Class-Incremental Learning via Knowledge Amalgamation

Marcus de Carvalho[1], Mahardhika Pratama[2], Jie Zhang[1], and Yajuan Sun[3]

[1] Nanyang Technological University, Singapore
{marcus.decarvalho,zhangj}@ntu.edu.sg
[2] University of South Australia, Adelaide, Australia dhika.pratama@unisa.edu.au
[3] A*Star SIMTech, Singapore sun_yajuan@simtech.a-star.edu.sg

Abstract. Catastrophic forgetting has been a significant problem hindering the deployment of deep learning algorithms in the continual learning setting. Numerous methods have been proposed to address the catastrophic forgetting problem where an agent loses its generalization power of old tasks while learning new tasks. We put forward an alternative strategy to handle the catastrophic forgetting with knowledge amalgamation (CFA), which learns a student network from multiple heterogeneous teacher models specializing in previous tasks and can be applied to current offline methods. The knowledge amalgamation process is carried out in a single-head manner with only a selected number of memorized samples and no annotations. The teachers and students do not need to share the same network structure, allowing heterogeneous tasks to be adapted to a compact or sparse data representation. We compare our method with competitive baselines from different strategies, demonstrating our approach's advantages. Source-code: github.com/Ivsucram/CFA

Keywords: Continual Learning · Transfer Learning · Knowledge Distillation.

1 Introduction

Computational learning systems driven by the success of deep learning have obtained great success in several computational data mining and learning system as computer vision, natural language processing, clustering, and many more [1]. However, although deep models have demonstrated promising results on unvarying data, they are susceptible to *catastrophic forgetting* when applied to dynamic settings, i.e., new information overwrites past experiences, leading to a significant drop in performance of previous tasks.

In other words, current learning systems depend on batch setting training, where the tasks are known in advance, and the training data of all classes are accessible. When new knowledge is introduced, an entire retraining process of the network parameters is required to adapt to changes. This becomes impractical in terms of time and computational power requirements with the continual introduction of new tasks.

To overcome catastrophic forgetting, learning agents must integrate continuous new information to enrich the existing knowledge. The model must then prevent the new information from significantly obstructing the acquired knowledge by preserving all or most of it. A learning system that continuously learns about incoming new knowledge consisting of new classes is called a *class-incremental learning* agent.

A class-incremental solution showcases three properties:

1. It should learn from a data stream that introduces different classes at different times,
2. It should provide a multi-class inference for the learned classes at any requested time,
3. Its computational requirements should be bounded, or grow slowly, to the number of classes learned.

Many strategies and approaches in the continual learning field attempt to solve the catastrophic forgetting problem in the class-incremental scenario. Regularization techniques [5] identify essential parameters for inference of previous tasks and avoid perturbing them when learning new tasks. Knowledge distillation methods have also been used [6], where knowledge from previous tasks and incoming tasks are jointly optimized. Inspired by work in reinforcement learning, memory replay has also been an important direction explored by researchers [19], where essential knowledge acquired from previous experiences is re-used for faster training, or retraining, of a learning agent.

In this paper, we propose a catastrophic forgetting solution based on knowledge amalgamation (CFA). Given multiple trained teacher models - each on a previous task - knowledge amalgamation aims to suppress catastrophic forgetting by learning a student model that handles all previous tasks in a single-head manner with only a selected number of memorized samples and no annotations. Furthermore, the teachers and the students do not need to share the same structure so that the student can be a compact or sparse representation of the teachers' models.

A catastrophic forgetting solution based on the knowledge amalgamation approach is helpful because it allows heterogeneous tasks to be adapted to a single-head final model. At the same time, knowledge amalgamation explores the relationship between the tasks without the need for any identifier during the amalgamation process, being smoothly integrated into already existing learning pipelines. This approach can be perceived as a post-processing continual learning solution, where a teacher model is developed for each task and flexibly combined into a single compact model when inference is required. As a result, it does not need to maintain specific network architectures for each task.

Contributions:

- A novel class-incremental learning approach via knowledge amalgamation which:
  - Allows teachers and students to present different structures and tasks;
  - Integrable into existing learning pipelines (including non-continual ones);

2 Related Work

A neural network model needs to learn a series of tasks in sequences in the continual learning setting. Thus, only data from the current task is available during training. Furthermore, classes are assumed to be clearly separated. As a result, catastrophic forgetting occurs when a new task is introduced and the model loses its generalization power of old tasks through learning.

Currently, there are three scenarios in which a continual learning experiment can be configured. *Task-incremental learning* is the easiest of the scenarios, as a model receives knowledge about which task needs to be processed. Models with task-specific components are the standard in this scenario, where the multi-headed output layer network represents the most common solution.

The second scenario, referred to as *domain-incremental learning*, does not have task identity available during inference, and models only need to solve the given task without inferring its task.

Finally, **class-incremental learning**, the third scenario, requires that the models must solve each task seen so far while at the same time inferring its task. The currently proposed method falls into this scenario. Furthermore, most real-world problems of incrementally learning new classes of objects also belong to this scenario.

Existing works to handle the continual learning problem are mainly divided into three categories:

- **Structure-based approach**: One reason for catastrophic forgetting to occur is that the parameters of a neural network are optimized for new tasks and no longer for previous ones. This suggests that not optimizing the entire network or expanding the internal model structure to deal with the new tasks while isolating old network parameters could attenuate catastrophic forgetting. PNN [8] pioneered this approach by adding new components to the network and freezing old task parameters during training. Context-dependent gating (XdG) [17] is a simple but popular approach that randomly assigns nodes to tasks. However, these approaches are limited to the *task-incremental learning* scenario by design, as task identity is required to select the correct task-specific components during training.
- **Regularization-based approach**: When task knowledge is only available during training time, training a different part of the network for each task can still happen, but then the whole network is used through inference. Standard methods in this approach estimate the importance of the network parameters for the previously learned tasks and penalize future changes accordingly. Elastic Weight Consolidation (EWC) [16] and its online counterpart (EWCo) [32] adopt the Fisher information matrix to estimate the importance of the network synapses. Synaptic intelligence (SI) [18] utilizes an accumulated gradient to quantify the significance of the network parameters.
- **Memory-based approach**: This strategy replays old, or augmented, samples stored in memory when learning a new task. Learning without forgetting (LWF) [6] uses a pseudo-data strategy where it labels the samples of

the current tasks using the model trained on the previous tasks, resulting in training that mixes hard-target (likely category according to previous tasks) with soft-target (predicted probabilities within all classes). Gradient episodic memory (GEM) [21] and Averaged GEM (A-GEM) [22] successfully boost continual learning performance by the usage of exact samples stored in memory to estimate the forgetting case and to constraint the parameter updates accordingly. Gradient-based Sample Selection (GSS) [27] focuses on optimizing the selection of samples to be replayed. Dark Experience Replay (DER/DER++) [28] and Function Distance Regularization (FDR) [29] use past samples and soft outputs to align past and current outputs. Hindsight Anchor Learning (HAL) [26] adds additional objectives into replaying, aiming to reduce forgetting of key learned data points.

Alternatively, methods can also take advantage of generative models for pseudo-rehearsal. For example, Deep Generative Replay (DGR) [19] utilizes a separated generative model that is sequentially trained on all tasks to generate samples from their data distribution. Additionally, knowledge distillation can be combined with DGR (DGR+distill) [20] to pair generated samples with soft target knowledge. Alternatively, methods can also take advantage of generative models for pseudo-rehearsal. For example, Deep Generative Replay (DGR) [19] utilizes a separated generative model that is sequentially trained on all tasks to generate samples from their data distribution. Additionally, knowledge distillation can be combined to DGR (DGR+distill) [20] to pair generated samples with soft target knowledge.

The proposed CFA is a memory-based approach that presents a novel way to perform continual learning using knowledge amalgamation, a derivation of knowledge distillation, and domain adaptation to merge several teacher models into a single student model.

2.1 Domain Adaptation

Transfer learning (TL) [2] is defined by the reuse of a model developed for a task to improve the learning of another task. Neural networks have been applied to TL because of their power in representing high-level features.

While there are many sub-topics of TL, we are deeply interested in domain adaptation (DA) [10]. While there are many approaches to measure and reduce the disparity between the distributions of these two domains, Maximum Mean Discrepancy (MMD) [11] and Kullback-Leibler divergence (KL) [12] are widely used in the literature. Our approach uses KL to approximate the representation of learning distributions between the teachers and the student, differing itself from the original knowledge amalgamation method [3], where the MMD approach is applied.

2.2 Knowledge Distillation

Knowledge distillation (KD) [9] is a method of transferring learning from one model to the other, usually by compression, where a larger teacher model supervises the training of a smaller student model. One of the benefits of KD is that it can handle heterogeneous structures, i.e., the teacher and the student do not need to share the same network structure. Instead, the teacher supervises the student training via its logits, also called the soft target. In other words, KD minimizes the distance between the student network output $\hat{z}$ and the logits z from a teacher network, generated from an arbitrary input sample:

$$\mathcal{L}_{KD} = ||\hat{z} - z||_2^2 \tag{1}$$

Although KD has become a field itself in the machine learning community, many approaches are still performed under a single teacher-student relationship, with a sharing task [9]. Contrary to these constraints, our method can process multiple and heterogeneous teachers, condensing their knowledge into a single student model covering all tasks.

2.3 Knowledge Amalgamation

Knowledge Amalgamation (KA) [3, 4] aims to acquire a compact student model capable of handling the comprehensive joint objective of multiple teacher models, each specialized in their task. Our approach extends the concept of knowledge amalgamation in [24, 3] to the continual learning environment.

3 Problem Formulation

In continual learning, within the class-incremental learning scenario, we experience a stream of data tuples (x_i, y_i) that satisfies $(x_i, y_i) \overset{iid}{\sim} P_{t_i}(X, Y)$, containing an input x_i and a target y_i organized into sequential tasks $t_i \in \mathcal{T} = 1, ...T$, where the total number of tasks T is unknown a priori. The goal is to learn a predictor $f : \mathcal{X} \times \mathcal{T} \to \mathcal{Y}$, which can be queried *at any time* to predict the target vector y associated to a test sample x, where $(x, y) \sim P_t$. Such test pair can belong to a task that we have observed in the past or the current task.

We define the knowledge amalgamation task as follows. Assume that we are given N teacher models $t_{i=1}^{N}$ trained a priori, each of which implements a specific task T. Let $\mathcal{D}_i$ denote the set of classes handled by model t_i. Without loss of generality, we assume $\mathcal{D}_i \neq \mathcal{D}_j, \forall i \neq j$. In other words, for any pair of models t_i and t_j, we assume they classify different tasks. The goal of knowledge amalgamation is to derive a compact single-head student model that can infer all tasks, in other words, to be able to simultaneously classify all the classes in $\mathcal{D} = \cup_{i=1}^{N} \mathcal{D}_i$. In other words, the knowledge amalgamation mechanism is done in the post-processing manner where all teacher models trained to a specific task are combined into a single model to perform comprehensive classification as per its teacher models. This approach provides flexibility over existing continual learning approaches because a teacher model can be independently built for a specific task. Their knowledge can later be amalgamated into a student model without loss of generalization power.

Algorithm 1: CFA

```
Input: Teacher models $\mathcal{T}_N$, Task Memory $\mathcal{M}$, Student model $\mathcal{S}$, number
       of epochs
Output: Amalgamated Model $\mathcal{S}$
1  for epoch in epochs do
2      M ← shuffle(M); # Optional
3      for sample $m$ in $\mathcal{M}$ do
4          begin Joint Representation Learning:
5              $\mathcal{L}_M = \mathcal{L}_R = 0$; # Loss initialization
6              $\hat{f}_S \leftarrow f_S = \leftarrow F_S$; # Student encoder
7              for $T_i$ in $\mathcal{T}_N$ do
8                  $\hat{f}_{Ti} \leftarrow f_{Ti} \leftarrow F_{Ti}$; # Teacher encoder
9                  $F'_{Ti} \leftarrow f_{Ti} \leftarrow \hat{f}_{Ti}$; # Teacher decoder
10                 $\mathcal{L}_M = \mathcal{L}_M + H(\hat{f}_S, \hat{f}_{T_i}) - H(\hat{f}_S)$; # Eq. 3
11                 $\mathcal{L}_R = \mathcal{L}_R + ||F'_{Ti} - F_{Ti}||_2^2$; # Eq. 4
12         begin Soft Domain Adaptation:
13             $y_T \leftarrow \mathcal{T}_N(m)$; # Stacked teachers' soft output
14             $D_{\mathrm{KL}_{\mathrm{soft}}} = H(\hat{y}_S, y_T) - H(\hat{y}_S)$; # Eq. 5
15         $\mathcal{L} = \alpha D_{\mathrm{KL}_{\mathrm{soft}}} + (1-\alpha)(\mathcal{L}_M + \mathcal{L}_R)$; # Eq. 6
16         $S_\theta = S_\theta - \lambda \nabla \mathcal{L}$; # Parameter learning
```

4 Proposed method

In this section, we introduce the proposed CFA and its details. The knowledge amalgamation element is an extension of [3, 4] and consists of two parts: a joint representational learning and a soft domain adaptation.

4.1 Joint Representation Learning

The joint representation learning scheme is depicted in Figure 1. The features of the teachers and those to be learned from the students are first transformed into a common feature space, and then two loss terms are minimized. First, a feature ensemble loss $\mathcal{L}_M$ encourages the features of the student to approximate those of the teachers in the joint space. Then a reconstruction loss $\mathcal{L}_R$ ensures the transformed features can be mapped back to the original space with minimum possible errors.

Adaptation Layer The adaptation layer aligns the output feature dimension of the teachers and students via a 1 x 1 convolution kernel [13] that generates a predefined length of output with different input sizes. Let F_S and F_{T_i} be respectively the original features of the student and teacher T_i, and f_S and f_{T_i}

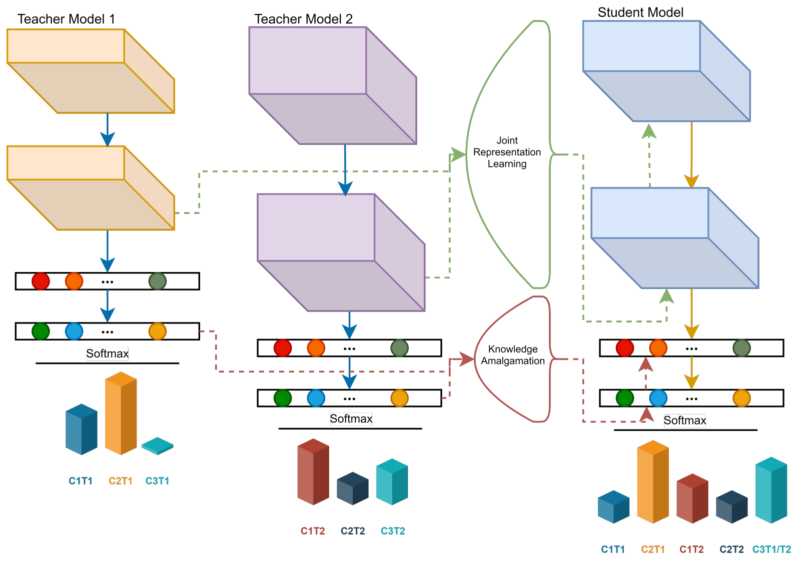

Fig. 1. A summarized workflow of the proposed CFA. It consists of two parts: A joint representation learning and knowledge amalgamation. In the joint representation learning, the features of the teachers (showing two here) and those to be learned by the students are first transformed into a joint space. Later on, knowledge amalgamation enforces a domain invariant feature space between the student and teachers via KL.

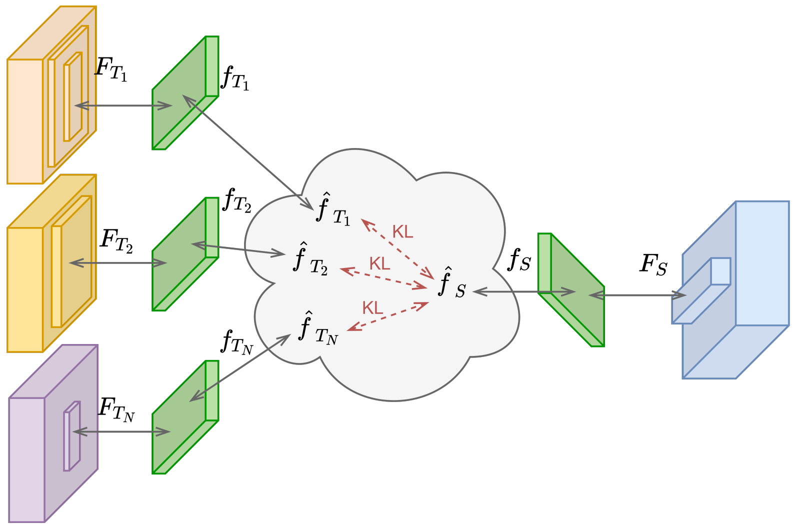

Fig. 2. An illustration of the shared extractor sub-network. A sub-network is shared between the teachers and the student during the joint representation learning procedure. This shared extractor aims to create a domain-invariant space via KL, which is then decompressed back into the student model.

their respective aligned features. In our implementation, f_S and f_{T_i} have the same size of F_S and F_{T_i}.

Shared Extractor Once the aligned features are derived, a naive approach would be to directly average the features of the teachers f_{T_i} as that of the student f_S. However, due to domain discrepancy of the training data and architectural differences of the teacher networks, the roughly aligned features may remain heterogeneous. To this end, the teachers and students share the parameters of a small learnable sub-network, illustrated in Figure 2. This shared extractor consists of three consecutive residual blocks of 1 stride. It converts f_{T_i} and f_S into the common representation spaces $\hat{f}_{T_i}$ and $\hat{f}_S$. In our implementation, $\hat{f}_{T_i}$ and $\hat{f}_S$ is half the size of f_{T_i} and f_S.

Knowledge Amalgamation To amalgamate knowledge from heterogeneous teachers, we enforce a domain invariant feature space between the student and teachers via the KL divergence, computed as follows:

$$D_{KL_i}(\hat{f}_S||\hat{f}_{T_i}) = H(\hat{f}_S, \hat{f}_{T_i}) - H(\hat{f}_S), \tag{2}$$

where $H(\hat{f}_S, \hat{f}_{T_i})$ is the cross entropy of $\hat{f}_{T_i}$ and $\hat{f}_S$ and $H(\hat{f}_S)$ is the entropy of $\hat{f}_S$.

We then aggregate all such pairwise KL losses between each teacher and the student, as shown in Figure 2, and write the overall discrepancy $\mathcal{L}_M$ in the shared space as:

$$\mathcal{L}_M = \sum_{i=1}^{N} D_{KLi}, \tag{3}$$

To further enhance the joint representation learning, we add an autoencoder [23] reconstruction loss between the original teachers' feature space. Let F'_{Ti} denote the reconstructed feature of teacher T_i, the reconstruction loss $\mathcal{L}_R$ is defined as

$$\mathcal{L}_R = \sum_{i=1}^{N} ||F'_{Ti} - F_{Ti}||_2, \tag{4}$$

4.2 Soft Domain Adaptation

Apart from learning the teacher's features, the student is also expected to produce identical or similar inferences as the teachers do. We thus also take the teachers' predictions by feeding unlabelled input samples to them and then supervise the student's training.

We assume that all teacher models handle non-overlapping classes, then directly stack their score vectors and use them as the student's target. A similar strategy can be used for teachers with overlapping classes, where the logits of repeating classes can be summed or averaged, but we do not explore it here. Instead of directly applying a cross-entropy loss between the student output and the teachers' soft output, as most knowledge distillation solutions do, we also enforce a domain invariant space into the discriminative (fully-connected) layers of the student by applying the KL between the student output and the stacked teachers' soft output.

Let y_T denote the stacked teachers' soft output and $\hat{y}_S$ denote the corresponding student soft output, then KL is applied as:

$$D_{KL_{soft}}(\hat{y}_S||y_T) = H(\hat{y}_S, y_T) - H(\hat{y}_S) \tag{5}$$

4.3 Final Loss

We incorporate the loss terms in Eqs 3, 4 and 5 into our final loss function. The whole framework is trained end-to-end by optimizing the following objective:

$$\mathcal{L} = \alpha D_{KL_{soft}}(\hat{y}_S||y_T) + (1-\alpha)(\mathcal{L}_M + \mathcal{L}_R) \tag{6}$$

where $\alpha \in [0, 1]$ is a hyper-parameter to balance the three terms of the loss function. By optimizing this loss function, the student network is trained from the amalgamation of its teachers without annotations.

5 Experiments

We evaluate CFA and its baselines under four benchmarks. Then, an ablation study gives further insight regarding CFA memory usage and internal procedures. Finally, we executed all CFA experiments using the same structure for the teachers and students; a ResNet18 backbone [25] as a feature extractor and two fully-connected layers ahead of it.

5.1 Replay Memory

To retrieve *proper*[4] logits from the teachers, CFA uses the replay memory strategy, where it records some previous samples to be replayed during the amalgamation process. The nearest-mean-of-exemplars strategy was used to build the replay memory, but any other sample selection strategy can be used.

Nearest-Mean-of-Exemplars strategy Consider $t_i(x)$ the logits of a teacher t_i on a specific task i. We compute the mean exemplar for each class in class y as $\mu_y = \frac{1}{||\mathcal{D}_i||}\sum_{x\in\mathcal{D}_i} t_i(x)$. A sample x is then added to the memory if there is free space or by descending sorting out the memory and x by their L_2 distance.

[4] Meaning, related to the original data distribution

5.2 Baselines setup

We set up two different configurations of CFA. $\text{CFA}_{\text{fixed}}$ uses the nearest-mean-of-exemplars replay memory strategy with a fixed memory footprint of 1000 samples. Meanwhile, CFA_{grow} uses the teachers' confidence replay memory strategy with a growing memory allowing 1000 samples per task. Hence, $\text{CFA}_{\text{fixed}}$ memory footprint maintains the same, independent of the number of classes learned so far[5], while CFA_{grow} memory footprint slowly grows are more classes are introduced.

All teachers and students have the same architecture, a pre-trained ResNet18 feature-extractor followed by two fully-connected layers, a $\mathbb{R}^{1000\times500}$ followed by a $\mathbb{R}^{500\times\text{output}}$. CFA is optimized under Adam with learning rate $\lambda = 10^{-4}$, hyper-parameter $\alpha = 0.5$, and 100 training epochs.

The other baselines are based on the source-code release by [28]. Their configuration is also detailed in the supplemental document. The ones which are memory-based contains a memory budget of 1000 samples per task, making them similar to CFA_{grow}.

All methods have been evaluated using the same computation environment, a Windows machine with an Intel Core i9-9900K 5.0 GHz with 32GB of main memory and an Nvidia GeForce 2080 Ti.

5.3 Metrics

The continual learning protocol is followed, where we observe three metrics:

$$\textbf{Average Accuracy: } \text{ACC} = \frac{1}{T}\sum_{i=1}^{T} R_{T,i} \tag{7}$$

$$\textbf{Backward Transfer: } \text{BWT} = \frac{1}{T-1}\sum_{i=1}^{T-1} R_{T,i} - R_{i,i} \tag{8}$$

$$\textbf{Forward Transfer: } \text{FWT} = \frac{1}{T-1}\sum_{i=2}^{T} R_{i-1,i} - \bar{b}_i \tag{9}$$

where $R \in \mathbf{R}^{TxT}$ is a test classification matrix, where $R_{i,j}$ represents the test accuracy in task t_j after completely learn t_i. The details are given by [21].

5.4 Benchmarks

SplitMNIST is a standard continual learning benchmark that adapts the entire MNIST problem [15] into five sequential tasks, with a total of 10 classes.

SplitCIFAR10 features the incremental class problem where the full CIFAR10 problem [14] is divided into five sequential tasks, with a total of 10 classes.

[5] Storage of the original teacher models parameters is still required, usually in secondary memory, as HDD or SSD.

Table 1. Numerical results over five execution runs.

Baseline	Metric (%)	*SplitMNIST*	*SplitCIFAR10*	*SplitCIFAR100*	*SplitTinyImageNet*
EWCo [16, 32]	ACC	19.13 ± 0.02	18.57 ± 2.04	7.91 ± 0.79	7.39 ± 0.03
	BWT	-97.37 ± 0.26	-88.51 ± 5.18	-83.80 ± 1.56	-71.79 ± 0.62
	FWT	-13.36 ± 1.38	-10.09 ± 5.64	-1.02 ± 0.15	-0.44 ± 0.16
LWF [6]	ACC	19.20 ± 0.06	16.27 ± 4.55	9.13 ± 0.43	0.41 ± 0.20
	BWT	-96.52 ± 0.94	-89.24 ± 9.60	-83.34 ± 5.57	-50.33 ± 3.25
	FWT	-11.92 ± 2.01	-11.05 ± 1.88	0.16 ± 0.71	-0.25 ± 0.03
ER [30]	ACC	23.41 ± 0.60	69.07 ± 3.31	27.41 ± 2.94	12.33 ± 1.23
	BWT	-93.83 ± 0.92	-24.15 ± 14.17	-66.35 ± 1.22	-71.07 ± 2.35
	FWT	-8.83 ± 3.14	-11.84 ± 0.40	-0.98 ± 0.08	-0.5 ± 0.05
AGEM [22]	ACC	9.19 ± 0.65	13.49 ± 4.12	0.94 ± 0.32	1.55 ± 0.32
	BWT	-40.52 ± 46.02	-47.26 ± -47.26	2.99 ± 5.74	-14.93 ± 2.12
	FWT	-8.97 ± 3.54	-6.06 ± -6.06	0.74 ± 1.74	-0.55 ± 0.20
DER [28]	ACC	60.85 ± 2.87	72.99 ± 6.43	**32.60 ± 9.77**	23.62 ± 3.30
	BWT	-42.80 ± 3.81	-22.71 ± 5.00	-44.64 ± 8.54	-52.19 ± 3.66
	FWT	-12.25 ± 2.71	-9.36 ± 8.93	-0.93 ± 0.09	-0.46 ± 2.12
DER++ [28]	ACC	72.86 ± 0.95	**77.86 ± 7.59**	**38.82 ± 8.28**	23.94 ± 2.52
	BWT	-24.64 ± 1.21	-16.27 ± 5.71	-49.03 ± 7.60	-43.82 ± 5.95
	FWT	-12.59 ± 0.48	-6.26 ± 8.81	-0.91 ± 0.07	-0.26 ± 2.16
FDR [29]	ACC	78.08 ± 3.41	48.00 ± 5.36	**32.26 ± 5.51**	13.30 ± 1.64
	BWT	-21.73 ± 4.36	-86.58 ± 4.37	-62.87 ± 5.83	-67.08 ± 1.69
	FWT	-10.10 ± 1.13	-11.41 ± 2.95	-0.87 ± 7.29	-0.67 ± 0.22
GSS [27]	ACC	24.69 ± 0.80	43.96 ± 2.86	13.94 ± 0.30	9.60 ± 0.84
	BWT	-91.69 ± 1.04	-55.71 ± 2.57	-78.21 ± 0.32	-69.36 ± 0.28
	FWT	-10.31 ± 1.96	-10.56 ± 3.30	-0.39 ± 0.55	-0.53 ± 0.05
HAL [26]	ACC	**88.25 ± 0.46**	50.11 ± 1.18	11.00 ± 2.87	3.23 ± 0.11
	BWT	-13.61 ± 0.62	-47.01 ± 2.14	-44.74 ± 1.84	-32.68 ± 4.10
	FWT	-8.81 ± 3.28	-11.70 ± 1.69	-0.97 ± 0.26	-0.24 ± 0.31
CFA$_{\text{fixed}}$ (Ours)	ACC	83.51 ± 1.35	74.96 ± 0.46	27.76 ± 2.28	23.44 ± 2.55
	BWT	-7.95 ± 1.53	-14.25 ± 1.76	-16.41 ± 1.49	-17.58 ± 1.89
	FWT	69.46 ± 9.41	54.28 ± 6.57	26.91 ± 3.17	20.49 ± 5.50
CFA$_{\text{grow}}$ (Ours)	ACC	**89.25 ± 3.66**	**79.40 ± 1.15**	**38.74 ± 3.26**	**32.50 ± 3.35**
	BWT	69.77 ± 1.31	49.00 ± 2.78	11.49 ± 2.65	23.33 ± 3.45
	FWT	91.77 ± 5.18	65.67 ± 8.22	21.84 ± 4.26	32.58 ± 5.12

SplitCIFAR100 features the incremental class problem where the complete CIFAR100 problem [14] is divided into 10 sequential subsets, totalling 100 classes.

SplitTinyImageNet features the incremental class problem where 200 classes from the full ImageNet [31] are resized to 64x64 colored pixels and divided into 10 sequential tasks.

5.5 Numerical results

We compare $\text{CFA}_{\text{fixed}}$ and CFA_{grow} against one regularization-based approaches (EWCo), one knowledge distillation approaches (LWF[6]), and six memory-based approaches (AGEM, DER, DER++, FDR, GSS, HAL).

Table 1 presents a metric summary between the chosen baselines and benchmarks. It demonstrates that $\text{CFA}_{\text{fixed}}$ and CFA_{grow} are comparable, or even stronger, in comparison with the current state-of-the art methods, specially when we take in consideration that $\text{CFA}_{\text{fixed}}$ presents a fixed memory footprint. Furthermore, both $\text{CFA}_{\text{fixed}}$ and CFA_{grow} have great BWT and FWT metrics, with CFA_{grow} being the only model providing positive values to all metrics. This means that CFA signalizes some zero-shot learning [21], although not explicitly focused here.

Furthermore, the most outstanding achievement of CFA is being able to achieve good continual learning performance when applied to an offline environment while maintaining competitive results. In other words, all other methods are fully continual learning procedures, which would require an organization to shift its entire learning pipeline from scratch. In contrast, CFA leverages the power of individual teachers trained on the tasks, be it in an online or offline environment. This scenario is expected in current organization pipelines, saving costs in an inevitable paradigm shift from offline to online learning agent technologies.

5.6 Ablation study

Memory Analysis Table 2 put the strongest baselines face to face to compare how their accuracies change over different memory budgets. $\text{CFA}_{\text{fixed}}$ maintains a competitive performance, even though it presents a fixed memory footprint. So, even though it has a performance similar to DER, DER++, and FDR, it benefits from using less memory and being applied to current offline learning pipelines.

Joint representation learning analysis As shown in Table 3, Joint Representation Learning (JTL) is the main adaptation driver, responsible for driving the student's latent space to represent different tasks. Furthermore, as we are using the same architecture for the teachers and students, the difference between $\alpha = 1.0$ and $\alpha = 0.5$ is not that significant here but immensely important when dealing with entire heterogeneous structures, as noted by [3]. When JTL

[6] A multi-class implementation was put forward to deal with class-incremental learning, as in [28]

Table 2. ACC(%) metrics over different budget memories.

Benchmark	Memory budget	$\text{CFA}_{\text{fixed}}$	CFA_{grow}	DER	DER++	FDR
Split CIFAR10	100	48.87 ± 5.26	**61.36 ± 2.02**	46.77 ± 3.12	51.91 ± 4.21	39.60 ± 4.54
	200	61.20 ± 4.35	**69.33 ± 1.54**	58.41 ± 3.23	64.92 ± 6.15	44.49 ± 4.31
	500	69.53 ± 3.30	**74.63 ± 1.20**	65.63 ± 5.95	**72.45 ± 6.85**	48.20 ± 5.30
	1000	74.96 ± 0.46	**79.40 ± 1.15**	72.99 ± 6.43	**77.86 ± 7.59**	41.91 ± 6.42
	2000	76.45 ± 1.90	**82.21 ± 2.52**	73.81 ± 5.12	77.44 ± 8.90	47.39 ± 7.01
Split CIFAR100	100	7.75 ± 1.20	**21.34 ± 2.30**	13.23 ± 0.00	**22.88 ± 4.90**	12.23 ± 4.50
	200	12.87 ± 2.12	**26.01 ± 2.53**	19.98 ± 0.00	23.78 ± 5.20	14.74 ± 2.24
	500	21.23 ± 2.23	**30.59 ± 3.54**	26.53 ± 5.23	**31.45 ± 6.43**	22.26 ± 4.21
	1000	27.76 ± 2.28	**38.74 ± 3.26**	32.60 ± 9.77	**38.82 ± 8.28**	32.26 ± 5.51
	2000	41.50 ± 3.45	**47.54 ± 3.18**	36.78 ± 9.88	43.45 ± 8.54	33.12 ± 5.40

Table 3. ACC(%) results with varying hyper-parameter α of the $\text{CFA}_{\text{fixed}}$ with 1000 of memory budget, controlling the influence of the Joint Representation Learning (JTL) and Soft Domain Adaptation (SDA) into its main loss.

Description	*Split CIFAR10*
$\alpha = 1.0$ | JTL(×)SDA(✓)	35.78 ± 10.78
$\alpha = 0.5$ | JTL(✓)SDA(✓)	74.96 ± 0.46
$\alpha = 0.0$ | JTL(✓)SDA(×)	69.68 ± 2.23

is disabled, the model has difficulties learning high-feature representations only with the soft domain adaptation (SDA), resulting in a tremendous catastrophic forgetting.

6 Conclusion

This paper proposes CFA, an approach to handle catastrophic forgetting for the class-incremental environment with knowledge amalgamation. CFA can amalgamate the knowledge of multiple heterogeneous trained teacher models, each for a previous task, into a single-headed student model capable of handling all tasks altogether.

We compared CFA with a set of competitive baselines under the class-incremental learning scenario, yielding positive generalization with excellent average accuracy and knowledge transfer capabilities, backed by backward and forward knowledge transfer metrics. At the same time, CFA demonstrated some zero-shot learning aptitude and handled an enormous number of classes simultaneously.

CFA presents a novel approach towards continual learning using knowledge amalgamation, enabling easy integration to current learning pipelines, enabling the shift from offline to online learning with a performance similar to or superior to the best of the only-online existing methods. Our approach is perceived as a post-processing approach of continual learning, distinguishing itself from existing approaches. Our future work is directed to explore the continual learning problem in multi-stream environments.

7 Acknowledgement

This work is financially supported by National Research Foundation, Republic of Singapore under IAFPP in the AME domain (contract no.: A19C1A0018).

References

1. Gama, J.: Knowledge discovery from data streams. CRC Press, Boca Raton, Fl. (2010)
2. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. on Knowl. and Data Eng. 22(10), 1345–1359 (Oct 2010)
3. Luo, S., Wang, X., Fang, G., Hu, Y., Tao, D., Song, M.: Knowledge amalgamation from heterogeneous networks by common feature learning. In: Kraus, S. (ed.) Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019. pp. 3087–3093. ijcai.org (2019). https://doi.org/10.24963/ijcai.2019/428, https://doi.org/10.24963/ijcai.2019/428
4. Shen, C., Wang, X., Song, J., Sun, L., Song, M.: Amalgamating knowledge towards comprehensive classification. Proceedings of the AAAI Conference on Artificial Intelligence 33(01), 3068–3075 (Jul 2019). https://doi.org/10.1609/aaai.v33i01.33013068, https://ojs.aaai.org/index.php/AAAI/article/view/4165
5. Lee, S.W., Kim, J.H., Jun, J., Ha, J.W., Zhang, B.T.: Overcoming catastrophic forgetting by incremental moment matching. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. p. 4655–4665. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (2017)
6. Li, Z., Hoiem, D.: Learning without forgetting. IEEE Transactions on Pattern Analysis and Machine Intelligence 40(12), 2935–2947 (2018). https://doi.org/10.1109/TPAMI.2017.2773081
7. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T.: Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences 114(13), 3521–3526 (2017). https://doi.org/10.1073/pnas.1611835114, https://www.pnas.org/content/114/13/3521
8. Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R.: Progressive neural networks. ArXiv abs/1606.04671 (2016)
9. Hinton, G., Dean, J., Vinyals, O.: Distilling the knowledge in a neural network. pp. 1–9 (03 2014)

10. , Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Mach. Learn. 79(1–2), 151–175 (May 2010)
11. Gretton, A., Borgwardt, K.M., Rasch, M.J., Schölkopf, B., Smola, A.: A kernel two-sample test. Journal of Machine Learning Research 13(25), 723–773 (2012), http://jmlr.org/papers/v13/gretton12a.html
12. Joyce, J.M.: Kullback-Leibler Divergence, pp. 720–722. Springer Berlin Heidelberg, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-04898-2327, https://doi.org/10.1007/978-3-642-04898-2327
13. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1–9 (2015). https://doi.org/10.1109/CVPR.2015.7298594
14. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto (2009)
15. LeCun, Y., Cortes, C.: MNIST handwritten digit database. ATT Labs (2010)
16. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G.,Rusu, A.A., Milan, K., Quan, J., Ramalho, T.: Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences 114(13), 3521–3526 (2017). https://doi.org/10.1073/pnas.1611835114, https://www.pnas.org/content/114/13/3521
17. Masse, N.Y., Grant, G.D., Freedman, D.J.: Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. Proceedings of the National Academy of Sciences 115(44), E10467–E10475 (2018). https://doi.org/10.1073/pnas.1803839115, https://www.pnas.org/content/115/44/E10467
18. Zenke, F., Poole, B., Ganguli, S.: Continual learning through synaptic intelligence. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 3987–3995. PMLR (06–11 Aug 2017), https://proceedings.mlr.press/v70/zenke17a.html
19. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/file/0efbe98067c6c73dba1250d2beaa81f9-Paper.pdf
20. van de Ven, G.M., Tolias, A.S.: Three scenarios for continual learning. ArXiv abs/1904.07734 (2019)
21. Lopez-Paz, D., Ranzato, M.A.: Gradient episodic memory for continual learning. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/file/f87522788a2be2d171666752f97ddebb-Paper.pdf
22. Chaudhry, A., Ranzato, M., Rohrbach, M., Elhoseiny, M.: Efficient lifelong learning with a-gem. ArXiv abs/1812.00420 (2019)
23. Rumelhart, D.E., McClelland, J.L.: Learning Internal Representations by Error Propagation, pp. 318–362 (1987)
24. Shen, C., Wang, X., Song, J., Sun, L., Song, M.: Amalgamating knowledge towards comprehensive classification. ArXiv abs/1811.02796 (2019)

25. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
26. Chaudhry, A., Gordo, A., Dokania, P.K., Torr, P., Lopez-Paz, D.: Using hindsight to anchor past knowledge in continual learning (2020). https://doi.org/10.48550/ARXIV.2002.08165, https://arxiv.org/abs/2002.08165
27. Aljundi, R., Lin, M., Goujaud, B., Bengio, Y.: Gradient Based Sample Selection for Online Continual Learning. Curran Associates Inc., Red Hook, NY, USA (2019)
28. Buzzega, P., Boschini, M., Porrello, A., Abati, D., Calderara, S.: Dark experience for general continual learning: a strong, simple baseline. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 15920–15930. Curran Associates, Inc. (2020)
29. , Benjamin, A.S., Rolnick, D., Kording, K.: Measuring and regularizing networks in function space (2018). https://doi.org/10.48550/ARXIV.1805.08289, https://arxiv.org/abs/1805.08289
30. Ratcliff, R.: Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. Psychological review 97 2, 285–308 (1990)
31. Tavanaei, A.: Embedded encoder-decoder in convolutional networks towards explainable ai. ArXiv abs/2007.06712 (2020)
32. Schwarz, J., Czarnecki, W., Luketina, J., Grabska-Barwinska, A., Teh, Y.W., Pascanu, R., Hadsell, R.: Progress amp; compress: A scalable framework for continual learning. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 4528–4537. PMLR (10–15 Jul 2018), https://proceedings.mlr.press/v80/schwarz18a.html