymbol{Z}^B$:

$$\mathcal{C}_{ij} \triangleq \frac{\sum_b z^A_{b,i} z^B_{b,j}}{\sqrt{\sum_B (z^A_{b,i})^2} \sqrt{\sum_B (z^B_{b,j})^2}} \tag{2}$$

with b denotes the mini-batch index and i, j are the vector dimension indices. Intuitively, by optimizing the cross-correlation matrix to be identity, Barlow Twins enforces the network to learn essential information that is invariant to the distortions (unit elements on the diagonal) while eliminating the redundancy information in the data (zero element elsewhere). In our implementation, we follow the standard practice in SSL to employ a projector on top of the slow network's last layer to obtain the representations $\boldsymbol{Z}^A, \boldsymbol{Z}^B$. For supervised learning with the fast network, which will be described in Sec. 3.2.2, we use the slow network's last layer as the representation $\boldsymbol{Z}$.

Optimization in Online Continual Learning In most SSL training, the LARS optimizer [71] is employed for distributed training across many devices, which takes advantage of a large amount of unlabeled data. However, in **online continual learning**, the episodic memory only stores a small number of samples, which are always changing because of the memory updating mechanism. As a result, the data distribution in the episodic memory always drifts after each iteration, and the SSL loss in DualNet presents different challenges compared to the traditional SSL optimization. Particularly, although the SSL objective in continual learning can be easily optimized using one device, we need to quickly capture the knowledge of the currently stored samples before the newer ones replace them. In this work, we propose to optimize the slow learner using the *Look-ahead* optimizer [72], which performs the following updates:

$$\tilde{\phi}_k \leftarrow \tilde{\phi}_{k-1} - \epsilon \nabla_{\tilde{\phi}_{k-1}} \mathcal{L}_{\mathcal{BT}}, \text{ with } \tilde{\phi}_0 \leftarrow \phi \text{ and } k = 1, \ldots, K \tag{3}$$

$$\phi \leftarrow \phi + \beta(\tilde{\phi}_K - \phi), \tag{4}$$

where β is the Look-ahead's learning rate and ϵ is the Look-ahead's SGD learning rate. As a special case of $K = 1$, the optimization reduces to the traditional optimization of $\mathcal{L}_{\mathcal{BT}}$ using SGD. By performing $K > 1$ updates using a standard SGD optimizer, the look-ahead weight $\tilde{\phi}_K$ is used to perform a momentum update for the original slow learner ϕ. As a result, the slow learner optimization can explore regions that are undiscovered by the traditional optimizer and enjoys faster training convergence [72]. Note that SSL focuses on minimizing the training loss rather than generalizing this loss to unseen samples, and the learned representation requires to be adapted to perform well on a downstream task. Therefore, such properties make the Look-ahead optimizer a more suitable choice over the standard SGD to train the slow learner. For the batch continual learning setting [13], because the model can learn the current task for many epochs, it is sufficient to train the slow learner with the standard SGD.

Lastly, we emphasize that although we choose to use Barlow Twins as the SSL objective, DualNets are compatible with any existing methods in the literature, which we will explore empirically in Sec. 4.1.3. Moreover, we can always train the slow learner in the background by optimizing Equation 1 synchronously with the continual learning of the fast learner, which we will detail in the following section.

3.2.2 The Fast Learner

Given a labeled sample $\{\boldsymbol{x}, y\}$, the fast learner's goal is utilizing the slow learner's representation to learn this sample via an adaptation mechanism. We propose a general context-free adaptation mechanism by extending and improving the channel-wise transformation [43], [58] to the general continual learning setting. Particularly, such strategies relies

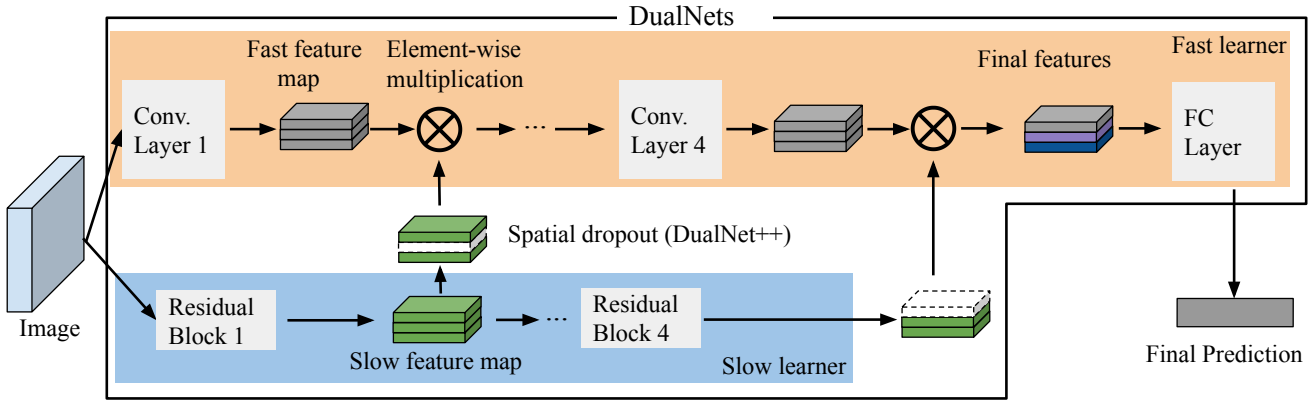

Fig. 3: An illustration of DualNets forward calculation during the supervised learning or inference phase on a standard ResNet [66] backbone. Given an input image, the slow learner (blue) first performs the forward pass to obtain the feature maps. Then, the fast learner (orange) perform its forward pass using both the slow and fast features. Best viewed in colors.

on low dimensional context vectors, such as task identifiers, to learn a channel-wise transformation coefficients. In the task-free setting, the model needs to learn such information from the raw, high dimensional images. To compensate for the increased learning complexity, we propose to implement the fast net as a smaller neural network instead of a simple linear layer [43]. Moreover, the transformation is pixel-wise instead of channel-wise to allow for a more fine-grained usage of the slow feature given the current image.

Formally, let $\{\boldsymbol{h}_i\}_{i=1}^{L}$ be the feature maps from the slow learner's layers on the image $\boldsymbol{x}$, e.g. $\boldsymbol{h}_1, \boldsymbol{h}_2, \boldsymbol{h}_3, \boldsymbol{h}_4$ are outputs from four residual blocks in ResNets [66], our goal is to obtain the adapted feature h'_L conditioned on the image $\boldsymbol{x}$. Therefore, we design the fast learner as a simple CNN with L layers, and the adapted feature $\boldsymbol{h}'_L$ is obtained as

$$\begin{aligned} \boldsymbol{m}_l &= g_{\boldsymbol{\theta},l}(\boldsymbol{h}'_{l-1}), \text{ with } \boldsymbol{h}'_0 = \boldsymbol{x} \text{ and } l = 1, \dots, L \\ \boldsymbol{h}'_l &= \boldsymbol{h}_l \odot \boldsymbol{m}_l, \quad \forall l = 1, \dots, L, \end{aligned} \tag{5}$$

where $\odot$ denotes the element-wise multiplication, $g_{\boldsymbol{\theta},l}$ denotes the l-th layer's output from the fast network $\boldsymbol{\theta}$ and has the same dimension as the corresponding slow feature $\boldsymbol{h}_l$. The fast net final layer's transformed feature $\boldsymbol{h}'_L$ will be fed into a classifier for prediction.

Thanks to the simplicity of the transformation, the fast learner is light-weight but still can take advantage of the slow learner's rich representation for better supervised learning. Meanwhile, the slow learner is mostly trained by the SSL loss to obtain a generic representation that is resistant to catastrophic forgetting. Figure 3 illustrates the fast and slow learners' interaction during the supervised learning or inference phase.

The Fast Learner's Objective To further facilitate the fast learner's knowledge acquisition during supervised learning, we also mix the current sample with previous data in the episodic memory, which is a form of experience replay (ER). Particularly, given the incoming labeled sample $\{\boldsymbol{x}, y\}$ and a mini-batch of memory data $\boldsymbol{M}$ belonging to a past task k, we consider the ER with a soft label loss [37] for the supervised learning phase as:

$$\begin{aligned} \mathcal{L}_{tr} =& \text{CE}(\pi(\text{DualNet}(\boldsymbol{x}), y) + \frac{1}{|\boldsymbol{M}|}\sum_{i=1}^{|\boldsymbol{M}|} \text{CE}(\pi(\hat{y}_i), y_i) + \\ &+ \lambda_{tr} D_{\text{KL}}\left(\pi\left(\frac{\hat{y}_i}{\tau}\right) \middle\| \pi\left(\frac{\hat{y}_k}{\tau}\right)\right), \end{aligned} \tag{6}$$

where CE is the cross-entropy loss , D_{KL} is the KL-divergence, $\hat{y}$ is the DualNet's prediction, $\hat{y}_k$ is snapshot of the model's logits (the fast learner's prediction) of the corresponding sample at the end of task k, $\pi(\cdot)$ is the softmax function with temperature τ, and λ_{tr} is the trade-off factor between the soft and hard labels in the training loss. Similar to [38], [43], Equation 6 requires minimal additional memory to store the soft label $\hat{y}$ in conjunction with the image $\boldsymbol{x}$ and the hard label y.

3.3 DualNet++

This section details DualNet++, an improved version of DualNet to tackle the challenges of complex transfer scenarios in continual learning [2]. In many real-world applications, data in continual learning may contain vastly different visual features, which could even be adversarial to one another [73]. Common approaches to tackle such challenges [2], [3] are mainly based on the dynamic architectures, which compartmentalize knowledge into different modules. Then, the model only uses the relevant subnetwork to learn a current task, which only transfers useful knowledge while alleviating the negative effects of unrelated features. In contrast, since DualNet already learned the slow, generic feature, we propose to improve the fast features to be more robust to tackle the complex transfer challenges without needing modularized representations.

To this end, we analyze the DualNet's drawback in achieving a good transfer when learning from complex continual learning streams. We argue that since the slow features in DualNets are obtained from all tasks' data, the original DualNet will learn the fast features dependent on the slow features. While this is helpful in the controlled environments with no negative transfers, it will hinder

the performance when there are tasks unrelated to one another. Thus, we propose DualNet++ that alleviates the co-adaptation between the fast and slow features, allowing it to learn more robust features that are useful for the current task. DualNet++ introduces a simple dropout layer [74] between the fast and slow learners' interactions. As a result, under the presence of negative transfer, the fast learner will not become dependent on the slow feature [9], [74] and can focus on learning features useful for the current inputs.

To implement DualNet++, we insert a spatial dropout layer [75] between the fast and slow learner interaction. Formally, DualNet++ replace the interaction in Eq. 5 as:

$$\begin{aligned} \boldsymbol{m}_l =& g_{\boldsymbol{\theta},l}(\boldsymbol{h}'_{l-1}), \text{ with } \boldsymbol{h}'_0 = \boldsymbol{x} \text{ and } l = 1, \dots, L \\ \boldsymbol{h}'_l =& \boldsymbol{D}_l \odot \boldsymbol{h}_l \odot \boldsymbol{m}_l, \quad \forall l = 1, \dots, L, \end{aligned} \tag{7}$$

where $\boldsymbol{D}_l \in \mathbb{R}^{n \times w \times h}$ is a spatial dropout mask obtained as

$$d_{l,i} \sim \text{Bernoulli}(p), \forall i = 1, \dots, n \tag{8}$$

$$\boldsymbol{D}_l \leftarrow \text{Repeat}(\boldsymbol{d}_l, n \times w \times h), \boldsymbol{d}_l = \{d_{l,1}, \dots, d_{l,n}\}. \tag{9}$$

In Eq. 8, Bernoulli(p) denotes a sample randomly drawn from a Bernoulli distribution with probability p, and Repeat in Eq. 9 denotes reshaping a vector to a particular dimensions by repeating its values along the required axes. Specifically, the dropout mask $\boldsymbol{D}$ is obtained by performing n independent dropout trial on a feature map of size $n \times w \times h$, and each trial will zero an entire channel. We also note that it is possible to apply the traditional dropout [74] on the pixels independently, or inserting dropout in the backbone networks. However, preliminary results of such strategies are not promising due to the incompatibility between dropout and batch normalization [76], [77]. Therefore, we decided to not explore these configurations further. In contrast, the spatial dropout is more suitable for convolutional neural networks, and is only inserted in the fast and slow networks' interaction, not between the hidden layers. Lastly, a recent work [78] also show promising results of applying dropout in continual learning. However, their studies only focus on the simple feed-forward architectures and left the convolution networks unexplored.

4 Experiments

We compare DualNets against competitive continual learning approaches in both the online [14] and offline [2], [13] scenarios. Our goal of the experiments is to investigate the following hypotheses: (i) DualNets can work well across different continual learning scenarios; (ii) DualNets are robust to the choice of the SSL loss; (iii) DualNets are scalable with the number of SSL training iterations; (iv) DualNet++ can efficiently learn under the presence of unrelated tasks and distribution shifts. In all experiments, DualNet++'s dropout ratio is set as $p = 0.1$ for the online setting, and $p = 0.2$ for the batch setting, unless otherwise stated.

4.1 Online Continual Learning Experiments

We first consider the *Online Continual Learning setting* [14] where both the tasks and samples within each task arrive sequentially. This setting presents a unique challenge where the catastrophic forgetting and facilitating knowledge transfer problems are entangled [38]. Thus, successful online continual learning solutions must achieve a good trade-off of these conflicting objectives.

4.1.1 Setup

Benchmarks We consider the "Split" continual learning benchmarks constructed from the miniImageNet [79], CORE50 dataset [80], and the ImageNet datasets [81] with 17, 10, and 10 continual learning tasks, respectively. Each task is created by randomly sampling without replacement five classes from the original dataset. We also consider both the task-aware and task-free protocols. We reserved three tasks from the miniImagenet dataset to cross-validate the hyper-parameters of all methods. The hyper-parameter configuration is then fixed during continual learning. We also found that these hyper-parameters could transfer well to the remaining benchmarks. For the task-aware (TA) protocol, the task identifier is available, and only the corresponding classifier is selected for training and evaluation. In contrast, the task-identifiers are not given in the task-free (TF) protocol, and the models have to predict all classes observed so far.

Evaluation Metrics We run the experiments five times and report the averaged accuracy of all tasks/classes at the end of training [14] (ACC(↑)), the forgetting measure [64] (FM(↓)) and backward transfer [14] (BWT(↑)), and the learning accuracy [35] (LA(↑)). Let $a_{i,j}$ be the model's accuracy evaluated on the testing data of task j after it is trained on task i. The above metrics are defined as:

- **Average Accuracy ACC(↑) (higher is better):**

$$\text{ACC}(\uparrow) = \frac{1}{T}\sum_{i=1}^{T} a_{T,i}. \tag{10}$$

- **Forgetting Measure FM(↓) (lower is better):**

$$\text{FM}(\downarrow) = \frac{1}{T-1}\sum_{j=1}^{T-1} \max_{l \in \{1,\dots T-1\}} a_{l,j} - a_{T,j}. \tag{11}$$

- **Backward Transfer BWT(↑) (higher is better):**

$$\text{BWT}(\downarrow) = \frac{1}{T-1}\sum_{j=1}^{T-1} a_{T,j} - a_{j,j}, \tag{12}$$

- **Learning Accuracy LA(↑) (higher is better):**

$$\text{LA}(\uparrow) = \frac{1}{T}\sum_{i=1}^{T} a_{i,i}. \tag{13}$$

The above metrics provide a comprehensive evaluation of online continual learning methods. Particularly, ACC(↑) reports the overall performance at the end of learning, and is usually used to compared among methods. FM(↓) and BWT(↑) report the average performance drop of each task at the end of learning and indicate the model's ability to address catastrophic forgetting. Lastly, LA(↑) measures the model's ability to learn new tasks indicating its ability to facilitate forward knowledge transfer. Notably, the LA(↑) metric we considered here is an unnormalized variant of the Intransigence Measure [64].

Baselines We compare DualNet and DualNet++ against a suite of state-of-the-art continual learning methods. First, we consider ER [39], a simple experience replay method that works consistently well across benchmarks. Then we

include DER++ [38], an ER variant that augments ER with a ℓ_2 loss on the soft labels. We also compare with CTN [43] a recent state-of-the-art method on the online task-aware setting. For all methods, the hyper-parameters are selected by performing grid-search on the cross-validation tasks.

Architecture We use a full ResNet18 [66] as the backbone for all methods. In addition, we construct the DualNet's fast learner as follows: the fast learner has the same number of convolutional layers as the number of residual blocks in the slow learners. A residual block and its corresponding fast learner's layer will have the same output dimensions. With this configuration, the fast learner's architecture is uniquely determined by the slow learner's network and only increased the number of parameters by about 20%. Lastly, all networks in this experiment are trained from scratch.

Training In the supervised learning phase, all methods are optimized by the (SGD) optimizer **over one epoch** with mini-batch size 10 on the Split miniImageNet and batch 32 for the CORE50 Split Imagenet benchmarks respectively. In the representation learning phase, we use the Look-ahead optimizer [72] to train the DualNets' slow learner as described in Sec. 3.2.1. We employ an episodic memory with *50 samples per task* and the Ring-buffer management strategy [14] in the task-aware setting. In the task-free setting, the memory is implemented as a reservoir buffer [82] with 100 samples per class. We simulate the synchronous training property in DualNet by training the slow learner with n iterations using the episodic memory data before observing a mini-batch of labeled data. Except for ImageNet, each experiments are repeated for five times.

Data pre-processing DualNet's slow learner follows the data transformations used in BarlowTwins [70]. For the supervised learning phase, we consider two options. First, the standard data pre-processing of no data augmentation during both training and evaluation, which is commonly implemented in existing studies [14], [39]. Second, we also train the baselines with data augmentation for a fair comparison. However, we observe the data transformation in [70] is too aggressive; therefore, we only implement the random cropping and flipping for the supervised training phase of these baselines. In all scenarios, the inference phase does not any use data augmentations.

4.1.2 Results of Online Continual Learning Benchmarks

We report the results of the online continual learning experiments in Tab. 1 and Tab. 2. Our DualNet's slow learner optimizes the Barlow Twins objective for $n = 3$ iterations between every incoming mini-batch of labeled data. We will report the impact of the SSL iteration in Sec. 4.1.4.

Generally, data augmentation creates more samples to train the models and provides improvements in all cases. Consistent with previous studies, we observe that DER++ performs slightly better than ER thanks to its soft-label loss. Similarly, CTN can perform better than both ER and DER++ because of its ability to model task-specific features. Overall, our DualNets consistently outperform other baselines by a large margin, even with the data augmentation propagated to their training. Specifically, DualNets are more resistant to catastrophic forgetting (lower FM) while greatly facilitating knowledge transfer (higher LA), which results in better overall performance, indicated by higher ACC. We also observe that DualNet++ performs marginally better than DualNet in all cases, suggesting the benefits of the spatial dropout regularization. Additionally, since our DualNets have a similar supervised procedure as DER++, this result shows that the DualNets' representation learning and fast adaptation mechanism are beneficial to continual learning. Lastly, we observe a similar trend on the large scale ImageNet benchmark where DualNets achieve better evaluation metrics compared to the vanilla strategies of ER and DER++.

4.1.3 Ablation Study of the Slow Learner Objectives and Optimizers

We now study the effects of the slow learner's objective and optimizer on the final performance of DualNets by considering several objectives to train the slow learner. First, we consider the *classification loss* to train the slow net, which reduces DualNet's representation learning to only supervised learning. Second, we consider various contrastive SSL losses, including SimCLR [83], SimSiam [84], and BYOL [68]. In this setting, DualNets' slow representation involves a direct optimization of a SSL loss and an indirect classification loss backpropagated via the fast learner.

We consider the Split miniImageNet-TA and TF benchmark with 50 memory slots per task and optimize each objective using the SGD and Look-ahead optimizers. Tab. 3 reports the result of this experiment. In general, we observe that SSL objectives achieve a better performance than the classification loss. Moreover, the Look-ahead optimizer consistently improves the performances on all objectives compared to the SGD optimizer. This result shows that the DualNets design is general and can work well with different slow learner's objectives. Interestingly, when using the Look-ahead optimizer, we observe a correlation between the SSL losses in DualNets with their performances in the standard SSL scenario [70]. This result suggests that DualNets can take advantage of future SOTA SSL losses to further improve the performance.

4.1.4 Ablation Study of Self-Supervised Learning Iterations

We now investigate DualNet's performances with different SSL optimization iterations n. Small values of n indicate there is little to no delay of labeled data from the continuum, and the fast learner has to query the slow learner's representation continuously. On the other hand, larger n simulates the situations where labeled data are delayed, which allows the slow learner to train its SSL objective for more iterations between each query from the fast learner. In this experiment, we gradually increase the SSL training iterations between each supervised update by varying from $n = 1$ to $n = 20$.

We run the experiments on both the Split miniImageNet benchmarks under the TA and TF settings. Fig. 4 reports the result of DualNet and DualNet++ in this scenario. In general, we observe that in all cases, the average accuracy ACC(↑) increases as more SSL iterations are allowed. The same conclusion also holds for the FM(↓) and LA(↑) metrics, although there are small fluctuations at $n = 3$ and $n = 10$. Moreover, DualNet++ is more stable than DualNet in this experiment by having smaller variance across different runs. We can conclude that both DualNet and DualNet++ are highly scalable with the number of SSL training iterations. This promising result demonstrates the DualNets' potential

TABLE 1: Evaluation metrics on the Split miniImageNet and CORE50 benchmarks. All methods use an episodic memory of 50 samples per task in the TA setting, and 100 samples per class in the TF setting. The "Aug" suffix denotes using data augmentation. We highlight the methods with best mean metrics in bold, and underline the second best methods

Method	Split miniImageNet-TA			Split miniImageNet-TF		
	ACC(↑)	FM(↓)	LA(↑)	ACC(↑)	FM(↓)	LA(↑)
ER [39]	58.24±0.78	9.22±0.78	65.36±0.71	25.12±0.99	28.56±1.10	49.04±1.56
ER-Aug	59.80±1.51	4.68±1.21	58.94±0.69	27.94±2.44	29.36±3.23	54.02±1.02
DER++ [38]	62.32±0.78	7.00±0.81	67.30±0.57	27.16±1.99	34.56±2.48	59.54±1.53
DER++-Aug	63.48±0.98	4.01±1.21	62.17±0.52	28.26±1.81	36.70±1.85	62.70±0.41
CTN [43]	65.82±0.59	**3.02±1.13**	67.43±1.37	N/A	N/A	N/A
CTN-Aug	68.04±1.23	3.94±0.98	69.84±0.78	N/A	N/A	N/A
DualNet [22]	73.20±0.68	3.86±1.01	74.12±0.12	36.86±1.36	28.63±2.26	63.46±1.97
DualNet++	**74.24±0.95**	**2.83±0.71**	**74.11±0.30**	**37.56±1.12**	**27.13±1.16**	**63.96±1.02**
Method	CORE50-TA			CORE50-TF		
	ACC(↑)	FM(↓)	LA(↑)	ACC(↑)	FM(↓)	LA(↑)
ER [39]	41.72±1.30	9.10±0.80	48.18±0.81	21.80±0.70	14.42±1.10	33.94±1.49
ER-Aug	44.16±2.05	5.72±0.02	47.83±1.61	25.34±0.74	15.28±0.63	37.94±0.91
DER [39]	46.62±0.46	4.66±0.46	48.32±0.69	22.84±0.84	13.10±0.40	34.50±0.81
DER++-Aug	45.12±0.68	5.02±0.98	47.67±0.08	28.10±0.80	10.43±2.10	36.16±0.19
CTN [43]	54.17±0.85	5.50±1.10	55.32±0.34	N/A	N/A	N/A
CTN-Aug	53.40±1.37	6.18±1.61	55.40±1.47	N/A	N/A	N/A
DualNet [22]	57.64±1.36	4.43±0.82	58.86±0.66	38.76±1.52	8.06±0.43	**40.00±1.67**
DualNet++	**59.07±1.30**	**2.86±0.92**	**59.23±1.03**	**39.42±1.80**	**7.08±2.25**	39.52±1.09

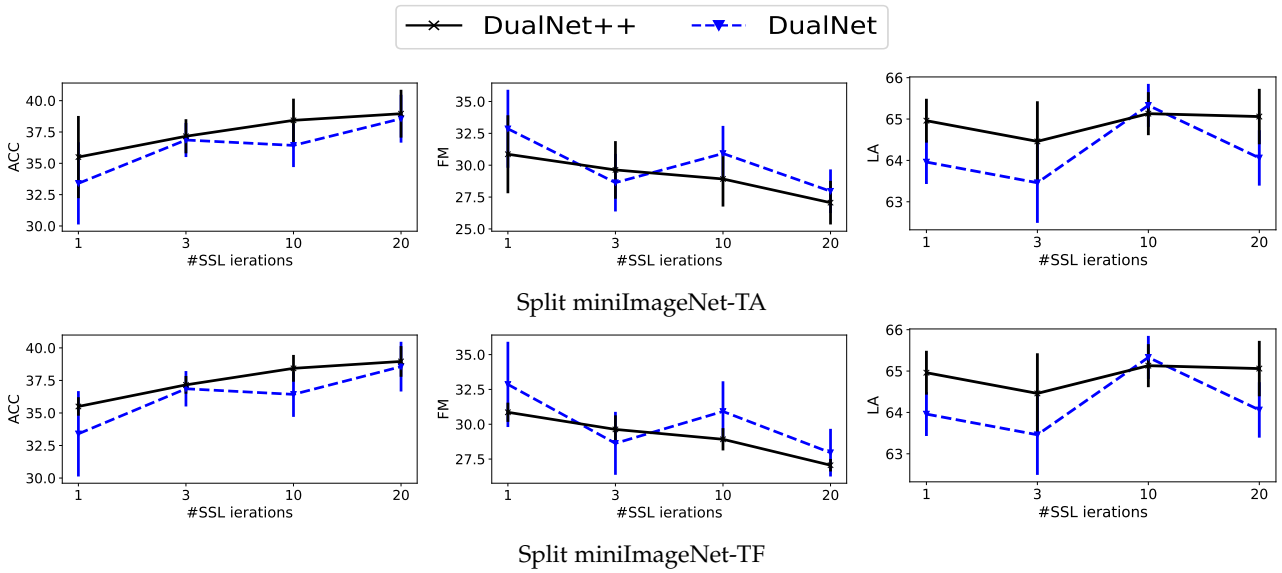

Fig. 4: Performance of DualNet and DualNet++ with different self-supervised learning iterations n.

TABLE 2: Evaluation metrics on the task-free Split ImageNet benchmark. Best result is in bold, second best is underlined

Method	ImageNet		
	ACC	FM	LA
ER	12.30	31.30	40.50
DER++	13.70	30.90	40.90
DualNet	14.87	30.07	**42.60**
DualNet++	**14.90**	**29.60**	41.70

to be deployed in real-world continual learning scenarios where labeled data is delayed [21], which allows the slow learner to learn in the background.

4.1.5 *Ablation Study of DualNets Components*

Compared to the standard ER strategy with soft labels [37], [38], DuelNets introduce an additional fast learner and a representation learning phase. In this experiment, we investigate the contribution of the fast learner on the Split miniImageNet benchmark, both the TA and TF settings. We create a variant, *Slow Learner*, that uses only a ResNet backbone to optimize both the supervised and SSL losses. Tab. 4 reports the result of this experiment. We can see that the slow learner variant binds both representation types into the same backbone and performs significantly worse than the original DualNet in both scenarios. It is worth noting that using only the slow learner to learn both representations is the strategy in concurrent works [56], [57]. This promising

TABLE 3: DualNet's performance under different slow learner objective and optimizers on the Split miniImageNet-TA benchmark

DualNet	SGD			Look-ahead		
	ACC(↑)	FM(↓)	LA(↑)	ACC(↑)	FM(↓)	LA(↑)
Barlow Twins [70]	70.99±1.01	**3.26±0.53**	70.93±0.99	**73.20±0.68**	**3.86±1.01**	**74.12±0.12**
SimCLR [83]	**71.49±1.01**	4.23±0.46	**72.64±1.20**	72.13±0.44	4.13±0.52	73.09±0.16
SimSiam [84]	70.55±0.98	4.93±1.31	71.90±0.65	71.94±0.64	4.21±0.28	72.93±0.38
BYOL [68]	69.76±2.12	4.23±1.41	70.33±0.87	71.73±0.47	**3.96±0.62**	72.06±0.28
Classification	68.50±1.67	5.53±1.67	**72.93±1.10**	70.96±1.08	6.33±0.28	73.92±1.14
DualNet++	SGD			Look-ahead		
	ACC(↑)	FM(↓)	LA(↑)	ACC(↑)	FM(↓)	LA(↑)
Barlow Twins [70]	71.10±1.03	**3.03±0.97**	70.96±1.12	**74.24±0.95**	**2.83±0.71**	**74.11±0.30**
SimCLR [83]	**71.16±0.72**	3.13±1.33	71.96±0.77	72.14±0.81	4.12±0.28	73.33±0.58
SimSiam [84]	69.56±0.11	4.53±0.54	71.66±1.01	71.99±1.33	4.13±0.45	73.01±0.57
BYOL [68]	69.16±1.12	4.24±1.21	69.93±1.46	71.63±1.12	4.96±1.88	72.73±0.77
Classification	69.83±0.36	5.26±0.54	**72.20±0.66**	70.96±0.52	5.10±0.57	72.96±0.86

TABLE 4: Evaluation of DualNet's slow learner on the Split miniImageNet TA and TF benchmarks

DualNet	Split miniImageNet-TA		
	ACC(↑)	FM(↓)	LA(↑)
Slow + Fast Nets	**73.20±0.68**	**3.86±1.01**	**74.12±0.12**
Slow Net	68.33±0.57	5.12±0.78	69.20±0.32
DualNet	Split miniImageNet-TF		
	ACC(↑)	FM(↓)	LA(↑)
Slow + Fast Nets	**36.86±1.36**	**28.63±2.26**	**63.46±1.97**
Slow Net	27.30±0.25	34.60±1.12	59.70±1.26

result corroborates with our motivation in Sec. 1 that it is more beneficial to separate the two representations into two distinct systems.

4.1.6 Semi-Supervised Continual Learning Setting

In real-world continual learning scenarios, there exist abundant unlabeled data, which are costly and even unnecessary to label entirely. Therefore, a practical continual learning system should be able to improve its representation using unlabeled samples while waiting for the labeled data. To test the performance of existing methods in such scenarios, we create a *semi-supervised continual learning* benchmark, where the data stream contains both labeled and unlabeled data. For this, we consider the Split miniImageNet-TA benchmark but provide labels randomly to a fraction (ρ) of the total samples, which we set to be $\rho = 4\%$, $\rho = 10\%$ and $\rho = 25\%$. The remaining samples are unlabeled and cannot be processed by the baselines we have considered so far. In contrast, such samples can go directly to the DualNet's slow learner to improve its representation while the fast learner stays inactive. Other configurations remain the same as the experiment in Sec. 4.1.2. For this experiment, we also consider and adapt Cassle [55], a method that introduces SSL to continual learning. Note that originally, Cassle was developed for a two phase learning: a pre-training the Cassle loss using the unlabeled data; and a linear evaluation phase. Therefore, we adapt Cassle to our setting by training Cassle simultaneously with supervised learning on the data stream.

Tab. 5 shows the results of this experiment. Under the limited labeled data regimes (e.g. $\rho = 4\%$ or $\rho = 10\%$),

TABLE 5: Evaluation metrics on the Split miniImageNet-TA benchmarks under the semi-supervised setting, where ρ denotes the fraction of data that is labeled

Method	$\rho = 4\%$		
	ACC(↑)	FM(↓)	LA(↑)
ER	40.26±0.27	2.76±1.24	34.69±1.09
DER++	39.46±0.42	3.92±0.46	33.02±1.01
CTN	47.86±0.66	2.96±1.01	44.56±0.29
Cassle	43.70±2.11	4.02±1.99	44.70±1.09
DualNet	51.79±0.58	2.66±0.97	**45.77±0.37**
DualNet++	**53.13±0.19**	**1.86±0.49**	44.79±0.21
Method	$\rho = 10\%$		
	ACC(↑)	FM(↓)	LA(↑)
ER	41.66±2.72	6.80±2.07	42.33±1.51
DER++	44.56±1.41	4.55±0.66	43.03±0.71
CTN	49.80±2.66	3.96±1.16	47.76±0.99
Cassle	45.80±1.86	5.90±0.96	46.66±1.07
DualNet	54.03±2.88	3.46±1.17	49.96±0.17
DualNet++	**58.03±0.99**	**2.16±0.59**	**53.56±0.11**
Method	$\rho = 25\%$		
	ACC(↑)	FM(↓)	LA(↑)
ER	50.13±2.19	6.76±1.51	51.90±2.16
DER++	51.63±1.11	6.03±1.46	52.36±0.55
CTN	55.90±0.86	3.84±0.32	55.69±0.98
Cassle	48.33±2.27	8.26±0.72	52.10±2.04
DualNet	62.80±2.40	3.13±0.99	59.60±1.87
DualNet++	**63.96±1.22**	**2.20±0.32**	**60.66±1.02**

the results of ER and DER++ drop significantly. Meanwhile, CTN can still maintain competitive performances thanks to additional information from the task identifiers, which remains untouched. On the other hand, both DualNet and DualNet++ can efficiently leverage the unlabeled data to improve its performance and outperform other baselines, even CTN. We also observe that Cassle achieves competitive performances in the low data regimes thanks to its strong SSL loss. However, since Cassle does not store past samples, its performance quickly degraded and is outperformed by other baselines when more labeled samples are available ($\rho = 25\%$). Lastly, DualNet++ consistently outperforms DualNet, and the gaps are larger as the labeled samples are more limited. This gap is attributed to the dropout layers

in DualNet++, which improve the fast learner's ability to use the slow features and to better perform supervised learning. Overall, the result demonstrates DualNets potential to work in a real-world environment, where data is partially labeled.

TABLE 6: Performance of the Offline model under different configuration on the Split miniImageNet-TA benchmark, * denotes the method is trained in the continual learning setting - for reference

Architecture	Method	Loss	Data Aug	ACC
Fast+Slow nets	DualNet*	SL+SSL	Yes	73.20±0.68
	Offline	SL	No	75.83±1.07
	Offline	SL	Yes	77.63±0.48
	Offline	SL+SSL	Yes	**77.98±0.16**
Slow net	Offline	SL	No	71.15±2.95
	Offline	SL	Yes	75.46±0.97

4.1.7 DualNet's Upper Bound

In our work, there are three factors affecting the DualNets' upper bound: (i) model architecture: slow net (standard backbone) versus fast and slow nets (DualNets); (ii) training loss: supervised learning loss (SL) or supervised and self-supervised learning losses (SL+SSL); and (iii) data augmentation. As a result, we believe that an upper bound of DualNet is a model having all three factors as DualNet (has fast and slow learners, optimized both SL and SSL losses with data augmentation) and is trained offline. The offline model has access to all tasks' data to simultaneously optimizes both the SL and SSL losses, which are backpropagated through both learners. Here we consider the offline model trained up to five epochs.

We explore different combinations of the aforementioned factors to train an Offline model on the Split miniImageNet-TA benchmark and report the result in Tab. 6. Note that the configuration of Slow Net + Offline + SL + no data augmentation is the previous result reported in [43]. Our argued upper bound for DualNet has the following configuration: Fast + Slow nets + Offline + SL + SSL + data augmentation. The result confirms the upper bound of DualNet. Moreover, in the offline training with all data, the SSL only contributes a minor improvement to the SL. However, in continual learning, SSL is more beneficial because its representation does not depend on the class label, and therefore more resistant to catastrophic forgetting when old task data is limited.

4.2 Batch Continual Learning Experiments

We now consider the *Batch Continual Learning setting* [13], where all data samples of a task arrive at each continual learning step. As a result, the model is allowed to train on these samples for multiple epochs before moving on to the next task. Thus, most methods do not suffer from the difficulties of training deep neural networks online and focus only on preventing catastrophic forgetting [38].

4.2.1 Setups

The CTrL Benchmark In the batch learning setting, we focus on exploring DualNet's ability to facilitate knowledge transfer in complex continual learning scenarios. To this end, we consider the CTrL benchmark [2], which was carefully designed to access the model's ability to selectively transfer knowledge while avoiding catastrophic forgetting. Before introducing the CTrL streams, we briefly summarize the concept of related tasks used in CTrL.

We denote a continual learning task as $\mathcal{T}$. Then, CTrL introduces four variants of $\mathcal{T}$ as: (1) $\mathcal{T}^-$: a task whose data is sampled from the same distribution as $\mathcal{T}$ but has a much smaller number of training samples; (2) $\mathcal{T}^+$ is similar to $\mathcal{T}^-$ but has much more training samples than $\mathcal{T}$; (3) $\mathcal{T}'$ is similar to $\mathcal{T}$ but has a different input distribution, i.e., different background colors; and (4) $\mathcal{T}''$ is similar to $\mathcal{T}$ but has a different output distribution, i.e., the label order is randomly permuted. In addition, there are no relationships between two tasks that have different subscripts. With this notation, the CTrL benchmarks introduce five continual learning streams to evaluate five basic knowledge transfer abilities comprehensively.

In the $\mathcal{S}^- = \{\mathcal{T}_1^+, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_5, \mathcal{T}_1^-\}$ stream, the last task is similar to the first one but it has much smaller training samples. Therefore, successful methods must remember and transfer the knowledge after learning four unrelated tasks.

In the $\mathcal{S}^+ = \{\mathcal{T}_1^-, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_5, \mathcal{T}_1^+\}$ stream, the first task is similar but has much smaller data than the last one, which requires the model to remember and update the knowledge after learning irrelevant tasks.

The $\mathcal{S}^{\text{in}} = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_5, \mathcal{T}_1'\}$ and $\mathcal{S}^{\text{out}} = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_5, \mathcal{T}_1''\}$ streams require the model to learn representations that are useful to either input or output distribution shifts.

Lastly, the $\mathcal{S}^{\text{pl}} = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_5\}$ stream test the model's ability to learn unrelated tasks with the potential interference from unrelated features. Tab. 7 provides the details of each stream in the CTrL benchmark. Interestingly, the $\mathcal{S}^-, \mathcal{S}^+, \mathcal{S}^{\text{in}}$ and $\mathcal{S}^{\text{out}}$ streams contain one unrelated task from the DTD dataset [87], which has very different visual features from the remaining tasks (see Tab. 7). Therefore, we believe the CTrL benchmark can access DualNets' ability to selective transfer useful knowledge under the presence of negative transfer.

Baselines We follow the experimental setting in [3] and compare DualNets with a suite of competitive baselines. First, we consider static architecture approaches of **EWC** [13] and **O-EWC** [90] (online EWC), which use a quadratic regularizer to penalize changes to important parameters of previous tasks according to the Fisher information. The original EWC [13] maintains an estimate of the Fisher information matrix for each task, while O-EWC maintains a moving average of the parameters importance. Next, we include experience replay **ER** [39], dark experience replay **DER++** [38], **CLSER** [48], and the naive **Finetune** strategy that trains a single model without any continual learning strategies. We also consider the task-free and task-aware variants of these baselines.

Second, we consider a suite of dynamic architecture approaches. The **Independent** [14] baseline trains a separate model for each task. HAT [29] proposes to learn hard attention masks to gate the backbone network, which prevents catastrophic forgetting. **SG-F** [91] proposes a task-specific structural network that learns to update existing modules, combine modules, and add new modules. **MNTDP** [2] organizes the backbone network into modules and efficiently

TABLE 7: Details of the CTrL benchmark streams built from five common datasets: CIFAR-10 [85], MNIST [86], DTD [87], F-MNIST (Fashion MNIST) [88], and SVHN [89]

Stream	Configuration	T1	T2	T3	T4	T5	T6
$\mathcal{S}^{+}$	**Dataset**	CIFAR-10	MNIST	DTD	F-MNIST	SVHN	CIFAR-10
	# Train samples	4000	400	400	400	400	400
	# Val. samples	2000	200	200	200	200	200
$\mathcal{S}^{-}$	**Dataset**	CIFAR-10	MNIST	DTD	F-MNIST	SVHN	CIFAR-10
	# Train samples	400	400	400	400	400	4000
	# Val. samples	200	200	200	200	200	2000
$\mathcal{S}^{\text{in}}$	**Dataset**	R-MNIST	CIFAR-10	DTD	F-MNIST	SVHN	R-MNIST
	# Train samples	4000	400	400	400	400	50
	# Val. samples	2000	200	200	200	200	30
$\mathcal{S}^{\text{out}}$	**Dataset**	CIFAR-10	MNIST	DTD	F-MNIST	SVHN	CIFAR-10
	# Train samples	4000	400	400	400	400	400
	# Val. samples	2000	200	200	200	200	200
$\mathcal{S}^{\text{pl}}$	**Dataset**	MNIST	DTDd	F-MNIST	SVHN	CIFAR-10	-
	# Train samples	400	400	400	400	4000	-
	# Val. samples	200	200	200	200	2000	-

searches the path configuration to connect the modules to solve a given task. Lastly, **LMC** [3], a recent dynamic architecture method that proposes to equip each module with a local structural component to predict its relevance for a given input, which does not require task identifier at test time [2] and provide a more systematic strategy to expand and search the layout over modules.

Training We follow the training procedure provided in [3] for a fair comparison. Particularly, for each task, we train each method over 100 epochs using the Adam optimizer [92]. We use a weak data augmentation of random flipping and random cropping for both the supervised learning phase of all methods and the self-supervised learning phase of DualNet. Furthermore, since DualNets are allowed to train for 100 epochs, it is sufficient to use the standard SGD to optimize the SSL loss. Our preliminary experiments show that in this setting, the Look-ahead and SGD optimizers achieve similar results. We also observe that CLSER has several hyper-parameters, and there are not any guidelines to set them for new applications. Thus, we considered the hyper-parameter configurations provided in [48] and reported the one with the best ACC(↑) . For the backbone network, we consider two variants: a full ResNet18 trained from scratch or pre-trained on ImageNet.

Regarding the model complexity, we anchor on the final model of LMC, the state-of-the-art method on this benchmark, to calculate the total parameters used. Then, we select the replay buffer size of each method so that their total parameters [3] equals to LMC. We repeat each experiment five times and report the average ACC(↑) and BWT(↑) at the end of learning. The metrics are defined in Sec. 4.1.

4.2.2 *Evaluation Metrics on CTrL*

Tab. 8 reports the evaluation metrics at the end of training on the CTrL benchmarks. We organize the results into three blocks: (i) task-free setting with a randomly initialized backbone; (ii) task-free setting with a pre-trained network from ImageNet; and (iii) task-aware setting with a randomly initialized network. In general, the results show that dynamic architecture methods, especially recent works such as the shared-head variants of MNTDP [2] and LMC [3], can perform competitively on this benchmark and outperforms the static architecture approaches. Among replay-based methods, we observe that, compared to the vanilla ER, CLSER effectively handled different distribution shifts while offering limited improvements in handling limited training samples for learning from unrelated tasks. On the other hand, Cassle can slightly outperforms ER and DER++ in some cases thanks to the SSL learning component, which is optimized for many iterations in this batch learning setting. However, both CLSER and Cassle still perform worse than our DualNets, which shows the benefits of our fast-and-slow learning framework in addressing challenging transfer scenarios. Overall, DualNet and DualNet++ outperform all task-free baselines considered with only one exception in the $\mathcal{S}^{\text{out}}$stream where they were outperformed by EWC and O-EWC. Moreover, in most cases, DualNet++ achieves improvements on ACC(↑) and BWT(↑) over DualNet, indicating the benefits of its spatial dropout layers in preventing negative transfer in continual learning. In the task-aware setting, we observe many competitive baselines with high performance. Notably, most baselines in this category are dynamic architecture approaches, which have long dominated the CTrL benchmark. Nevertheless, DualNet++ can perform favorably and achieve top-2 performances in several streams. To the best of our knowledge, this is the first result showing a static architecture method consistently performing comparable with the dynamic architecture ones on the CTrL benchmarks. We also highlight that although they achieved strong results, dynamic architecture methods in the Task-aware category exhibit high variance on different runs, which arises from them training larger models on a small number of samples. On the other hand, DualNets perform consistently and have small variances in all cases.

We also select ER and DER++ and compare them with DualNets when using the pre-trained ImageNet as an initialization. We observe that this strategy generally improves the performances in scenarios where the first task has limited

2. LMC still requires task identifiers during training, which we consider as a task-aware method.

3. total parameters = model parameters + memory parameters in floating point numbers.

TABLE 8: Evaluation metrics on the CTrL benchmark, we report the average accuracy ACC(↑) and backward transfer BWT(↑) at the end of training. The (H) suffix denotes the hard module selection of LMC, * denotes methods that use more parameters. We highlight the methods with best mean metrics in bold, and underline the second best methods

Method	$\mathcal{S}^-$		$\mathcal{S}^+$		$\mathcal{S}^{\text{in}}$		$\mathcal{S}^{\text{out}}$		$\mathcal{S}^{\text{pl}}$	
	ACC(↑)	BWT(↑)	ACC(↑)	BWT(↑)	ACC(↑)	BWT(↑)	ACC(↑)	BWT(↑)	ACC(↑)	BWT(↑)
Task-free & Train from scratch										
Finetune	47.5±1.5	-14.9±1.4	31.4±3.7	-29.3±3.8	39.7±5.0	-23.9±5.7	45.4±4.0	-15.5±3.7	29.1±3.1	-29.2±3.2
Finetune-L	52.1±1.4	-15.7±1.7	38.2±3.2	-25.08±3.3	49.3±2.0	-18.4±2.0	49.3±2.1	-18.4±2.0	37.1±2.1	-26.0±2.2
ER	61.5±0.5	-3.1±1.2	59.9±0.5	0.2±0.9	50.0±1.4	-12.6±0.2	51.0±0.5	-6.3±1.2	54.6±2.2	-5.7±2.3
DER++	63.0±0.7	-2.6±0.4	59.9±0.9	0.1±0.9	55.8±2.6	-10.5±1.9	55.4±0.6	-0.9±0.4	58.1±1.7	-3.3±0.6
CLSER	62.1±0.5	-3.2±0.4	59.1±1.2	0.1±0.9	58.0±1.3	-7.7±1.5	51.0±0.8	-4.9±0.6	57.6±0.4	-1.2±1.1
MNTDP	41.9±2.5	-2.8±0.6	43.2±1.3	-10.8±2.0	32.7±13.6	-15.2±13.2	37.9±2.7	-5.8±3.5	35.1±3.6	-16.4±4.6
SG-F	29.5±3.5	-35.3±4.0	20.4±4.4	-39.3±6.7	24.4±5.6	-38.7±4.0	30.5±4.5	-34.0±5.5	19.4±1.0	-41.8±1.6
Cassle	65.2±0.5	-4.2±0.5	60.1±1.5	-0.1±1.7	56.4±4.0	-9.3±4.0	53.4±0.9	-9.1±2.1	58.5±0.9	-2.9±0.9
DualNet	65.5±0.6	-1.8±0.4	61.1±0.7	1.3±0.6	59.1±0.5	-6.8±0.5	55.7±0.4	-0.7±0.6	58.9±2.2	-1.8±0.1
DualNet++	**68.7±0.7**	**-0.1±0.1**	**62.9±0.8**	**2.9±1.4**	**61.6±1.1**	**-5.1±0.6**	**57.3±0.9**	**-0.6±0.3**	**59.7±1.2**	**0.2±0.4**
Task-free & Initialized from a pre-trained ImageNet model										
ER	62.5±0.8	-2.6±0.8	62.2±1.3	0.7±1.4	52.4±3.8	-10.9±3.4	51.3±2.2	-6.8±1.8	55.2±1.8	-4.8±1.2
DER++	64.0±0.7	-2.4±1.3	63.3±0.8	**0.9±0.9**	57.1±0.9	-10.2±0.9	54.6±1.4	-4.7±1.1	58.9±2.8	-4.5±2.3
DualNet	66.7±0.7	-2.4±0.2	67.9±0.2	0.6±0.5	62.8±0.9	-6.4±1.1	55.2±1.0	-3.2±0.6	63.9±2.1	-2.7±0.7
DualNet++	**68.3±0.5**	**-2.2±0.7**	**68.3±0.4**	0.6±0.6	**63.1±0.7**	**-6.2±1.5**	**55.3±0.6**	**-3.1±0.7**	**64.2±2.4**	-2.4±0.5
Task-aware & Train from scratch										
Independent*	62.7±0.9	0.0	63.2±0.8	0.0	63.1±0.7	0.0	63.1±0.7	0.0	63.9±0.5	0.0
EWC	62.7±0.7	-3.6±0.9	53.4±1.8	-2.3±0.4	56.3±2.5	-9.1±3.3	**62.5±0.9**	-3.6±0.9	52.3±1.4	-5.7±1.3
O-EWC	62.0±0.7	-3.2±0.7	54.6±0.7	-1.3±1.0	54.2±3.1	-10.8±3.1	62.4±0.4	-3.0±0.9	52.3±1.4	-5.7±1.3
HAT	63.7±0.7	-1.3±0.6	61.4±0.5	-0.2±0.2	50.1±0.8	0.0±0.1	61.9±1.3	-3.2±1.3	61.2±0.7	-0.1±0.2
SG-F	63.6±1.5	0.0	61.5±0.6	0.0	65.5±1.8	0.0	64.1±0.0	0.0	62.0±1.3	0.0
MNTDP	66.3±0.8	0.0	62.6±0.8	0.0	63.1±0.7	0.0	63.1±0.7	0.0	63.9±0.5	0.0
LMC	67.2±1.5	-0.5±0.4	62.2±4.5	2.3±1.6	68.5±1.7	-0.1±0.1	55.1±3.4	-7.4±4.0	**63.5±1.9**	-1.0±1.5
LMC(H)	64.9±1.9	-0.2±0.2	55.8±2.5	-0.3±1.2	67.6±2.7	-0.8±1.0	54.2±3.6	-2.9±2.0	53.8±5.7	3.1±5.5
ER	62.9±0.4	-0.7±1.1	55.9±1.2	1.7±0.9	54.8±3.2	-4.2±3.7	60.7±1.0	-1.5±0.5	55.6±1.3	-1.2±1.5
DER++	65.3±0.5	0.2±0.7	60.2±1.4	0.9±0.5	59.0±3.8	-3.8±2.0	64.7±1.4	0.4±0.2	58.5±1.2	0.0±1.7
CLSER	63.9±0.7	-0.1±0.9	56.7±1.1	1.9±0.7	60.2±1.8	-3.6±1.7	61.3±1.7	-0.2±0.9	58.2±1.1	-0.4±1.1
Cassle	66.4±1.2	-3.2±1.0	62.1±0.7	2.5±0.5	61.7±1.2	-5.7±0.8	65.8±0.9	-3.5±1.5	58.8±0.5	-1.2±0.5
DualNet	68.1±0.7	0.2±0.4	62.2±1.0	**4.6±1.3**	63.8±1.6	-3.4±0.8	67.3±0.7	**0.7±1.5**	59.6±0.9	-1.2±0.1
DualNet++	**69.6±1.1**	**1.3±0.8**	**63.3±0.9**	4.3±0.5	64.1±2.2	**-3.4±1.1**	**68.1±0.5**	**0.7±1.8**	62.2±0.6	**0.1±1.1**

training data ($\mathcal{S}^+$), there are input distribution shifts ($\mathcal{S}^{\text{in}}$), or learning from unrelated tasks ($\mathcal{S}^{\text{pl}}$). On the other hand, when the first task has abundant training samples ($\mathcal{S}^-$), using a pre-trained network only offered marginal improvements. Lastly, the $\mathcal{S}^{\text{out}}$ stream has sufficient training samples in the first task and introduces a label permutation, which a pre-trained network may not address. In this case, we did not observe significant differences between using a pre-trained network or training from scratch.

4.2.3 Transfer Results on CTrL

We now take a closer look at the transferring capabilities of different methods on the CTrL benchmark. Recall that in the $\mathcal{S}^-$, $\mathcal{S}^+$, $\mathcal{S}^{\text{in}}$, $\mathcal{S}^{\text{out}}$ streams, only the first and last tasks are closely related, while the intermediate ones are distractors. Therefore, the ACC(↑) metric alone might not be sufficient to evaluate the model's ability to transfer because it also measures the performance of unrelated tasks. Thus, in these streams, it is helpful to look at the accuracy of the first and last task explicitly [3], and compare them with a reference model, Independent, that trains a separate model for each task. We report the results of this experiment in Tab. 9. We can see that except for the $\mathcal{S}^+$stream, both DualNet and DualNet++ achieve significantly better accuracy on the last task and have a higher differences (Δ) compared to the reference model. On the $\mathcal{S}^+$stream, we observe that DualNets are marginally worse than a few baselines, suggesting that remembering and continuing to learn from a limited representation is challenging for DualNets. This phenomenon is easy to understand since learning good representations from limited data with distractors is a challenging problem. Overall, the results suggest that DualNets can remember long-term knowledge in several cases and learn robust representations to distribution shifts, which facilitates successful continual learning in complex scenarios. Moreover, retaining and continuing learning from limited experiences (the $\mathcal{S}^+$stream) presents a promising future research direction.

4.2.4 Robustness to The Dropout Ratio

In literature, the dropout ratio for fully connected layers are commonly set as $p = 0.5$ while this value is set to lower values, e.g. $p = 0.1$ or $p = 0.15$, for convolutional layers [75]. We now investigate how this hyper-parameter affects DualNet++ performance. We consider the task-aware setting and report DualNet++ with different dropout ratios in Tab. 10. It is worth noting that by removing the dropout layer (setting $p = 0.0$), DualNet++ reduces to DualNet. The results show that DualNet++ is robust to and beneficial from the small dropout ratios. Specifically, DualNet++ achieves

TABLE 9: Transfer results on the CTrL benchmark. All methods are task-aware. We report the accuracy of the first and last tasks of each stream and Δ, the difference between the last task accuracy from the model compared to the reference **Independent** model. We highlight the best mean metrics in bold, and underline the second best results

Method	$\mathcal{S}^-$			$\mathcal{S}^+$			$\mathcal{S}^{\text{in}}$			$\mathcal{S}^{\text{out}}$		
	ACC T1	ACC T6	Δ	ACC T1	ACC T6	Δ	ACC T1	ACC T6	Δ	ACC T1	ACC T6	Δ
Independent	65.5±0.7	41.8±1.0	0	41.3±2.9	65.6±0.5	0	98.5±0.2	76.9±4.9	0	65.9±0.6	43.5±1.6	0
MNTDP	63.0±3.6	56.9±5.1	15.1	**43.2±0.7**	**65.9±0.8**	**0.3**	**98.9±0.1**	93.3±1.6	16.4	65.0±1.2	57.7±1.7	14.2
LMC	65.2±0.4	60.0±1.1	18.2	42.9±0.9	60.6±1.9	-4.7	98.7±0.1	92.5±7.6	15.6	65.2±0.2	59.8±1.1	16.3
LMC(H)	62.2±0.4	63.0±1.7	21.2	43.1±0.6	62.2±0.7	-3.4	98.7±0.1	88.3±1.6	11.4	65.5±0.6	42.0±21.9	-1.5
SG-F	64.9±0.4	49.1±7.3	7.3	43.1±0.4	61.7±1.7	-3.9	98.8±0.1	80.4±6.8	3.5	65.0±0.4	51.5±6.5	8
DER++	68.5±1.5	69.9±0.8	28.1	36.7±2.4	58.8±1.7	-6.8	98.1±0.2	93.4±2.3	16.5	67.4±2.3	66.9±2.0	23.4
DualNet	71.8±0.8	71.9±0.8	**27.2**	41.2±0.4	64.5±0.8	-1.1	**98.9±0.1**	**94.8±2.4**	**17.9**	69.6±1.8	68.4±1.7	24.9
DualNet++	**72.6±0.9**	**72.8±0.6**	**31.0**	40.0±0.1	64.8±0.8	-0.8	**98.9±0.1**	**94.8±1.5**	**17.9**	**71.0±1.4**	**71.4±1.2**	**27.9**

TABLE 10: DualNet++ with different dropout ratio p on the CTrL benchmark using the task-aware evaluation. With the ratio $p = 0.0$, DualNet++ reduces to the standard DualNet. Best mean results are highlighted in bold

DualNet++(A)	$\mathcal{S}^-$		$\mathcal{S}^+$		$\mathcal{S}^{\text{in}}$		$\mathcal{S}^{\text{out}}$		$\mathcal{S}^{\text{pl}}$	
	ACC(↑)	BWT(↑)	ACC(↑)	BWT(↑)	ACC(↑)	BWT(↑)	ACC(↑)	BWT(↑)	ACC(↑)	BWT(↑)
$p = 0.0$	68.1±0.7	0.2±0.4	62.2±1.0	4.6±1.3	63.8±1.6	-3.4±0.8	67.3±0.7	0.7±1.5	58.6±0.9	-1.2±0.1
$p = 0.1$	68.8±0.8	0.7±0.3	61.6±1.1	1.3±0.7	63.6±2.0	**4.7±0.6**	**69.1±1.4**	**1.4±0.6**	61.5±1.8	1.4±0.6
$p = 0.2$	**69.6±1.1**	**1.3±0.8**	**63.3±0.9**	**4.3±0.5**	**64.1±2.2**	-3.4±1.1	68.1±0.5	0.7±1.8	**62.2±0.6**	0.1±1.1
$p = 0.3$	66.8±1.2	-1.0±0.7	61.2±0.6	3.5±0.2	63.5±1.6	-2.6±0.7	67.6±0.8	0.4±0.5	61.3±1.4	0.6±0.5
$p = 0.5$	67.1±0.6	-0.7±1.1	60.7±1.0	3.6±0.3	63.2±0.2	-1.8±1.1	67.2±1.2	-0.2±0.7	61.2±0.9	0.7±0.1

similar performances when $p = 0.1$ and $p = 0.2$, and both can outperform the standard DualNet ($p = 0.0$). When using larger dropout ratio, e.g. $p = 0.3$ and $p = 0.5$, we observe a performance drop on all streams in the CTrL benchmark, which is consistent with the conventional usage of dropout layers in convolutional networks.

4.3 Summary of Results

We now provide a summary of the experimental results.

We have conducted experiments on various continual learning settings and examined different scenarios, ranging from the traditional settings to the complex scenarios of semi-supervised learning or complex transfer scenarios. In most cases, DualNet and DualNet++ outperform the baselines, often quite significantly. There is an exception of the $\mathcal{S}^+$and $\mathcal{S}^{\text{in}}$streams where the model needs to learn from limited data with different input distributions, which presents an interesting future work. We also found DualNets to be robust to the choice of SSL loss, an benefited from the LA optimizer and more SSL training iterations in the online continual learning setting. Between DualNet and DualNet++, DualNet++ achieves similar performances to DualNet in the controlled environment where labeled data is plentiful. On the other hand, DualNet++ offers significant improvements in scenarios where labeled data is limited (semi-supervised setting) or there exists negative knowledge transfer from unrelated tasks (CTrL benchmark).

5 Conclusion

Inspired by the Complementary Learning System theory, we propose a novel fast-and-slow learning framework for continual learning and conceptualize it into the DualNets paradigm. DualNets (DualNet and DualNet++) comprise two key learning components: (i) a slow learner that focuses on learning a general and task-agnostic representation using the memory data; and (ii) a fast learner focuses on capturing new supervised learning knowledge via a novel adaptation mechanism. Moreover, the fast and slow learners complement each other while working synchronously, resulting in a holistic continual learning method. Our experiments on challenging benchmarks demonstrate the efficacy of DualNets. Lastly, extensive and carefully designed ablation studies show that DualNets are robust the hyper-parameter configurations, scalable with more resources, and can work well in several challenging continual learning scenarios.

Limitations and Future Work Because the DualNet's slow learner can always be trained in the background, it incurs computational costs that need to be properly managed. For large-scale systems, such additional computations can increase infrastructural costs substantially. Therefore, it is important to manage the slow learner to balance between performance and computational overheads. In addition, implementing DualNet to specific applications requires additional considerations to address its inherent challenges. For example, medical image analysis applications may require paying attention to specific regions in the image or considering the data imbalance. However, since we demonstrate the efficacy of DualNet in general settings, such properties are not considered. In practice, it would be more beneficial to capture such domain-specific information to achieve better results. For general applications, we also expect that a more sophisticated objective to train the slow learner can further improve the results or greatly reduces the training complexity, which is an important direction towards reducing the training costs of DualNets. Lastly, through extensive experiments, we identified the scenario of $\mathcal{S}^{\text{in}}$and $\mathcal{S}^+$, continual learning from limited data under distribution shifts, to be challenging for DualNets. This suggests a promising future research direction to develop a better, more robust representation learning from limited training samples and can be robust to distribution shifts [93].

Algorithm 1: Pseudo-code to train DualNets.

Algorithm `TrainDualNet` $(\boldsymbol{\theta}, \boldsymbol{\phi}, \mathcal{D}^{tr}_{1:T})$
  Require: slow learner $\boldsymbol{\phi}$, fast learner $\boldsymbol{\theta}$, episodic memory $\mathcal{M}$, inner updates N, Look-ahead inner updates K (for Look-ahead)
  Init: $\boldsymbol{\theta}, \boldsymbol{\phi}, \mathcal{M} \leftarrow \varnothing$
  for $t \leftarrow 1$ ***to*** T **do**
    for $j \leftarrow 1$ ***to*** $n_{batches}$ **do** // Receive the dataset D^{tr}_t sequentially
      Receive a mini batch of data $\mathcal{B}_j$ from $\mathcal{D}^{tr}_t$
      $\mathcal{M} \leftarrow$ **MemoryUpdate**$(\mathcal{M}, \mathcal{B}_j)$ // Update the episodic memory
      for $i \leftarrow 1$ ***to*** ∞ **do** // Train the slow learner synchronously
        Train the slow learner using the `Look-ahead procedure`
      for $n \leftarrow 1$ ***to*** N **do** // Train the fast learner synchronously
        $\boldsymbol{M}_n \leftarrow$ Sample$(\mathcal{M})$
        $\mathcal{B}_n \leftarrow \boldsymbol{M}_n \cup \mathcal{B}_j$
        SGD update the slow learner: $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \nabla_{\boldsymbol{\phi}} \mathcal{L}_{tr}(\mathcal{B}_n)$
        SGD update the fast learner $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \nabla_{\boldsymbol{\theta}} \mathcal{L}_{tr}(\mathcal{B}_n)$
    $\mathcal{M}^{em}_t \leftarrow \mathcal{M}^{em}_t \cup \{\pi(\hat{y}/\tau)\}$
  return $\boldsymbol{\theta}, \boldsymbol{\phi}$

Procedure `Look-ahead` $(\boldsymbol{\phi}, \mathcal{M})$
  $\tilde{\boldsymbol{\phi}}_0 \leftarrow \boldsymbol{\phi}$
  for $k \leftarrow 1$ ***to*** $K-1$ **do**
    $\boldsymbol{M}_k \leftarrow$ Sample$(\mathcal{M}) \cup \mathcal{B}_j$
    Obtains two views of $\boldsymbol{M}_k$: $\boldsymbol{M}^A_k, \boldsymbol{M}^B_k$
    Calculate the Barlow Twins loss: $\mathcal{L}_{\mathcal{BT}}(\tilde{\phi}_k, \boldsymbol{M}^A_k, \boldsymbol{M}^B_k)$
    SGD update the slow learner: $\tilde{\boldsymbol{\phi}}_{k+1} \leftarrow \boldsymbol{\phi}_k - \epsilon \nabla_{\boldsymbol{\phi}_k} \mathcal{L}_{\mathcal{BT}}$
  Look-ahead update the slow learner: $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \beta(\tilde{\boldsymbol{\phi}}_K - \boldsymbol{\phi})$
  return $\boldsymbol{\phi}$

APPENDIX A
DUALNETS PSEUDO CODE

We provide DualNets' pseudo-code in Algorithm 1.

ACKNOWLEDGMENTS

The first author, Quang Pham, gratefully acknowledges the support by the Lee Kuan Yew Fellowship awarded by Singapore Management University.

REFERENCES

[1] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly, "Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory." *Psychological review*, vol. 102, no. 3, p. 419, 1995.

[2] T. Veniat, L. Denoyer, and M. Ranzato, "Efficient continual learning with modular networks and task-driven priors," in *International Conference on Learning Representations*, 2020.

[3] O. Ostapenko, P. Rodriguez, M. Caccia, and L. Charlin, "Continual learning via local module composition," *Advances in Neural Information Processing Systems*, vol. 34, pp. 30 298–30 312, 2021.

[4] R. J. Douglas, C. Koch, M. Mahowald, K. Martin, and H. H. Suarez, "Recurrent excitation in neocortical circuits," *Science*, vol. 269, no. 5226, pp. 981–985, 1995.

[5] D. Kumaran, D. Hassabis, and J. L. McClelland, "What learning systems do intelligent agents need? complementary learning systems theory updated," *Trends in cognitive sciences*, vol. 20, no. 7, pp. 512–534, 2016.

[6] R. C. O'Reilly, R. Bhattacharyya, M. D. Howard, and N. Ketz, "Complementary learning systems," *Cognitive science*, vol. 38, no. 6, pp. 1229–1248, 2014.

[7] D. N. Barry and E. A. Maguire, "Remote memory and the hippocampus: A constructive critique," *Trends in cognitive sciences*, vol. 23, no. 2, pp. 128–142, 2019.

[8] S. Tonegawa, M. D. Morrissey, and T. Kitamura, "The role of engram cells in the systems consolidation of memory," *Nature Reviews Neuroscience*, vol. 19, no. 8, pp. 485–498, 2018.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[10] D. Sahoo, Q. Pham, J. Lu, and S. C. H. Hoi, "Online deep learning: Learning deep neural networks on the fly," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, 2018.

[11] R. Aljundi, E. Belilovsky, T. Tuytelaars, L. Charlin, M. Caccia, M. Lin, and L. Page-Caccia, "Online continual learning with maximal interfered retrieval," in *Advances in Neural Information Processing Systems*, 2019, pp. 11 849–11 860.

[12] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[13] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, 2017.

[14] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 6467–6476.

[15] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, 2019.

[16] K. Javed and M. White, "Meta-learning representations for continual learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 1818–1828.

[17] D. Rao, F. Visin, A. Rusu, R. Pascanu, Y. W. Teh, and R. Hadsell, "Continual unsupervised representation learning," *Advances in Neural Information Processing Systems*, 2019.

[18] A. Gepperth and C. Karaoguz, "A bio-inspired incremental learning

architecture for applied perceptual problems," *Cognitive Computation*, vol. 8, no. 5, pp. 924–934, 2016.

[19] G. I. Parisi, J. Tani, C. Weber, and S. Wermter, "Lifelong learning of spatiotemporal representations with dual-memory recurrent self-organization," *Frontiers in neurorobotics*, vol. 12, p. 78, 2018.

[20] K. Javed and F. Shafait, "Revisiting distillation and incremental classifier learning," in *Asian conference on computer vision*. Springer, 2018, pp. 3–17.

[21] T. Diethe, T. Borchert, E. Thereska, B. Balle, and N. Lawrence, "Continual learning in practice," *arXiv preprint arXiv:1903.05202*, 2019.

[22] Q. Pham, C. Liu, and S. Hoi, "Dualnet: Continual learning, fast and slow," *Advances in Neural Information Processing Systems*, vol. 34, pp. 16 131–16 144, 2021.

[23] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with a-gem," *International Conference on Learning Representations (ICLR)*, 2019.

[24] M. Delange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[25] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[26] J. Yoon, E. Yang, J. Lee, and S. J. Hwang, "Lifelong learning with dynamically expandable networks," *International Conference on Learning Representations (ICLR)*, 2018.

[27] X. Li, Y. Zhou, T. Wu, R. Socher, and C. Xiong, "Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting," in *International Conference on Machine Learning*, 2019, pp. 3925–3934.

[28] C. Fernando, D. Banarse, C. Blundell, Y. Zwols, D. Ha, A. A. Rusu, A. Pritzel, and D. Wierstra, "Pathnet: Evolution channels gradient descent in super neural networks," *arXiv preprint arXiv:1701.08734*, 2017.

[29] J. Serra, D. Suris, M. Miron, and A. Karatzoglou, "Overcoming catastrophic forgetting with hard attention to the task," in *Proceedings of the 35th International Conference on Machine Learning-Volume 80*. JMLR. org, 2018, pp. 4548–4557.

[30] J. von Oswald, C. Henning, J. Sacramento, and B. F. Grewe, "Continual learning with hypernetworks," *International Conference on Learning Representations (ICLR)*, 2020.

[31] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3987–3995.

[32] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 139–154.

[33] H. Ritter, A. Botev, and D. Barber, "Online structured laplace approximations for overcoming catastrophic forgetting," in *Advances in Neural Information Processing Systems*, 2018, pp. 3738–3748.

[34] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Machine learning*, vol. 8, no. 3-4, pp. 293–321, 1992.

[35] M. Riemer, I. Cases, R. Ajemian, M. Liu, I. Rish, Y. Tu, and G. Tesauro, "Learning to learn without forgetting by maximizing transfer and minimizing interference," *International Conference on Learning Representations (ICLR)*, 2019.

[36] Y. Liu, Y. Su, A.-A. Liu, B. Schiele, and Q. Sun, "Mnemonics training: Multi-class incremental learning without forgetting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 245–12 254.

[37] G. M. van de Ven and A. S. Tolias, "Generative replay with feedback connections as a general strategy for continual learning," *arXiv preprint arXiv:1809.10635*, 2018.

[38] P. Buzzega, M. Boschini, A. Porrello, D. Abati, and S. Calderara, "Dark experience for general continual learning: a strong, simple baseline," in *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020.

[39] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. K. Dokania, P. H. Torr, and M. Ranzato, "On tiny episodic memories in continual learning," *arXiv preprint arXiv:1902.10486*, 2019.

[40] C. de Masson D'Autume, S. Ruder, L. Kong, and D. Yogatama, "Episodic memory in lifelong language learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[41] F.-K. Sun, C.-H. Ho, and H.-Y. Lee, "Lamol: Language modeling for lifelong language learning," in *International Conference on Learning Representations*, 2019.

[42] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne, "Experience replay for continual learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 348–358.

[43] Q. Pham, C. Liu, D. Sahoo, and S. C. Hoi, "Contextual transformation networks for online continual learning," *International Conference on Learning Representations (ICLR)*, 2021.

[44] H. Yin, P. Li *et al.*, "Mitigating forgetting in online continual learning with neuron calibration," *Advances in Neural Information Processing Systems*, vol. 34, pp. 10 260–10 272, 2021.

[45] E. Arani, F. Sarfraz, and B. Zonooz, "Learning fast, learning slow: A general continual learning method based on complementary learning system," in *International Conference on Learning Representations*, 2022.

[46] T. L. Hayes, K. Kafle, R. Shrestha, M. Acharya, and C. Kanan, "Remind your neural network to prevent catastrophic forgetting," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VIII 16*. Springer, 2020, pp. 466–483.

[47] R. Kemker and C. Kanan, "Fearnet: Brain-inspired model for incremental learning," in *International Conference on Learning Representations*, 2018.

[48] E. Arani, F. Sarfraz, and B. Zonooz, "Learning fast, learning slow: A general continual learning method based on complementary learning system," in *International Conference on Learning Representations*, 2022.

[49] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?" in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 201–208.

[50] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[51] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[52] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1126–1135.

[53] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[54] X. He, J. Sygnowski, A. Galashov, A. A. Rusu, Y. W. Teh, and R. Pascanu, "Task agnostic continual learning via meta learning," *arXiv preprint arXiv:1906.05201*, 2019.

[55] E. Fini, V. G. T. da Costa, X. Alameda-Pineda, E. Ricci, K. Alahari, and J. Mairal, "Self-supervised models are continual learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9621–9630.

[56] H. Cha, J. Lee, and J. Shin, "Co2l: Contrastive continual learning," in *Proceedings of the IEEE/CVF International conference on computer vision*, 2021, pp. 9516–9525.

[57] P. S. Bhat, B. Zonooz, and E. Arani, "Task agnostic representation consolidation: a self-supervised based continual learning approach," in *Conference on Lifelong Learning Agents*. PMLR, 2022, pp. 390–405.

[58] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[59] J. Requeima, J. Gordon, J. Bronskill, S. Nowozin, and R. E. Turner, "Fast and flexible multi-task classification using conditional neural adaptive processes," in *Advances in Neural Information Processing Systems*, 2019, pp. 7959–7970.

[60] V. Dumoulin, E. Perez, N. Schucher, F. Strub, H. d. Vries, A. Courville, and Y. Bengio, "Feature-wise transformations," *Distill*, 2018, https://distill.pub/2018/feature-wise-transformations.

[61] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," in *Advances in Neural Information Processing Systems*, 2017, pp. 506–516.

[62] M. Caccia, P. Rodriguez, O. Ostapenko, F. Normandin, M. Lin, L. Caccia, I. Laradji, I. Rish, A. Lacoste, D. Vazquez *et al.*, "Online fast adaptation and knowledge accumulation: a new approach to continual learning," *arXiv preprint arXiv:2003.05856*, 2020.

[63] Q. Pham, D. Sahoo, C. Liu, and S. C. Hoi, "Bilevel continual learning," *arXiv preprint arXiv:2007.15553*, 2020.

[64] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 532–547.

[65] R. Aljundi, K. Kelchtermans, and T. Tuytelaars, "Task-free continual learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 254–11 263.

[66] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[67] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[68] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," *Advances in Neural Information Processing Systems*, 2020.

[69] F. M. Carlucci, A. D'Innocente, S. Bucci, B. Caputo, and T. Tommasi, "Domain generalization by solving jigsaw puzzles," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2229–2238.

[70] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow twins: Self-supervised learning via redundancy reduction," *arXiv preprint arXiv:2103.03230*, 2021.

[71] Y. You, I. Gitman, and B. Ginsburg, "Large batch training of convolutional networks," *arXiv preprint arXiv:1708.03888*, 2017.

[72] M. Zhang, J. Lucas, J. Ba, and G. E. Hinton, "Lookahead optimizer: k steps forward, 1 step back," *Advances in Neural Information Processing Systems*, 2019.

[73] L. Wang, M. Zhang, Z. Jia, Q. Li, C. Bao, K. Ma, J. Zhu, and Y. Zhong, "Afec: Active forgetting of negative transfer in continual learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 22 379–22 391, 2021.

[74] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[75] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 648–656.

[76] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[77] X. Li, S. Chen, X. Hu, and J. Yang, "Understanding the disharmony between dropout and batch normalization by variance shift," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 2682–2690.

[78] S. I. Mirzadeh, M. Farajtabar, and H. Ghasemzadeh, "Dropout as an implicit gating mechanism for continual learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 232–233.

[79] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching networks for one shot learning," in *Advances in neural information processing systems*, 2016, pp. 3630–3638.

[80] V. Lomonaco and D. Maltoni, "Core50: a new dataset and benchmark for continuous object recognition," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research. PMLR, 2017, pp. 17–26.

[81] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, pp. 211–252, 2015.

[82] J. S. Vitter, "Random sampling with a reservoir," *ACM Transactions on Mathematical Software (TOMS)*, vol. 11, no. 1, pp. 37–57, 1985.

[83] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[84] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 750–15 758.

[85] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[86] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[87] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 3606–3613.

[88] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[89] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," 2011.

[90] F. Huszár, "Note on the quadratic penalties in elastic weight consolidation," *Proceedings of the National Academy of Sciences*, vol. 115, no. 11, pp. E2496–E2497, 2018.

[91] J. A. Mendez and E. Eaton, "Lifelong learning of compositional structures," *arXiv preprint arXiv:2007.07732*, 2020.

[92] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR (Poster)*, 2015.

[93] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. [Online]. Available: http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf

Quang Pham Quang Pham is currently a research scientist at Institute for Infocomm Research (I^2R), Agency for Science, Technology and Research (A*STAR). He obtained his PhD degree from School of Computing and Information Systems at Singapore Management University. His research interests include continual learning and deep learning.

Chenghao Liu Chenghao Liu is currently a senior applied scientist of Salesforce Research Asia. Before, he was a research scientist in the School of Information Systems (SIS), Singapore Management University (SMU), Singapore. He received his Bachelor degree and Ph.D degrees from the Zhejiang University. His research interests include large-scale machine learning (online learning and deep learning) with application to tackle big data analytics challenges across a wide range of real-world applications.

Steven C. H. Hoi Steven C. H. Hoi is currently the Managing Director of Salesforce Research Asia, and a Professor of Information Systems at Singapore Management University, Singapore. He received his Bachelor degree from Tsinghua University, P.R. China, in 2002, and his Ph.D degree in computer science and engineering from The Chinese University of Hong Kong, in 2006. He has served as the Editor-in-Chief for Neurocomputing Journal, general co-chair for ACM SIGMM Workshops on Social Media, program co-chair for the fourth Asian Conference on Machine Learning, book editor for "Social Media Modeling and Computing", guest editor for ACM Transactions on Intelligent Systems and Technology. He is an IEEE Fellow and ACM Distinguished Member.