Class-Incremental Learning with Cross-Space Clustering and Controlled Transfer

Arjun Ashok, K J Joseph, Vineeth N Balasubramanian
Indian Institute of Technology Hyderabad
arjun.ashok@cse.iith.ac.in, {cs17m18p100001, vineethnb}@iith.ac.in

Abstract

In class-incremental learning, the model is expected to learn new classes continually while maintaining knowledge on previous classes. The main challenge here lies in preserving the ability of the model to effectively represent prior classes in the feature space, while adapting it to represent new classes as well. In this paper, we develop two distillation-based objectives for class incremental learning that leverage the structure of the feature space to both maintain accuracy on previous classes as well as enable learning of new classes. In our first objective termed cross-space clustering, we propose to use the entire feature space of the previous model to characterize specific regions that all instances of a class should optimize toward, and regions that they should stay away from. This enables the model to reliably cluster all instances of a class in the current space, and further gives rise to a sense of "herd-immunity", allowing all samples of a class to jointly combat the model from forgetting the class. As part of our second objective termed controlled transfer, we tackle incremental learning from the novel perspective of inter-class transfer. We develop an objective that explicitly estimates and conditions the model on the semantic similarities between incrementally arriving classes and prior classes. This allows the model to learn the incoming classes in such a way that it maximizes positive forward transfer from similar prior classes, and minimizes negative backward transfer on dissimilar prior classes. We perform extensive experiments on two benchmark datasets, adding our method on top of three prominent class-incremental learning methodologies, and show that our method improves performance consistently on wide range of settings. Our code is available at this http URL.

1. Introduction

Incremental learning is a paradigm of machine learning where learning objectives are introduced to a model incrementally in the form of *phases* or *tasks*, and the model must possess the ability to dynamically learn new tasks while maintaining knowledge on previously seen tasks. The differences of this setup from a *static training* scenario is that no information about the tasks are available upfront, and the model is not allowed to *retrain* from scratch on encountering new tasks. A fundamental dilemma in incremental learning is the stability-plasticity trade-off [30], where stability concerns maintaining accuracy on the previous tasks, and plasticity concerns learning the current task completely. In their naive form, deep learning models are too plastic, incurring *catastrophic forgetting* [16] of old tasks when exposed to new ones, as the model changes significantly during training.

Class-incremental learning (CIL) [35, 29] is a specific sub-paradigm of incremental learning where tasks are composed of new classes and we seek to learn a *unified* model that can represent and classify all classes seen so far equally well. The main challenge in class-incremental learning lies in how knowledge over a long stream of classes must be consolidated. Regularization-based methods [21, 4, 50, 9] quantify and preserve important parameters corresponding to prior tasks. Another set of approaches [28, 10, 14, 43] focus on modifying the learning algorithm to ensure that gradient updates do not conflict with the previous tasks. In dynamic architecture methods [48, 45, 25, 1, 34], the network architecture is modified by expansion or masking when encountering new tasks during learning. Replay-based methods [35, 18, 13, 44, 6, 7, 27, 8, 24, 2, 40, 39, 47] store a subset of each previous task in a separate memory, and replay the tasks when learning a new one, to directly preserve the knowledge on those tasks. A wide variety of such methods have been developed recently, and have attained promising results in the CIL setting. A number of these methods [35, 18, 13, 7, 2, 40, 8, 24] use variants of knowledge distillation [17], where the model corresponding to the previous task is stored and utilized to prevent the current task's model from diverging too much from its previous state.

Our work herein falls under distillation-based methods. Prior work has advocated for utilizing distillation to directly

constrain an example's position or angle with its previous position in the feature space [18], to preserve pooled convolutional outputs of an instance [13], or to maintain the distribution of logits that the model's classifier outputs on the data [2, 35]. We argue that preserving the features or predictions of a model on independent individual instances are only useful to a certain extent, and do not characterize and preserve properties of a class captured globally by the model as a whole. Class-level semantics may be more important to be preserved in the class-incremental learning setting, to holistically prevent individual classes from being forgotten. To this end, we develop an objective termed **Cross-Space Clustering** (CSC) that uses points spanning the feature space to characterize *entire regions* that an example should stay away from, and those that the example should belong to – to ensure that the class representation is well-preserved. Our objective indirectly establishes multiple goals at once: (i) it encourages the model to cluster all instances of a given class; (ii) ensures that these clusters are well-separated; and (iii) regularizes to preserve *class cluster positions* as a single entity in the feature space. This provides for a class-consolidated distillation objective, prodding instances of a given class to "unite" and thus prevent the class from being forgotten.

Next, as part of our second objective, we tackle the class-incremental-learning problem from a different perspective. While all prior distillation objectives seek better ways to preserve properties of representations in the feature space [35, 18, 13, 7, 2, 40, 8, 24], we believe that controlling *inter-class transfer* is also critical for class-incremental learning. This comes from the observation that forgetting often results from *negative backward transfer* from new classes to previous classes, and plasticity is ensured when there is *positive forward transfer* from prior classes to new ones [28]. To this end, we develop an objective called **Controlled Transfer** (CT) that controls and regularizes transfer of features between classes at a fine-grained level. We formulate an objective that estimates the relative similarity between an incoming class and all previous classes, and conditions the current task's model on these estimated similarities. This encouraging *new classes* to be situated optimally in the feature space, ensuring maximal positive transfer and minimal negative transfer.

A unique characteristic of our objectives is their ability to extend and enhance existing distillation-based CIL methodologies, without any change to their methodologies. We verify this by adding our objectives to three prominent and state-of-the art CIL methods that employ distillation in their formulation: iCARL [35], LUCIR [18] and POD-Net [13]. We conduct thorough experimental evaluations on benchmark incremental versions of large-scale datasets like CIFAR-100 and ImageNet subset. We perform a comprehensive evaluation of our method, considering a wide variety of experimental settings. We show that our method consistently improves incremental learning performance across datasets and methods, at no additional cost. We further analyze and present ablation studies on our method, highlighting the contribution of each of our components.

2. Related Work

2.1. Incremental Learning

In the incremental learning setting, a model is required to consistently learn new tasks, without compromising performance on the old tasks. Incremental learning methodologies can be split into five major categories, each of which we review below.

Regularization-based methods focus on quantifying the importance of each parameter in the network, to prevent the network from excessively changing the important parameters pertaining to a task. These methods include EWC [21], SI [50], MAS [4] and RWalk [9]. These importance estimates are used later to constrain the appropriate weights when learning a new task.

Algorithm-based methods comprise of methods that seek to avoid forgetting from the perspective of the network training algorithm. They modify gradients such that updates to weights do not not deteriorate performance on previously seen tasks. Methods such as GEM [28], A-GEM [10], OGD [14] and NSCL [43] fall under this category.

Architecture-based methods, modify the network architecture dynamically to fit more tasks, by expanding the model by adding more weights [48, 45], or masking and allocating subnetworks to specific tasks [38], or by gating the parameters dynamically using a task identifier [1].

Exemplar-based methods (also called replay-based or rehearsal methods) assume that a small subset of the class can be stored in a memory. They replay the class later along with the incoming new classes, directly preventing them from being forgetten. One set of works focus on reducing the recency bias due to the new classes being in majority at every phase [44, 6, 18, 7]. Another set of works focus on optimizing which samples to choose as exemplars to better represent the class distributions [27, 5].

Distillation-based methods use the model learned until the previous task as a teacher and provide extra supervision to the model learning the current tasks (the student). Since the entire datasets of the previous tasks are inaccessible, these methods typically enforce distillation objectives on the current data [26, 12], data from an exemplar memory [18, 13, 35, 7, 2], external data [24] or synthetic data [51]. Since our method falls under this category, we extend our discussion on related methods below.

Early works in this category distill logit scores [26, 35] or attention maps [12] of the previous model. iCARL [35] proposes to enforce distillation on new tasks as well exem-

plars from old tasks, along with herding selection of exemplars and nearest-mean-of-exemplars (NME) based classification. GD [24] calibrate the confidence of the model's outputs using external unlabelled data, and propose to distill the calibrate outputs instead. LUCIR [18] introduces a less-forget constraint that encourages the orientation of a sample in the current feature space to be similar to the sample's orientation in the old feature space. Apart from that, LUCIR proposes to use cosine-similarity based classifiers and a margin ranking loss that mines hard negatives from the new classes to better separate the old class to additionally avoid ambiguities between old and new classes. PODNet [13] preserves an example's representation throughout the model with a spatial distillation loss. The authors of SSIL [2] show that general KD preserves the bias due to additional classes, and propose to use task-wise KD. Co2L [8] introduces a contrastive learning based self-supervised distillation loss that preserves the exact feature relations of a sample with its augmentations and other samples from the dataset. The authors of GeoDL [40] introduce a term that enhances knowledge distillation by performing KD across low-dimensions path between the subspaces of the two models, considering the gradual shift between models.

The main difference of our cross-space clustering objective from these works is that we do not optimize to preserve the properties of individual examples, and instead preserve the previously learned semantics or properties of each class in a *holistic manner*. Our formulation takes into account the global position of a class in the feature space, and optimizes all samples of the class towards the same region, making the model indifferent to instance-level semantics. Further, classes are supervised with specific "negative" regions all over the feature space, also intrinsically giving rise to better separation between class clusters.

Our controlled transfer objective, on the other hand, attempts to regularize transfer between tasks. MER [36], an algorithm-based method is related to our high-level objective. MER works in the online continual learning setup, combining meta-learning [15, 31] with replay. Their method optimizes such that the model receives weight updates are restricted to those directions that agree with the gradients of prior tasks. We tackle a similar objective, however in a different perspective of using the *structure* of the feature space of the previous model to align the current feature space, in order to maximize transfer. Our novelty here lies in how we explicitly estimate *inter-class semantic similarities* in a continual task stream, and utilize them to appropriately position new tasks representations, regularizing transfer.

2.2. Knowledge Distillation

Hinton et al. [17] introduced knowledge distillation (KD) in their work as a way to transfer knowledge from an ensemble of teacher networks to a smaller student network. They show that there is dark knowledge in the logits of a trained network that can give more structure about the data, and use them as soft targets to train the student. Since then, a number of other works have explored variants of KD. Attention Transfer [49] focused on the attention maps of the network instead of the logits, while FitNets [37] also deal with intermediate activation maps of a network. Several other papers have enforced criteria based on multi-layer representations of a network [46, 19, 20, 3, 22].

Among these, our controlled transfer objective shares similarities with a few works that propose to exploit the mutual relation between data samples for distillation. Tung and Mori [41] propose a distillation objective that enforces an L2 loss in the student that constraints the similarities between activation maps of two examples to be consistent with those of the teacher. The authors of Relational KD [32] additionally propose to preserve the angle formed by the three examples in the feature space by means of a triplet distillation objective. Extending this direction, Correlation Congruence [33] models relations between examples by means of kernels, to enforce the same objectives with better relation estimates.

The difference of our controlled transfer objective from these works lies in the high-level objective in the context of the incremental learning setting, as well as the low-level formulation in terms of the loss objective. All the above works propose to use sample relations in the feature space to provide additional supervision to a student model that learns the same classes from scratch, by regularizing the feature relations of the student. Our objective also exploits sample relations in the feature space, however, it does not seek to *preserve* feature relations between data points or to *model* higher-order similarities in the feature space, which are not relevant in the incremental learning scenario. The main challenge in incremental learning is in how we can reduce the effect that a new class has on the representation space, to minimize forgetting.

Our novelty lies in how we estimate a measure of relative similarity between an *unseen class* and each previously seen class, and utilize them to control where the *new samples* are located in the embedding space, in relation to the old samples. Our specific formulation indirectly promotes forward transfer of features from prior classes similar to the new class, and prevents negative backward transfer of features from the new class to dissimilar previous classes.

3. Method

After a brief introduction to the problem setting in Sec. 3.1, we explain in detail each of our objectives in Sec. 3.2 and Sec. 3.3 respectively, and discuss the final objective function in Sec. 3.4.

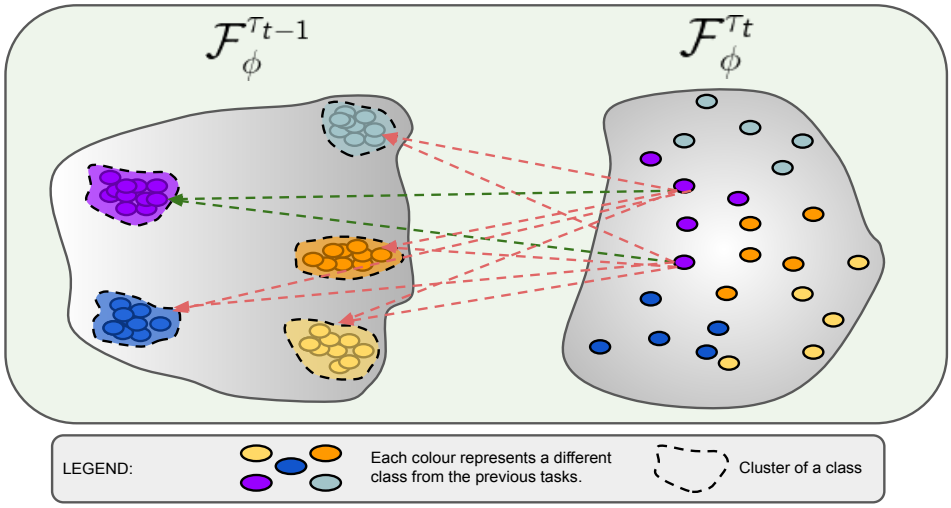

Figure 1. We illustrate our cross-space clustering (CSC) objective. We show instances from 5 different classes and their positions in the the spaces of $\mathcal{F}_{\phi}^{\mathcal{T}_{t-1}}$ and $\mathcal{F}_{\phi}^{\mathcal{T}_t}$ respectively. Classes are well represented in $\mathcal{F}_{\phi}^{\mathcal{T}_{t-1}}$, however their representations are dispersed in the $\mathcal{F}_{\phi}^{\mathcal{T}_t}$. Here we illustrate the constraint imposed on an instance of the violet class, based on the cluster position of its own class (indicated by the green arrows) and the positions of every other class (indicated by the red arrows). Note how the exact same constraint is applied on all instances of a class (illustrated here with 2 instances of the violet class). (best viewed in color)

3.1. Problem Setting

In the incremental learning paradigm, a set of tasks $\mathcal{T}_t = \{\tau_1, \tau_2, \cdots, \tau_t\}$ is introduced to the model over time, where $\mathcal{T}_t$ represents the tasks that the model has seen until time step t. τ_t denotes the task introduced at time step t, which is composed of images and labels sampled from its corresponding task data distribution: $\tau_t = (\mathbf{x}_i^{\tau_t}, y_i^{\tau_t}) \sim p_{data}^{\tau_t}$. Each task τ_t contains instances from a disjoint set of classes. $\mathcal{F}^{\mathcal{T}_t}$ denotes the model at time step t, once it has learned the set of tasks $\mathcal{T}_t$. Without loss of generality, $\mathcal{F}^{\mathcal{T}_t}$ can be expressed as a composition of two functions: $\mathcal{F}^{\mathcal{T}_t}(\mathbf{x}) = (\mathcal{F}_{\boldsymbol{\phi}}^{\mathcal{T}_t} \circ \mathcal{F}_{\boldsymbol{\theta}}^{\mathcal{T}_t})(\mathbf{x})$, where $\mathcal{F}_{\boldsymbol{\phi}}^{\mathcal{T}_t}$ represents a feature extractor, and $\mathcal{F}_{\boldsymbol{\theta}}^{\mathcal{T}_t}$ denotes a classifier. The challenge in incremental learning is to learn a model that can represent and classify all seen classes equally well, at any point in the task stream.

While training $\mathcal{F}^{\mathcal{T}_t}$ on the current task τ_t, the model does not have access to all the data from previous tasks. Exemplar-based methods [35, 5, 44, 27, 6] sample a very small coreset of each task data $e_t \in \tau_t$ at the end of task τ_t and store it in a memory buffer $\mathcal{M}_t = \{e_1, e_2, \cdots, e_t\}$, which contains the coresets of all tasks seen until time t. When learning a new task at time step t, the task's data τ_t is combined with samples from the memory containing exemplars of each previous task $\mathcal{M}_{t-1}$. Therefore, the dataset that the model has at time step t is $\mathcal{D}_t = \tau_t \cup \mathcal{M}_{t-1}$. In distillation-based methods, we assume access to the previous model $\mathcal{F}^{\mathcal{T}_{t-1}}$ which has learned the stream of tasks $\mathcal{T}_{t-1}$. The model $\mathcal{F}^{\mathcal{T}_{t-1}}$ is frozen and not updated, and is instead used to guide the learning of the current model $\mathcal{F}^{\mathcal{T}_t}$. Typically, distillation-based methods constrain the model by distilling features [18, 13], attention maps [12] or logits [26] of data points in the current dataset $\mathcal{D}_t$. Effectively utilising the previous model is key to balancing stability and plasticity. Excess constraints tied to the previous model can prevent the current task from being learned, and poor constraints can lead to easy forgetting of previous tasks.

3.2. Cross-Space Clustering

Our *cross-space clustering* objective leverages the entire feature space of $\mathcal{F}^{\mathcal{T}_{t-1}}$, to identify specific regions that all instances of a class are optimized to stay within, and other harmful regions that they are prevented from drifting towards. Since all the samples from a dataset cannot be accessed at once, we instead approximate points from the entire data distribution using mini-batches. We illustrate our cross-space clustering objective in Fig. 1.

Consider that the model $\mathcal{F}^{\mathcal{T}_t}$ is trained on mini-batches $\{x_i, y_i\}_{i=1}^k$ sampled from D_t. Our cross-space clustering objective enforces the following loss on the model:

$$L_{CSC} = \frac{1}{k^2} \sum_{i=1}^{k} \sum_{j=1}^{k} \Bigg(\Big(1 - \cos(\mathcal{F}_{\boldsymbol{\phi}}^{\mathcal{T}_t}(x_i), \mathcal{F}_{\boldsymbol{\phi}}^{\mathcal{T}_{t-1}}(x_j)) \Big) * ind(y_i == y_j) \Bigg) \quad (1)$$

where ind is an indicator function that returns **1** when its inputs are equal and **-1** otherwise, and $\cos(a, b)$ denotes the

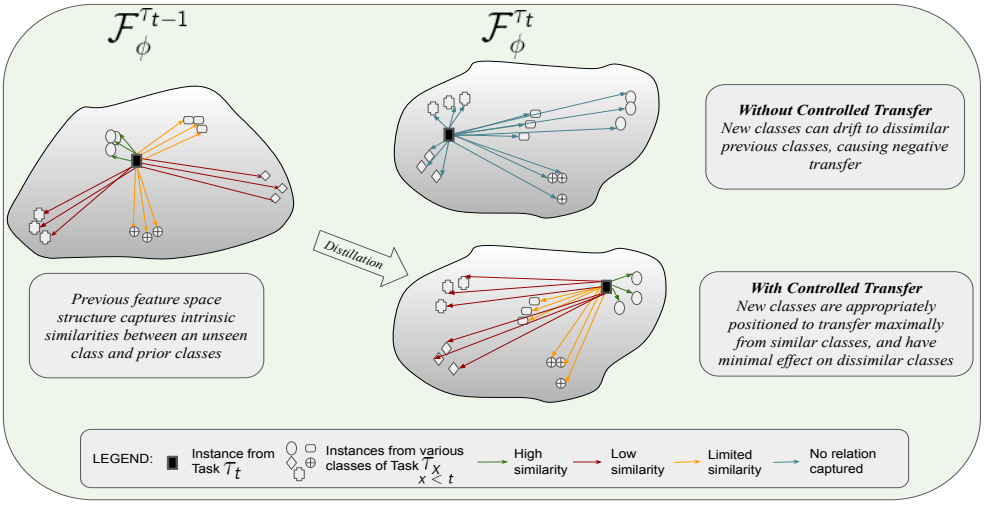

Figure 2. We illustrate our **controlled transfer** objective. We show the positions of instances from five random classes taken from previous tasks $\tau_x; x < t$, and one unseen incoming class from the current task τ_t, in $\mathcal{F}_{\phi}^{\mathcal{T}_{t-1}}$ and $\mathcal{F}_{\phi}^{\mathcal{T}_t}$ respectively. With our objective, the new task instances are regularized to position themselves appropriately, to prevent negative transfer to dissimilar classes, and to encourage positive transfer from similar classes (best viewed in color)

cosine similarity between two vectors a and b.

Physical Interpretation: For pairs of samples x_i and x_j, $\mathcal{F}^{\mathcal{T}_t}(x_i)$ is enforced to minimize cosine distance with $\mathcal{F}^{\mathcal{T}_{t-1}}(x_j)$ when they are of the same class ($y_i == y_j$), and maximize cosine distance with $\mathcal{F}^{\mathcal{T}_{t-1}}(x_j)$ when they are of different classes ($y_i ! = y_j$). We expand upon the objective and its implications separately below.

Explanation: Consider that there are l examples of class n in the considered batch k, and hence $k-l$ samples belonging to classes other than n.

First of all, sample $x_i \in n$ is allowed to see the previous feature positions of all of the l samples of the same class, and is regularized to be *equally close* to all these positions. Since multiple positions are used in the previous feature space and equal constraints are applied, points only see an approximation of its class (cluster) in the previous feature space, and do not see individual feature positions. This inherently removes the dependency of the distillation on the specific position of the sample within its class, and instead optimizes the sample to move towards a point that can preserve the class as a whole. This also serves a more flexible constraint, as an example is not required to maintain its exact same position [18, 13], and can move freely within the characterized region.

Next, *every sample* x_i belonging to a class n in the batch is given the *exact same constraints* with no difference. This leads to all of them being optimized *jointly* to a single stark region belonging to their class. Repeating this process for several iterations implicitly leads to model to implicitly *cluster* all samples of a class in the *current* feature space $\mathcal{F}^{\mathcal{T}_t}$ in the specific characterized regions. With respect to clustering, an important point is that our loss is *cross-space* in the sense, it does not encourage clustering of features of a class using features from the same model [8], as clustering within the same model neither exploits prior knowledge about the classes, nor imposes any constraints on the location of the clusters. Our formulation instead encourages a model to keep all these clusters at specific points provided by the previous feature space, thereby directly distilling and preserving the *cluster positions* in the feature space as well. Hence, our objective uses approximate cluster positions from $\mathcal{F}^{\mathcal{T}_{t-1}}$ (as explained in the previous paragraph) to in-turn cluster samples at specific positions in $\mathcal{F}^{\mathcal{T}_t}$. Since all samples are optimized to cluster at the same region and preserve the relative position of the region, all points of the class are optimized to *unite* and jointly protect the class. Such a formulation gives rise to a sense of *herd-immunity* of classes from forgetting, which better preserves the classes as the model drifts.

Finally, with very few exemplars stored per-class in the memory, our objective proposes to maximally utilize the entire memory[1] as well as the current task, leveraging them to identify *negative regions* that an instance is maintained to lie away from. In our particular formulation, x_i belonging to class n is enforced to stay equally away from the positions of all other $k - l$ examples from the entire previous space. This indirectly tightens the cluster of class n in $\mathcal{F}^{\mathcal{T}_t}$ along multiple directions in the feature space.

Differences from prior work: Prior distillation-based methods [18, 13, 40] only apply *sample-to-sample* cross-

[1] A batch of sufficient size typically contains at least one sample from each previous class, serving as a rough approximation of the memory

space constraints, to preserve the representational properties of the previous space. The core difference of our method from all others lies in how it applies *class-to-region* constraints. Here, *class* denotes how all samples of a class are jointly optimized with the same constraints, and *region* denotes how the samples are optimized towards and away from specific *regions* instead of towards individual points.

3.3. Controlled Transfer

Catastrophic forgetting has been characterized to arise due to the inability to access enough data of previous tasks [44, 18, 6], change in important parameters [21, 50, 4], representation drift [26, 13, 8], conflicting gradients [28, 10, 14, 43] and insufficient capacity of models [48, 45, 1]. All these works ignore the semantic similarities between tasks and their relation to forgetting. We argue that knowing the degree of semantic similarity between two classes can, in fact, be very useful in incremental learning: When a previous class is *dissimilar* to the class currently being learned, the model must learn to treat that class distinctively and minimally impact it, so that the semantic specialities of that class are not erased. Conversely, when there is a previous class which is *similar* to the class currently being learned, the model must maximally transfer features from that class, to learn the current class in the best possible way. With these goals, we propose an incremental learning objective that *explicitly quantifies* inter-class similarities, and *controls* transfer between classes in every phase. Fig. 2 illustrates our controlled transfer objective.

Notation: We first describe the general notation that we use to denote the similarity between samples in a space. Consider two samples x_i and x_j from a dataset D_k, and a model $\mathcal{F}^{\mathcal{T}_k}$. We denote the similarity between x_i and x_j computed on the feature space of $\mathcal{F}^{\mathcal{T}_k}$ as $z^{\mathcal{T}_k}_{x_i,x_j} = \cos(\mathcal{F}^{\mathcal{T}_k}_{\phi}(x_i), \mathcal{F}^{\mathcal{T}_k}_{\phi}(x_j))$ where $\cos(a,b)$ denotes cosine similarity between two vectors a and b. We denote the the normalized distribution of similarities that an individual sample x_i has with *every sample* in D_k, in the feature space of $\mathcal{F}^{\mathcal{T}_k}$ as

$$H^{\mathcal{T}_k}_{x_i,D_k,T} = \left\{ \frac{(z^{\mathcal{T}_k}_{x_i,x_j}/T)}{\sum_{g=1}^{|D_k|}(z^{\mathcal{T}_k}_{x_i,x_g}/T)} \right\}_{j=1}^{|D_k|} \tag{2}$$

where T is the temperature that is used to control the entropy of the distribution. $H^{\mathcal{T}_k}_{x_i,D_k,T}$ is a row matrix, where the value in each column j of the matrix denotes the normalized similarity between x_i and x_j, relative to every sample in the dataset D_k.

Formulation: We first aim to estimate the similarities between a *new class* $\mathcal{C}_{new} \in \tau_t$ and every previously seen class $\mathcal{C}_{old} \in \tau_k \in \mathcal{T}_{t-1}$. $\mathcal{C}_{old}$ is well represented the model $\mathcal{F}^{\mathcal{T}_{t-1}}$; the new class $\mathcal{C}_{new}$ has not yet been learned by any model. It is not possible to use the drifting feature space of $\mathcal{F}^{\mathcal{T}_t}$ to represent $\mathcal{C}_{new}$; even representing $\mathcal{C}_{new}$ once it has been learned by $\mathcal{F}^{\mathcal{T}_t}$ would heavily bias the representations towards $\mathcal{C}_{new}$ due to the well-known recency bias [44, 2]. Our formulation instead proposes to utilize the *dark knowledge* that the *previous model* possesses about an *unseen class*: if the representations of an unseen class $\mathcal{C}_{new}$ lie relatively close to or overlaps the class representations of a previous class in $\mathcal{F}^{\mathcal{T}_{t-1}}$, it indicates that the two classes share semantic similarities. On the other hand, if the representations of an unseen class $\mathcal{C}_{new}$ lie relatively far from a previous class in $\mathcal{F}^{\mathcal{T}_{t-1}}$, it indicates that the two classes do not have any semantic features in common. We propose to use these approximate similarities captured by $\mathcal{F}^{\mathcal{T}_{t-1}}$ in our objective explained below.

Consider a mini-batch of $B^{\mathcal{T}_t}_n$ of size s that contains samples $\{(x^{\mathcal{T}_t}_i, y^{\mathcal{T}_t}_i)\}$ randomly sampled from D_t. This mini-batch $B^{\mathcal{T}_t}_n$ is composed of p samples from the current task denoted by $P = (x^{\tau_t}_i, y^{\tau_t}_i)^p_{i=1}$, and q samples taken from the memory, denoted by $Q = (x^{\tau_k}_i, y^{\tau_k}_i)^q_{i=1}$, where $k < t$. Due to the majority of samples in D_t belonging to the current task τ_t, in general, $p >> q$. In an effort to control the transfer between a new and an old sample, our objective regularizes the normalized similarity (closeness) that a sample from the current task $(x^{\tau_t}_i, y^{\tau_t}_i) \in \tau_t$ has with every sample from any previous class $(x^{\tau_k}_i, y^{\tau_k}_i)$, where $k < t$. This is enforced by minimizing the KL Divergence of the similarity distribution of $x^{\tau_t}_i \in P$ over Q, in the *current space* $\mathcal{F}^{\mathcal{T}_t}_{\phi}$, with the similarity distribution computed in the *previous space* $\mathcal{F}^{\mathcal{T}_{t-1}}_{\phi}$, as follows

$$L_{Transfer} = \frac{1}{p}\sum_{i=1}^{p} KL(H^{\tau_t}_{x_i,Q,T} || H^{\tau_{t-1}}_{x_i,Q,T}) \tag{3}$$

This loss modifies the position of the current classes in the current feature space $\mathcal{F}^{\mathcal{T}_t}_{\phi}$ such that they have *high similarity* with (lie close to) prior classes that are *very similar*, and have *low similarity* with (lie far from) those previous classes that are *dissimilar* to it. This encourages *positive forward transfer* of features to the current classes from selected previous classes that are similar, as both their embeddings are optimized to have high similarity in the current space. This helps the model learn the current task better by leveraging transferred features, and lessens the impact that the new task has on the representation space. Conversely, this discourages (negative) backward transfer from the current classes to specific dissimilar classes, as their embeddings are optimized to have low similarity in the current space. Consequently, the features of these specific classes are further shielded from being erased, leading to the semantics of those classes being preserved more in the current space, which directly results in lesser forgetting of those classes.

Table 1. The table shows results on **CIFAR100** when our method is added to three top-performing approaches [35, 18, 13]. The red subscript highlights the relative improvement. $\mathcal{B}$ denotes the number of classes in the first task. $\mathcal{C}$ denotes the number of classes in every subsequent task.

Dataset	CIFAR100						ImageNet-Subset					
Settings	$\mathcal{B}=50$			$\mathcal{B}=\mathcal{C}$			$\mathcal{B}=50$			$\mathcal{B}=\mathcal{C}$		
Methods	$\mathcal{C}=1$	$\mathcal{C}=2$	$\mathcal{C}=5$	$\mathcal{C}=1$	$\mathcal{C}=2$	$\mathcal{C}=5$	$\mathcal{C}=1$	$\mathcal{C}=2$	$\mathcal{C}=5$	$\mathcal{C}=1$	$\mathcal{C}=2$	$\mathcal{C}=5$
iCaRL [35]	43.39	48.31	54.42	30.92	36.80	44.19	55.81	57.34	65.97	40.75	55.92	60.93
iCaRL + CSCCT	46.15$_{+2.76}$	51.62$_{+3.31}$	56.75$_{+2.33}$	34.02$_{+3.1}$	**39.60**$_{+2.8}$	46.45$_{+2.26}$	57.01$_{+1.2}$	58.37$_{+1.03}$	66.82$_{+0.8}$	42.46$_{+1.71}$	57.45$_{+1.53}$	62.60$_{+1.67}$
LUCIR [18]	50.26	55.38	59.40	25.40	31.93	42.28	60.44	66.55	70.18	36.84	46.40	56.78
LUCIR + CSCCT	52.95$_{+2.69}$	56.49$_{+1.13}$	62.01$_{+2.61}$	28.12$_{+2.72}$	34.96$_{+3.03}$	44.03$_{+1.55}$	61.52$_{+1.08}$	67.91$_{+1.36}$	71.33$_{+1.15}$	37.86$_{+1.02}$	47.55$_{+1.15}$	58.07$_{+1.29}$
PODNet [13]	56.88	59.98	62.66	33.58	36.68	45.27	67.27	73.01	75.32	44.94	58.23	66.24
PODNet + CSCCT	**58.80**$_{+1.92}$	**61.10**$_{+1.12}$	**63.72**$_{+1.06}$	**36.23**$_{+2.65}$	39.3$_{+2.62}$	**47.8**$_{+2.53}$	**68.91**$_{+1.64}$	**74.35**$_{+1.34}$	**76.41**$_{+1.09}$	**46.06**$_{+1.12}$	**59.43**$_{+1.2}$	**67.49**$_{+1.25}$

3.4. Final Objective Function

The independent nature of our objectives make them suitable to be applied on top of any existing method to improve its performance. Our final objective combines $L_{Cross-Cluster}$ (1) and $L_{Transfer}$ (3) with appropriate coefficients:

$$L_{CSCCT} = L_{method} + \alpha * L_{Cross-Cluster} + \beta * L_{Transfer} \tag{4}$$

where L_{method} denotes the objective function of the specific method used, and α and β are loss coefficients for each of our objectives respectively. We term our method CSCCT, indicating **C**ross-**S**pace **C**lustering and **C**ontrolled **T**ransfer.

4. Experiments and Results

We conduct extensive experiments adding our method to three prominent methods in class-incremental learning [35, 18, 13].

Protocols: In the class-incremental learning setting, prior work has experimented with two protocols: **a)** training with half the total number of classes in the first task, and equal number of classes from each subsequent task [18, 13, 44], and **b)** training with the same number of classes in each task, including the first [35, 2, 7]. We experiment with both these protocols to demonstrate the applicability of our method. The first setting has the advantage of gaining access to strong features in the first task, while the second tests an extreme continual learning setting. Both these settings are plausible in a real-world incremental classification setting. On CIFAR100, the remaining 50 classes in the first setting or the full 100 classes in the second setting are grouped into 1, 2 and 5 classes per task. On ImageNet-Subset, the classes are split into 2, 5 and 10 classes per task. Hence, our experiments are conducted on *long streams of small tasks*, as well as *short streams of large tasks*.

Datasets and Evaluation Metric: Following prior works [18, 13, 35, 7], we test on the incremental versions of CIFAR-100 [23] and ImageNet-Subset [35]. CIFAR100 contains 100 classes, with 500 images per class, and each of dimension 32×32. ImageNet-Subset is a subset of the ImageNet-1k dataset [11], and contains 100 classes, with over 1300 images per class. Each image is of size 224×224. All our results denote average incremental accuracy following prior work [18, 13]. We follow the original papers in their inference methodology: On LUCIR [18] and PODNet [13], classification is performed as usual using the classifier logits, while on iCARL [35], classification is based on nearest-mean-of-exemplars.

Implementation Details: Following prior work [18, 13], we use a ResNet-32 and ResNet-18 on CIFAR100 and ImageNet-Subset respectively. On CIFAR100, we use a batch size of 128 and train for 160 epochs, with an initial learning rate of 0.1 that is decayed by 0.1 at the 80^{th} and 120^{th} epochs respectively. On ImageNet-Subset, we use a batch size of 64 and train for 90 epochs, with an initial learning rate of $2e^{-2}$ that is decayed by 0.1 at the 30^{th} and 60^{th} epochs respectively. All our experiments are reported on an exemplar memory size of 20 examples per class. We use *herding selection* [35] for exemplar sampling. We set loss coefficients α and β to 3 and 1.5 respectively.

4.1. Quantitative Results

We add our method to three state-of-the-art class-incremental learning methodologies: iCARL [35], LUCIR [18] and PODNet [13]. Table 1 showcases results on CIFAR100 and ImageNet-Subset. We see a consistent improvement across all these settings when CSCCT is added to them. Specifically, on CIFAR100, adding CSCCT to iCARL [35], LUCIR [18] and PODNet [13] provides strong relative improvement of 2.76%, 2.28% and 1.99% respectively averaged across all settings, while on the much more high-dimensional ImageNet-Subset, adding our method to the respective baselines provides consistent relative improvements of 1.32%, 1.17% and 1.35%.

Evaluating iCARL [35], LUCIR [18] and PODNet [13] on the equal class protocol show that LUCIR [18] suffers from a severe performance degradation due to its inherent reliance on a large initial task, while iCARL [35] and PODNet [13] do not. On CIFAR100, simply adding our method to iCARL [35] gives it strong boosts of $2.2\% - 3.1\%$ in this setting, bringing it much closer to the state-of-the-art PODNet [13]. Overall, our method improves performance

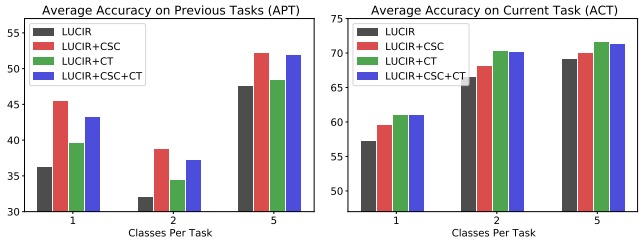

Figure 3. Average accuracy on previous tasks (APT) and average accuracy on the current task (ACT), plotted across various settings on the CIFAR-100 dataset

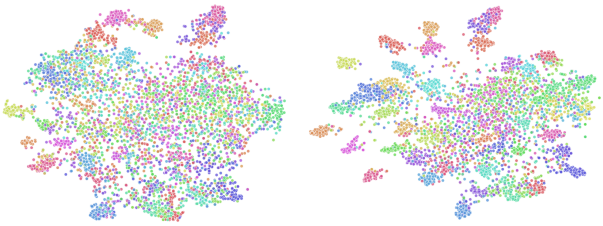

Figure 4. T-SNE [42] visualizations of the base 50 classes of CIFAR100 in the embedding space, after all 100 classes have been learned (**Left**: LUCIR [18], **Right**: LUCIR+CSC)

consistently across both settings, showing that our formulation does not rely on a large initial task to learn strong representations.

5. Ablation Study and Analysis

5.1. Effect of Each Component on Average Incremental Accuracy

Table 2. Ablating each objective on CIFAR100. Maroon denotes 2^{nd} best result.

Settings	$\mathcal{B}=50$			$\mathcal{B}=\mathcal{C}$		
Methods	$\mathcal{C}=1$	$\mathcal{C}=2$	$\mathcal{C}=5$	$\mathcal{C}=1$	$\mathcal{C}=2$	$\mathcal{C}=5$
LUCIR [18]	50.26	55.38	59.4	25.4	31.93	42.28
LUCIR + CSC	52.04	55.95	60.45	27.16	32.89	42.98
LUCIR + CT	51.5	55.87	61.97	26.53	33.98	43.69
LUCIR + CSCCT	**52.95**	**56.49**	**62.01**	**28.12**	**34.96**	**44.03**

In Table 2, we ablate each component of our objective, and show the average incremental accuracy. It can be seen that each of our objectives can improve accuracy independently. In particular, CSC is more effective when the number of classes per task is extremely low, while the CT objective stands out in the improvement it offers when there are more classes per task. Overall, combining our objectives achieves the best performance across all settings.

5.2. Effect of Each Component on Stability/Plasticity

To further investigate how each component is useful specifically in the incremental learning setup, we look into how each component improves the stability and plasticity of the model under various settings. The left plot of Fig. 3 shows the average accuracy on previous tasks (denoted as APT), which is calculated by averaging the mean accuracy on all previous tasks, obtained at the end of *every task* in the stream. This serves as an indicator of the **stability** of the model. Mathematically, APT can be expressed as

$$APT = \frac{\sum_{t=2}^{T}\left(\frac{\sum_{k=1}^{t-1} Acc(\tau_k)}{t-1}\right)}{T-1} \tag{5}$$

where $Acc(\tau_k)$ denotes accuracy on the test set of task k.

The right plot of Fig. 3 shows the average accuracy on the current task (denoted as ACT), which is calculated by averaging the mean accuracy on the tasks calculated immediately after the task has been learned. This is an indicator of the **plasticity** of the model. ACT is expressed as

$$ACT = \frac{\sum_{t=1}^{T} Acc(\tau_t)}{T} \tag{6}$$

Across all considered settings, *both of our objectives* increase *stability* as well as *plasticity* of the base model. However, the effect of the **CSC objective** is much more pronounced on the **stability** of the model. This aligns with intuition that the CSC helps in preserving previous classes better in the representation space. At the same time, the **CT objective** impacts the **plasticity** consistently more than the CSC objective, as it mainly aims at appropriately positioning the current task samples to maximize transfer.

5.3. Embedding Space Visualization

In Fig. 4, we present T-SNE [42] visualizations of the embedding space, without and with our CSC objective (1). The 50 classes learned in the initial task are plotted in the embedding spaces of both models, once all the 100 classes have been learned. It can be seen that applying the CSC objective results in better clusters of prior classes in the feature space, compared to the baseline. The number of overlapping classes are reduced to a significant extent, as our objective ensures that the clusters are well-separated.

6. Conclusion

In this paper, we introduced two complementary distillation-based objectives for class-incremental learning. Our first objective called *cross-space clustering* clusters all instances of a class and preserves cluster positions as a whole using a global view of the representation space, enabling instances to counteract forgetting jointly. Our second objective called *controlled transfer* models relationships between incoming and prior classes, and controls the positive and negative transfer between classes. We perform extensive experiments on two benchmark datasets across a wide range of experimental settings to showcase the effectiveness of our objectives.

Acknowledgements

This work has been partly supported by the funding received from DST and Intel through the IMPRINT program. KJJ thanks TCS for their PhD Fellowship. We are grateful to the anonymous reviewers for their valuable feedback.

References

[1] Davide Abati, Jakub M. Tomczak, Tijmen Blankevoort, Simone Calderara, Rita Cucchiara, and Babak Ehteshami Bejnordi. Conditional channel gated networks for task-aware continual learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3930–3939, 2020.

[2] Hongjoon Ahn, Jihwan Kwak, Su Fang Lim, Hyeonsu Bang, Hyojun Kim, and Taesup Moon. Ss-il: Separated softmax for incremental learning. *ICCV*, 2021.

[3] Sungsoo Ahn, Shell Xu Hu, Andreas C. Damianou, Neil D. Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9155–9163, 2019.

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

[5] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *NeurIPS*, 2019.

[6] Eden Belouadah and Adrian Daniel Popescu. Il2m: Class incremental learning with dual memory. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 583–592, 2019.

[7] Francisco Manuel Castro, Manuel J. Marín-Jiménez, Nicolás Guil Mata, Cordelia Schmid, and Alahari Karteek. End-to-end incremental learning. *ECCV*, 2018.

[8] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. *ICCV*, 2021.

[9] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018.

[10] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-GEM. In *International Conference on Learning Representations*, 2019.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[12] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5133–5141, 2019.

[13] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, 2020.

[14] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3762–3773. PMLR, 2020.

[15] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

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

[17] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, 2015.

[18] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 831–839, 2019.

[19] Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. *ArXiv*, abs/1707.01219, 2017.

[20] Jangho Kim, Seonguk Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. *NeurIPS*, 2018.

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

[22] Animesh Koratana, Daniel Kang, Peter Bailis, and Matei A. Zaharia. Lit: Learned intermediate representation training for model compression. In *ICML*, 2019.

[23] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Citeseer*, 2009.

[24] Kibok Lee, Kimin Lee, Jinwoo Shin, and Honglak Lee. Overcoming catastrophic forgetting with unlabeled data in the wild, 2019.

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

[26] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:2935–2947, 2018.

[27] Yaoyao Liu, Anan Liu, Yuting Su, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12242–12251, 2020.

[28] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

[29] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation on image classification, 2021.

[30] Martial Mermillod, Aurelia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in Psychology*, 2013.

[31] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[32] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3962–3971, 2019.

[33] Baoyun Peng, Xiao Jin, Jiaheng Liu, Shunfeng Zhou, Yichao Wu, Yu Liu, Dongsheng Li, and Zhaoning Zhang. Correlation congruence for knowledge distillation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5006–5015, 2019.

[34] Jathushan Rajasegaran, Munawar Hayat, Salman Hameed Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for continual learning. In *NeurIPS*, 2019.

[35] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, G. Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5533–5542, 2017.

[36] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, , and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019.

[37] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *ICLR*, 2015.

[38] Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. *ICML*, 2018.

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

[40] Christian Simon, Piotr Koniusz, and Mehrtash Harandi. On learning the geodesic path for incremental learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1591–1600, 2021.

[41] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1365–1374, 2019.

[42] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[43] Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 184–193, 2021.

[44] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Raymond Fu. Large scale incremental learning. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 374–382, 2019.

[45] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3013–3022, 2021.

[46] Junho Yim, Donggyu Joo, Ji-Hoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7130–7138, 2017.

[47] Hongxu Yin, Pavlo Molchanov, Zhizhong Li, José Manuel Álvarez, Arun Mallya, Derek Hoiem, Niraj Kumar Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8712–8721, 2020.

[48] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *International Conference on Learning Representations*, 2018.

[49] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *ICLR*, 2017.

[50] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3987–3995. JMLR.org, 2017.

[51] Mengyao Zhai, Lei Chen, Fred Tung, Jiawei He, Megha Nawhal, and Greg Mori. Lifelong gan: Continual learning for conditional image generation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2759–2768, 2019.