Scientific Research Publishing

International Journal of Intelligence Science, 2023, 13, 33-47
https://www.scirp.org/journal/ijis
ISSN Online: 2163-0356
ISSN Print: 2163-0283

Incremental Learning Based on Data Translation and Knowledge Distillation

Tan Cheng*, Jielong Wang

Xiamen Institute of Data Intelligence, Xiamen, China
Email: *chengtan@ict.ac.cn

How to cite this paper: Cheng, T. and Wang, J.L. (2023) Incremental Learning Based on Data Translation and Knowledge Distillation. *International Journal of Intelligence Science*, **13**, 33-47.
https://doi.org/10.4236/ijis.2023.132003

Received: March 1, 2023
Accepted: April 18, 2023
Published: April 21, 2023

Abstract

Recently, deep convolutional neural networks (DCNNs) have achieved remarkable results in image classification tasks. Despite convolutional networks' great successes, their training process relies on a large amount of data prepared in advance, which is often challenging in real-world applications, such as streaming data and concept drift. For this reason, incremental learning (continual learning) has attracted increasing attention from scholars. However, incremental learning is associated with the challenge of catastrophic forgetting: the performance on previous tasks drastically degrades after learning a new task. In this paper, we propose a new strategy to alleviate catastrophic forgetting when neural networks are trained in continual domains. Specifically, two components are applied: data translation based on transfer learning and knowledge distillation. The former translates a portion of new data to reconstruct the partial data distribution of the old domain. The latter uses an old model as a teacher to guide a new model. The experimental results on three datasets have shown that our work can effectively alleviate catastrophic forgetting by a combination of the two methods aforementioned.

Keywords

Incremental Domain Learning, Data Translation, Knowledge Distillation, Catastrophic Forgetting

1. Introduction

Unlike traditional offline learning that trains model once on a whole dataset, incremental learning is a learning paradigm that allows a model to be continually updated on a series of incremental data. Offline learning requires retaining historical data and training it along with newly acquired data. This approach usual-

ly requires huge storage space and high re-training time consumption. Incremental learning is only trained on those new data, which is of great significance compared to offline learning in real-world applications.

According to [1] [2], incremental learning consists of three scenarios: task-incremental learning (Task-IL), class-incremental learning (Class-IL) and domain-incremental learning (Domain-IL). Input distribution is changed with the increasing tasks in all three scenarios. All of them aim to continually train a model on new data to solve all tasks seen so far. On the other hand, they differ in serval ways. Label distribution remains unchanged in Domain-IL and varies in Task-IL and Class-IL with the increasing tasks. The main difference between Task-IL and Class-IL is that models are informed about which task needs to be performed in Task-IL while Class-IL does not [2]. In addition, Task-IL typically uses a "multi-headed" output layer in its network architecture, whereas Class-IL employs a "single-headed" network. **Figure 1** illustrates in detail the differences and similarities among the three scenarios.

Incremental learning, however, faces a serious challenge—catastrophic forgetting [3]. To resolve this problem, we propose a novel framework for learning a unified classifier under the domain-incremental setting. It incorporates two components to mitigate catastrophic forgetting: 1) data translation based on transfer learning, which translates new data into the data that can be correctly recognized by the old model and reconstructs the partial distribution of previous-task data. 2) Knowledge distillation, which is an effective way to preserve the knowledge learned in previous phases. Combining the two components can effectively balance the performance of old and new tasks. We systematically compare different

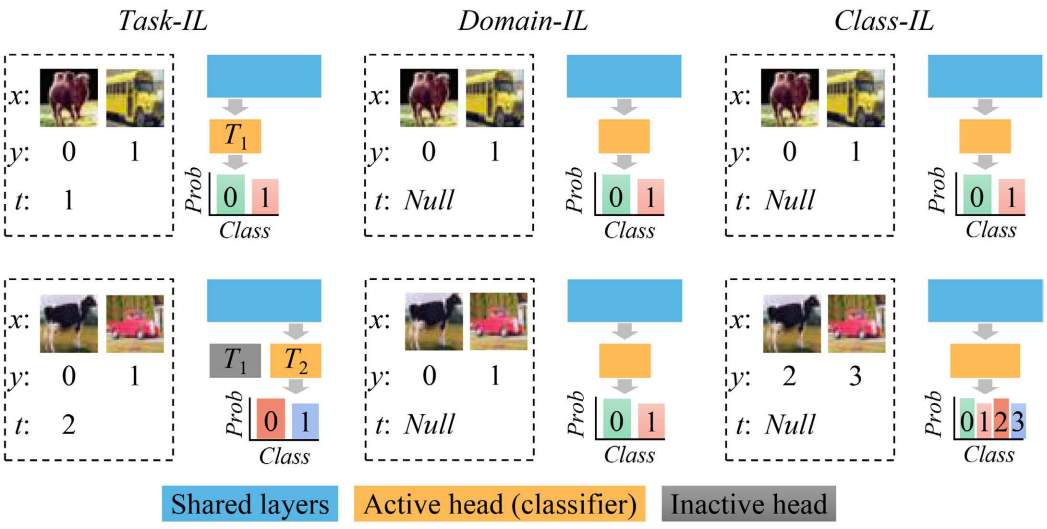

Figure 1. Here is an initial task $\mathcal{T}_1$ ("camel - bus") and an incremental task $\mathcal{T}_2$ ("cattle - pickup truck"). In each subplot, the left side (x, y, t) shows the input (image, label, task-id) of the model, and the right side indicates the network architecture and output. The output layer shows the difference among the three incremental scenarios. In Domain-IL, the classification task is "animals-vehicles", and the label distribution remains steady. In Class-IL, both new and old class labels are in the same output space (single-headed). In Task-IL, the output spaces are disjoint between tasks (multi-headed).

methods such as EWC [4], SI [5], MAS [6] on three incremental datasets consisting of multiple domains. The experimental results strongly prove the efficacy of our work.

2. Related Work

In recent years, a number of methods have been proposed to tackle catastrophic forgetting problem in the field of incremental learning. They are summarized in a slice of work [7] [8] [9] [10]. In this section, we briefly discuss these methods.

Regularization-based methods. The approaches of this strategy such as *EWC*, *SI*, *MAS* are also known as parameter-based methods. Their main idea is to identify and protect the weights in the old model that play a key role in the testing phase, typically by adding additional regularization terms to the loss function. They mainly differ in how assessing the weight importance: *EWC* estimates the weight importance through the diagonal value of Fisher information matrix; *SI* evaluates by calculating how sensitive the loss function is to parameters; *MAS* uses the sensitivity of network output to estimate the weight importance. However, it is difficult to design reasonable weight evaluation metrics during incremental processes [11]. In addition, such methods often require manually adjusting the hyper-parameter for regularization term loss.

Distillation-based methods. Knowledge distillation has been widely used in incremental learning areas, such as *UCIR* [11], *CCIL* [12], *WA* and *PODNet* [13]. In these methods, old model acts teacher of incremental model. *UCIR* calculates distillation loss from embedding, expecting the feature output of incremental model to be consistent with old models. *CCIL* demonstrates that a combination of cross-entropy loss and distillation loss that balances intra-task and inter-task learning can resolve catastrophic forgetting. WA utilizes knowledge distillation and weight aligning in classifier layer to maintain the discrimination and fairness in incremental learning. *PODNet* proposes the pooled outputs distillation loss which is a set of constraints over the output of each intermediate convolutional layer to prevent from forgetting. However, since the teacher model has only old task knowledge, these methods usually need to store some previous data to compute distillation loss.

Parameter isolation methods. This family consists of fixed network methods and dynamic architectures [7]. In fixed network methods, old task parts are usually masked out during training new tasks. For example, *PathNet* [14] imposes at parameters level and *HAT* [15] imposes at unit level. Dynamic architectures methods learn new tasks by increasing the capacity of network. And typical methods are *PNN* [16], *RCL* [17] and *DER* [18]. *PNN* first fixes the network structure and parameters corresponding to the old task, and for incremental tasks, it increases the network width and connects the output of the old network to the new network. Unlike *PNN*'s increased fixed network width, *RCL* expands by reinforcement learning. *DER* freezes learned representation to retain old knowledge and dynamically expands new feature extractors to learn new tasks. Obviously, their networks grow with increasing tasks.

Meta-learning based methods. Jathushan Rajasegaran *et al.* [19] claim that meta-learning is suited for incremental learning since tasks are progressively introduced. And they propose *iTAML* which stores some previous data to learn a generic model by meta-updates for incremental learning. At inference time, the task is predicted using generalized model parameters first, and then these generalized parameters are updated to the task-specific parameters to predict classes belonging to the respective task. This process is time-consuming [19]. *Meta-DR* [13] combines meta-learning with domain randomization to ease adaptation to new tasks. *Meta-DR* needs to have access to data from both the old and new domains in order to align the distributions [20].

Compared to these works, our work is of the third flavor: unlike regularization-based methods protect the old model's parameters, we completely free the parameters in training. Our approach neither preserves previous data to compute distillation loss, nor expands the network as task increases. Most of previous work focused on Class-IL [4] [5] [6] [18] [19] [21] and Task-IL [4] [5] [6] [14] [15] [16] [17] studies, while ours aims to alleviate catastrophic forgetting in Domain-IL [4] [5] [6] [13], such as the concept drift in reality. The incremental data with extremely different feature styles, such as DomainNet [22] widely used in domain adaptation [23], is not considered in this work since their underlying features are hardly reused by network in incremental learning. This is the limitation of our approach.

3. Our Approach

3.1. Preliminaries: Domain-IL

Formally, define $\mathcal{T}=\{\mathcal{T}_1,\mathcal{T}_2,\cdots,\mathcal{T}_n\}$ as a series of continual N-classification tasks. And $\mathcal{D}=\{\mathcal{D}_1,\mathcal{D}_2,\cdots,\mathcal{D}_n\}$ are the domain data for each task, where x is a data and y indicates its label. At the t-th stage, the goal of Domain-IL is training a model $\mathcal{M}_t=\{\theta_t\}$ on data $\mathcal{D}_t$ to perform the current task $\mathcal{T}_t$ as well as previous tasks $\mathcal{T}_{1:t-1}$, *i.e.*, $\mathcal{D}_{1:t}$ are candidates at test time. Note that $\mathcal{D}_{1:t-1}$ cannot be accessed at this stage.

3.2. Knowledge Distillation and Data Translation

Knowledge distillation, as discussed in [24], transfers knowledge from one network (teacher) to another (student). In traditional training, ground truth is used as a hard-label to optimize cross-entropy loss. Differing from this, knowledge distillation uses the teacher's output distribution as a soft-label to optimize distillation loss. As shown in Equation (1), where $p(y\mid x_i;\theta_{stu})$ and $p(y\mid x_i;\theta_{tea})$ are the prediction distribution of teacher model and student model on data x_i.

$$\mathcal{L}_{\text{distill}}=\frac{1}{n}\sum_{i=1}^{n}\left[-p(y\mid x_i;\theta_{tea})\log\left(p(y\mid x_i;\theta_{stu})\right)\right] \tag{1}$$

Domain-IL is known to have characteristics that the input distribution varies and the label distribution remains unchanged with increasing tasks. Knowledge distillation is suited for Domain-IL due to its ability of knowledge transferring.

We treat an old model as a teacher to guide a new model to memorize old knowledge. Despite the old knowledge is storing in old model, the old data is inaccessible in incremental learning. A number of works [11] [12] directly stores some old data to compute distillation loss, which take additional storage. DeGAN [25] utilizes generative adversarial net (GAN) to synthesis fake old images for knowledge distillation. However, training a functional GAN requires additional training consumption. Unlike these methods, the goal of our approach is to re-memorize the input distribution of all previous tasks using just last-task model and new data without any additional space to store historical data or training a complete GAN for fake data generation. This distribution is further trained to compute distillation loss.

Suppose that two models are trained on two different domains separately. We define the data that are predicted correctly with high confidence by the both of models as public domain data and the rest of data as their own private domain data. Our approach proceeds as follow. At t-th incremental stage, $\mathcal{M}_{t-1}=\{\theta_{t-1}\}$ is a pre-trained model learned at $t-1$ stage and $\mathcal{D}_t$ is the training data for this stage. We freeze the parameters of $\mathcal{M}_{t-1}$ and make predictions for $\mathcal{D}_t$. As mentioned above, the training data $\mathcal{D}_t$ is divided into public domain data x_{pub} and private domain data x_{pri} based on whether its predicted probability of label is greater than a given threshold p or not. We translate the data from private domain to public domain (P2P) by transfer learning. **Algorithm 1** and **Figure 2**

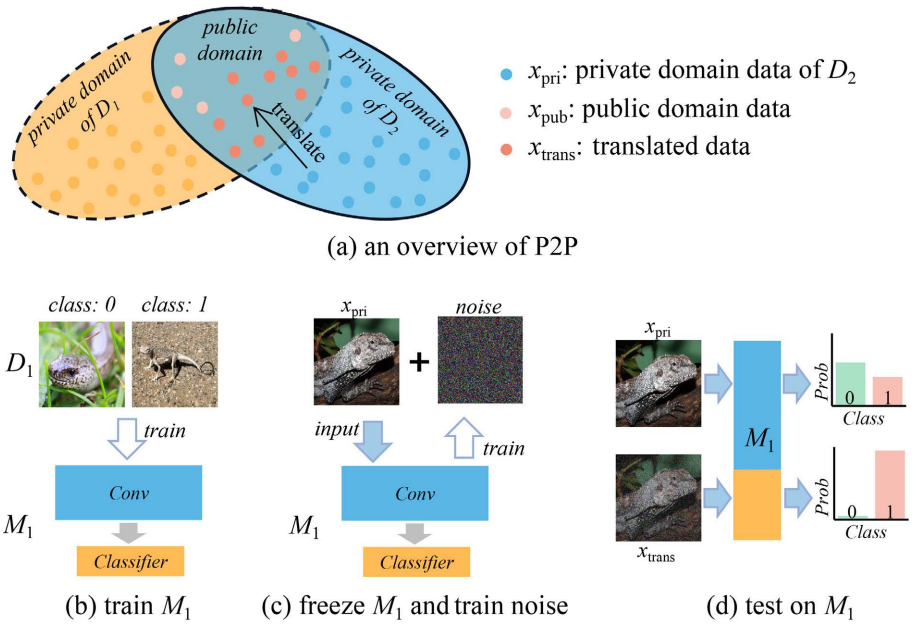

Figure 2. An overview of the proposed P2P at the $t=2$ incremental stage. P2P translates x_{pri} from the private domain of D_2 to the public domain. In subplot (a), the dotted elliptic means previous domain, while solid elliptic means current training domain. Subplot (b) represents a model M_1 trained on first task ("thunder snake - agama lizard"). Subplot (c) shows a process of P2P that translates a private data x_{pri} ("frilled lizard") from D_2. After training, the translated data x_{trans}, which is generated by adding a series of noises on x_{pri}, could be predicted correctly by M_1 with a high confidence as shown in subplot (d).

elaborate the process of P2P. x_{pri} predicted by $\mathcal{M}_{t-1}$ results in a large cross-entropy loss. P2P is a process of training x_{pri} to reduce loss, which treats x_{pri} as learnable parameters. Firstly, x_{pri} is added with a random noise δ and then fed into $\mathcal{M}_{t-1}$ to compute cross-entropy loss and gradient. The gradient will be the noise δ for the next iteration. We then repeat the above process several times. In the end, the translated data x_{trans} tends to be predicted correctly by $\mathcal{M}_{t-1}$ with a high confidence.

$$x_{trans} = \underset{x=x_{pri}+\delta}{\arg\min}\left[\mathcal{L}_{\text{cross-entropy}}\left(\theta_{t-1}; x, y\right)\right] \tag{2}$$

This training process is intuitively like "adding"/"erasing" some of the features that $\mathcal{M}_{t-1}$ considers important/unimportant to data, allowing it to be gradually predicted correctly. P2P is inspired by M2m [26], which is a rosy method for the imbalanced classification. M2m augments less-frequent classes via translating samples from more-frequent classes by adding a series of noise. There is a main difference between P2P and M2m: M2m translates between classes, while P2P translates between different domains of the same class.

We find that the translated data looks exactly like the original data from human eye and could be still predicted correctly by the model that is trained on $\mathcal{D}_t$. This means that the translated data generated by P2P does not reconstruct a complete data distribution of old domain but it enriches the diversity of public domain. Nevertheless, the translated data and the public domain data reflect the partial distribution of old domain. We apply knowledge distillation to these data to retain the information of old domain.

Algorithm 1. Framework of proposed P2P

Require: previous model $\mathcal{M}_{t-1}$, a batch dataset $D = \{x_i, y_i\}_{i=1}^{n}$, the probability threshold p, the number of iterators k, learning rate α.

Ensure: A translated dataset D_{trans}, a private domain dataset D_{pri}, a public domain dataset D_{pub}.

1. Freeze $\mathcal{M}_{t-1}$
2. Compute the predicted probability distribution for $D: P = \text{Softmax}\left(\mathcal{M}_{t-1}(x)\right)$
3. Divide $D: D_{pub} = \left\{\left(x_i, y_i\right) \mid P_{y_i} \geq p\right\}, D_{trans} = \left\{\left(x_i, y_i\right) \mid P_{y_i} < p\right\}$
4. For $x \in D_{pri}$, initialize $x^* = x + \delta$ with a small random noise δ
5. **for** $iter = 1$ to k **do**
6. $\quad \delta = \nabla_{x^*}\left[\mathcal{L}_{\text{cross-entropy}}\left(\theta_{t-1}; x^*, y\right)\right]$
7. $\quad x^* = x^* - \alpha \dfrac{1}{\|\delta\|_2}$
8. **end for**
9. $D_{trans} = \left\{\left(x^*, y\right) \mid \bar{y}\left(x^*, \theta_{t-1}\right) = y\right\}$
10. **Return** D_{trans}, D_{pri}, D_{pub}

3.3. An Overview of P2P-KD for Domain-IL

In this section, we present P2P-KD that applies knowledge distillation along

with data translation for Domain-IL. P2P-KD first uses P2P to generate a series of translation data, and then uses the old model as a teacher to guide the new model's learning, so that the new model can remember the knowledge that the old model has already learned. **Algorithm 2** demonstrates the entire process of P2P-KD in Domain-IL. The first task is learned by a standard training, e.g., empirical risk minimization with cross-entropy loss. As for the other stage t-th, $\mathcal{M}_t$ initializes with $\mathcal{M}_{t-1}$ and $\mathcal{M}_{t-1}$ teaches $\mathcal{M}_t$ during training. Distillation loss is included in these stages. It is important to note that not all training data optimizes distillation loss because the teacher model may make incorrect predictions for some private domain data, which could cause a negative influence on the student model. We compute distillation loss for public domain data and cross-entropy loss for private domain data. Public domain data consists of two parts: the data can be predicted correctly by $\mathcal{M}_{t-1}$ with high confidence (x_{pub}) and the translated data (x_{trans}). The more similar the output prediction distribution of $\mathcal{M}_t$ and $\mathcal{M}_{t-1}$ is on the public domain data, the less forgetful $\mathcal{M}_t$ is to the previous tasks. Distillation loss's constraint can prevent the incremental model from overfitting the current task. To sum up, the overall loss in one batch is as follows, where β is a hyper-parameter used to adjust the weight of distillation loss.

$$\mathcal{L} = \mathcal{L}_{\text{cross-entropy}}\left(\theta_t; \mathcal{D}_{pri}\right) + \beta \mathcal{L}_{\text{distill}}\left(\theta_t; \mathcal{D}_{pub}, \theta_{t-1}\right) \tag{3}$$

Given the strong forgetting constraint and low computational cost, we compute only distillation loss for public domain data. This is different from GD [21] and CCIL [12], in which distillation loss and cross-entropy loss are computed for the same data in their works.

Algorithm 2. An overview of P2P-KD for Domain-IL

1. $t = 2$
2. **while true do**
3. **Input:** previous model $\mathcal{M}_{t-1}$, training dataset D_t, other hyper-parameters for
4. **Algorithm 1**: p, k, α
5. **Output:** model $\mathcal{M}_t$
6. Initialize $\mathcal{M}_t = \mathcal{M}_{t-1}$
7. Freeze $\mathcal{M}_{t-1}$
8. **while true do**
9. Sample a batch dataset D from D_t
10. Get D_{trans}, D_{pri}, D_{pub} using **Algorithm 1**
11. $D_{pub} = D_{pub} \bigcup D_{trans}$
12. Train $\mathcal{M}_t$ by minimizing Equation (3)
13. **end while**
14. $t = t + 1$
15. **end while**

4. Experiments

4.1. Experimental Setup

Datasets. We conduct three incremental datasets to evaluate methods: Digit5,

Incremental-CIFAR, Incremental-Animal. All three datasets are 10-classes classified datasets consisting of multiple domains. Digit5 is sampled from five different sources: MNIST [27], MNIST-M [28], SVHN, Synthetic Digits [28] and USPS. Following [29], 25000 images are sampled from training subset and 9000 images are sampled from testing subset in MNIST, MNIST-M, SVHN, Synthetic Digits. We take the entire USPS dataset that contains only ~9300 images in total. Incremental-CIFAR and Incremental-Animal are sampled manually from CIFAR100 [30] and ImageNet [31] according the superclass rule, which have 3 and 4 domains respectively. Every class contains 500 images for training and 100 images for evaluation in Incremental-CIFAR. In Incremental-Animal, every class has ~1300 images for training and 50 images for evaluation which come from validation set because the ground truth of test data is not published. For more details about Incremental-CIFAR and Incremental-Animal, please refer to Table 1 and Table 2.

Table 1. Illustration of incremental-CIFAR.

Superclass	Task1	Task2	Task3
Aquatic mammals	beaver	otter	seal
Fish	aquarium_fish	ray	trout
Flowers	poppy	rose	tulips
Fruit	apple	orange	pear
Insects	bee	beetle	cockroach
Large carnivores	leopard	lion	tiger
Large herbivores	camel	cattle	elephant
Small mammals	hamster	mouse	squirrel
Trees	maple_tree	ak_tree	palm_tree
Vehicles	bus	pickup_trunk	tractor

Table 2. Illustration of incremental-ANIMAL.

Superclass	Task1	Task2	Task3	Task4
Birds	n01530575	n01532829	n01560419	n01582220
Geckos	n01629819	n01630670	n01631663	n01632458
Butterflies	n02276258	n02277742	n02279972	n02280649
Lizards	n01687978	n01688243	n01689811	n01692333
Snakes	n01728572	n01728920	n01729322	n01729977
Spiders	n01773157	n01773549	n01773797	n01774384
Weasels	n02441942	n02442845	n02443114	n02443484
Dogs	n02085620	n02085782	n02085936	n02086079
Wolves	n02114367	n02114548	n02114712	n02086079
Cats	n02123045	n02123159	n02123394	n02123597

Evaluation metric. Similar to GD (Lee, 2019), we report the performance by two metrics: the average incremental accuracy (ACC) and the average forgetting (FGT). Let $(x,y)\in\mathcal{D}_i^{test}$ be a test data from i-th task and $\bar{y}(x,\theta_t)$ be the prediction of t-th model, such that the following $A_{i,t}$ measures the accuracy of the t-th model at the i-th task, where $i\le t$ and I is the indication function.

$$A_{i,t}=\frac{1}{\left|D_i^{test}\right|}\sum_{(x,y)\in D_i^{test}} I\left(\bar{y}(x;\theta_t)=y\right) \tag{4}$$

Based on $A_{i,t}$, for a series of incremental task $\mathcal{T}=\{\mathcal{T}_1,\mathcal{T}_2,\cdots,\mathcal{T}_n\}$, ACC is defined as:

$$\mathrm{ACC}=\frac{1}{n-1}\sum_{t=2}^{n}A_{1:t,t} \tag{5}$$

Note that the performance of the first task is not considered, as it is not domain-incremental learning. Unlike ACC measures the overall performance directly, FGT measures the amount of forgetting on previous tasks, by averaging the performance decay:

$$\mathrm{FGT}=\frac{1}{\sum_{k=1}^{n-1}k}\sum_{t=2}^{n}\sum_{i=1}^{t-1}\max\left(A_{i,i}-A_{i,t},0\right) \tag{6}$$

Training details. All models are implemented with PyTorch and trained on one RTX-2080Ti GPU. In all experiments, we used SGD [32] with a momentum of 0.9. It is important to note that P2P proceeds online and does not require additional storage space. For the experiments on Digit5, AlexNet [33] is adopted as backbone. The learning rate starts from 0.1 and is divided by 10 after 50, 80 epochs (100 epochs in total) in the first task. We reset it to 0.001 to train incremental tasks 200 epochs. Because we found that using a larger learning rate, such as 0.1, will cause the model to be more biased towards new data, leading to forgetting of old tasks. The images are resized to 32 × 32 and randomly rotated as input. ResNet18 [34] is adopted as backbone in the experiments on Incremental-CIFAR and Incremental-Animal. For the first task and incremental tasks, the learning rate starts from 0.1 and is divided by 10 after 80, 100 and 50, 80 epochs (120 epochs and 100 epochs in total), respectively. Random cropping and horizontal flip are used to augment the training images. In addition, the images in Incremental-Animal are resized to 224 × 224. For other hyper-parameters in **Algorithm 1**, p, k, α are respectively set to 0.9, 10, 0.1 in all experiments, same as the M2m [26] setting. We find that the private domain data in three datasets can be easily translated within 10 iterators. β in Equation (3) is set to 4 and the temperature for smoothing softmax probabilities [24] is set to 2 for distillation.

4.2. Evaluation

Comparison of methods. Five methods are compared in our work: Finetune, Oracle, EWC, SI, MAS. We provide the performance of a model finetuned on new data as baseline. As an upper bound, Oracle method stores all training data of previous tasks and retrains them at every stage. Among prior works, three

state-of-the-art methods are compared: EWC, SI, MAS. For a fair comparison, we use grid search ([1e−3, 1e5]) to search hyper-parameter λ for these methods, which is an important hyper-parameter to adjust the weight of regularization term loss. In addition, the same augmentation used in P2P-KD is employed in training all the competitors. Each experiment is repeated five times with different random seeds. Furthermore, we experiment with two different task order protocols: in-order and reverse-order. For example, "MNIST → MNIST-M → SVHN → Synthetic Digits → USPS" is the in-order protocol and "USPS → Synthetic Digits → SVHN → MNIST-M → MNIST" is the reverse-order protocol for Digit5. The task order of the other two datasets is shown in Table 1 and Table 2. Besides that, we compare P2P-KD with Meta-DR briefly since Meta-DR does not open source codes. Please refer to **Appendix**.

Table 3 summarizes the results of these methods. Overall, P2P-KD outperforms significantly the non-oracle methods on Incremental-CIFAR and Incremental-Animal under two different task order protocols. In the case of Incremental-CIFAR under the in-order protocol, P2P-KD improves ACC by ~10% and FGT by ~9%. More specific results are shown in Figure 3. Higher ACC and lower FGT show the efficacy of the proposed learning scheme, which means that P2P-KD could more effectively learn new knowledge while alleviating the effects of forgetting. On Digit5 dataset, the ACC and FGT differences between all non-oracle methods are less than 2% and 3%, respectively. This is mainly due to the fact that the five domains are similar, especially in MNIST, MNIST-M and USPS. However, it is worth noting that P2P-KD still achieves the highest ACC

Table 3. Table type styles.

Dataset	Digit5		Incremental-CIFAR		Incremental-Animal	
Metric	ACC (↑)	FGT (↓)	ACC (↑)	FGT (↓)	ACC (↑)	FGT (↓)
Protocol			in-order			
Oracle	95.92 ± 0.06	0.75 ± 0.04	84.83 ± 1.45	0.78 ± 0.58	90.94 ± 0.47	0.69 ± 0.25
Finetune	90.93 ± 0.39	6.84 ± 0.46	67.97 ± 1.75 2	24.63 ± 1.70	70.71 ± 0.87	26.57 ± 1.32
EWC	91.82 ± 0.30	3.52 ± 0.23	68.87 ± 1.49	21.86 ± 1.42	74.11 ± 0.85	18.97 ± 1.00
MAS	91.50 ± 0.39	5.46 ± 0.39	69.46 ± 1.32	17.85 ± 1.16	75.78 ± 1.50	11.48 ± 1.53
SI	91.81 ± 0.38	4.56 ± 0.46	68.84 ± 1.67	20.28 ± 1.81	74.38 ± 0.76	17.05 ± 1.67
P2P-KD	92.99 ± 0.46	3.88 ± 0.66	77.48 ± 1.75	10.83 ± 1.52	81.99 ± 0.63	6.89 ± 0.66
Protocol			reverse-order			
Oracle	96.19 ± 0.05	0.60 ± 0.07	84.25 ± 0.55	0.29 ± 0.17	89.73 ± 0.39	0.36 ± 0.16
Finetune	88.21 ± 0.38	11.45 ± 0.65	69.59 ± 0.64	20.93 ± 0.91	66.37 ± 2.01	28.11 ± 1.19
EWC	90.38 ± 0.36	7.51 ± 0.33	70.02 ± 0.89	19.23 ± 1.22	67.32 ± 1.51	25.05 ± 1.10
MAS	90.00 ± 0.34	7.48 ± 0.38	70.39 ± 0.70	15.41 ± 0.98	67.84 ± 1.38	19.24 ± 1.12
SI	89.91 ± 0.21	89.71 ± 0.56	70.32 ± 1.10	16.19 ± 1.64	67.62 ± 1.68	23.39 ± 1.44
P2P-KD	89.71 ± 0.56	8.27 ± 0.72	78.40 ± 0.43	8.39 ± 0.77	77.86 ± 1.10	13.05 ± 1.73

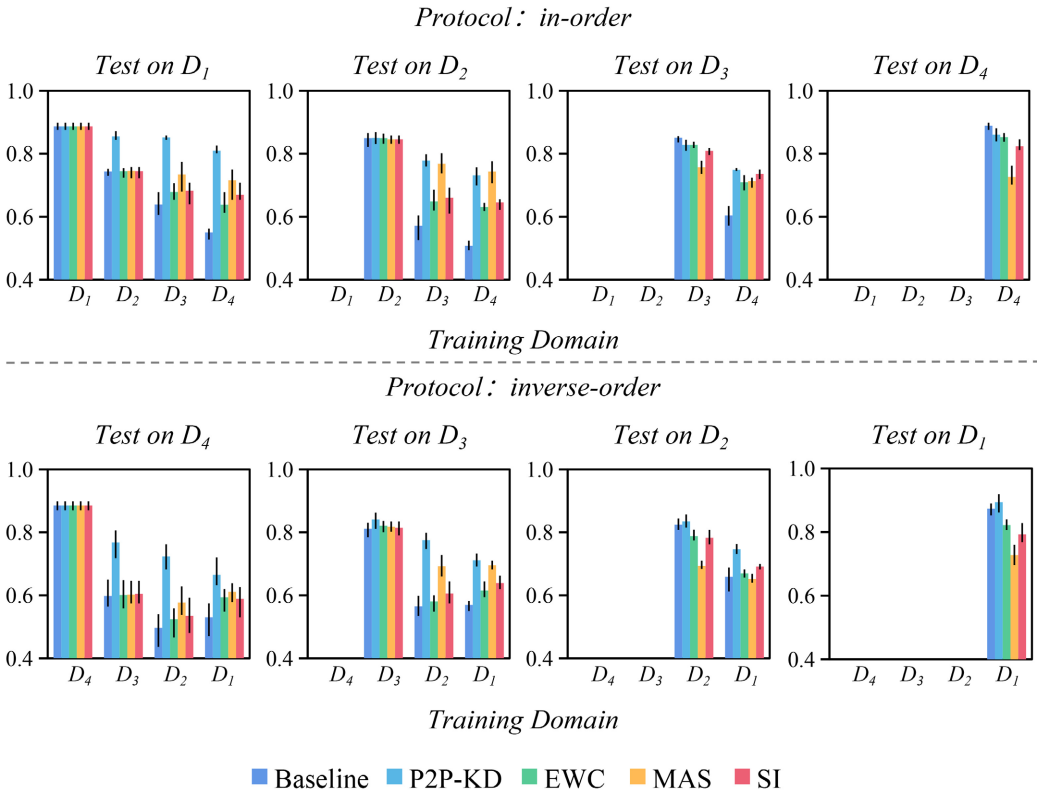

Figure 3. Incremental-Animal classification accuracy for two protocols. The subplot shows per-task performance on the test set throughout training sequence. Namely, $A_{i,t}$ defined in Equation (4).

among all the methods. This suggests that even in domains with similar characteristics, P2P-KD can still effectively leverage knowledge transfer and avoid catastrophic forgetting.

Ablation study. Our proposed method contains two components: P2P and KD. To analyze their effects, we also set up ablation experiments on three incremental datasets. First, we remove the distillation loss in our method and name it P2P-CE. Namely, both the original data and the translated data use cross-entropy loss to train. Second, we remove the data translated by P2P on P2P-KD. For a fair comparison, we oversample the data in public domain (x_{pub} defined in Section 3.2) to the same amount of P2P translated and we name it Oversampling-KD (OS-KD). Moreover, we set the same hyper-parameters to ablation methods as P2P-KD uses. We report an ablation study in Table 4. In general, compared to P2P-CE, P2P-KD greatly improves ACC by ~1%, ~7.5%, ~9% and reduces FGT by ~2%, ~15.5%, ~20% on Digit5, Incremental-CIFAR, Incremental-Animal respectively. Compared to OS-KD, P2P-KD improves ACC by ~3%, and FGT by ~6% on Incremental-Animal. They achieve similar performance on Digit5 and Incremental-CIFAR. We observe that Incremental-Animal is more complex and the public domain in Incremental-Animal is sparser than in the other two datasets. This makes it more challenging for the network to learn and adapt to the new data. However, our proposed method can increase the diversity of the training data through P2P, which improves the performance of the network on

Table 4. Ablation study.

Dataset	Digit5		Incremental-CIFAR		Incremental-Animal	
Metric	ACC (↑)	FGT (↓)	ACC (↑)	FGT (↓)	ACC (↑)	FGT (↓)
Protocol	in-order					
P2P-CE	91.81 ± 0.43	5.69 ± 0.54	69.53 ± 1.77	26.96 ± 1.23	71.77 ± 0.71	30.41 ± 0.47
OS-KD	91.87 ± 0.53	5.79 ± 0.74	76.61 ± 1.49	10.60 ± 1.23	80.97 ± 0.15	11.35 ± 1.06
P2P-KD	92.99 ± 0.46	3.88 ± 0.66	77.48 ± 1.75	10.83 ± 1.52	81.99 ± 0.63	6.89 ± 0.66
Protocol	reverse-order					
P2P-CE	88.73 ± 0.23	10.87 ± 0.35	71.13 ± 1.08	23.13 ± 1.23	69.76 ± 1.70	29.33 ± 0.69
OS-KD	90.36 ± 0.25	7.58 ± 0.34	77.08 ± 0.65	9.99 ± 0.80	74.37 ± 1.39	18.80 ± 1.17
P2P-KD	89.71 ± 0.56	8.27 ± 0.72	78.40 ± 0.43	8.39 ± 0.77	77.86 ± 1.10	13.05 ± 1.73

Incremental-Animal. The experimental results demonstrate that P2P-KD significantly outperforms P2P-CE on both ACC and FGT metrics. Moreover, in most experimental cases, it performs slightly better than OS-KD. This represents that KD plays a key role in alleviating catastrophic forgetting and P2P further boosts the performance of network.

5. Conclusion

In this paper, inspired by M2m (Kim & Jeong, 2020), we propose domain-incremental learning algorithms which combine data translation based on transfer learning with knowledge distillation into a unified training framework. The experiments demonstrate that the efficacy of the proposed algorithm, which achieves competitive performance on three main incremental datasets and outperforms other methods in most cases. The results further demonstrate the key role of knowledge distillation in mitigating catastrophic forgetting and the efficacy of data translation in augmenting the diversity of the public domain.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Hsu, Y.C., Liu, Y.C., Ramasamy, A. and Kira, Z. (2018) Re-Evaluating Continual Learning Scenarios: A Categorization and Case for Strong Baselines. Preprint. https://arxiv.org/abs/1810.12488

[2] van de Ven, G.M. and Tolias, A.S. (2019) Three Scenarios for Continual Learning. https://arxiv.org/abs/1904.07734

[3] McClosk ey, M. and Cohen, N.J. (1989) Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. *Psychology of Learning and Motivation*, **24**, 109-165. https://doi.org/10.1016/S0079-7421(08)60536-8

[4] Kirkpatrick, J., Pascanu, R., Rabinowitz, N.C., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska Barwinska, A., *et al.* (2016) Overcoming

Catastrophic Forgetting in Neural Networks. *Proceedings of the National Academy of Sciences of the United States of America*, **114**, 3521-3526. https://doi.org/10.1073/pnas.1611835114

[5] Zenke, F., Poole, B. and Ganguli, S. (2017) Continual Learning through Synaptic Intelligence. *Proceedings of the* 34*th International Conference on Machine Learning*, Sydney, 6-11 August 2017, 3987-3995.

[6] Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M. and Tuytelaars, T. (2018) Memory Aware Synapses: Learning What (Not) to Forget. *Proceedings of the Computer Vision-ECCV* 2018-15*th European Conference*, Munich, 8-14 September 2018, 144-161. https://doi.org/10.1007/978-3-030-01219-9_9

[7] Delange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T. (2021) A Continual Learning Survey: Defying Forgetting in Classification Tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **44**, 3366-3385. https://doi.org/10.1109/TPAMI.2021.3057446

[8] Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S. (2019) Continual Lifelong Learning with Neural Ntworks: A Review. *Neural Networks*, **113**, 54-71. https://doi.org/10.1016/j.neunet.2019.01.012

[9] Belouadah, E., Popescu, A. and Kanellos, I. (2020) A Comprehensive Study of Class Incremental Learning Algorithms for Visual Tasks. *Neural Networks*, **135**, 38-54. https://doi.org/10.1016/j.neunet.2020.12.003

[10] Zhao, B., Xiao, X., Gan, G., Zhang, B. and Xia, S.T. (2020) Fairness in Class Incremental Learning. *Proceedings of the* 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, WA, 13-19 June 2020, 13205-13214. https://doi.org/10.1109/CVPR42600.2020.01322

[11] Hou, S., Pan, X., Loy, C.C., Wang, Z. and Lin, D. (2019) Learning a Unified Classifier Incrementally via Rebalancing. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, 16-20 June 2019, 831-839. https://doi.org/10.1109/CVPR.2019.00092

[12] Mittal, S., Galesso, S. and Brox, T. (2021) Essentials for Class Incremental Learning. 2021 *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, Nashville, TN, 19-25 June 2021, 3508-3517. https://doi.org/10.1109/CVPRW53098.2021.00390

[13] Volpi, R., Larlus, D. and Rogez, G. (2020) Continual Adaptation of Visual Representations via Domain Randomization and Meta-Learning. 2021 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Nashville, TN, 20-25 June 2021, 4441-4451.https://doi.org/10.1109/CVPR46437.2021.00442

[14] Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A.A., Pritzel, A. and Wierstra, D. (2017) PathNet: Evolution Channels Gradient Descent in Super Neural Networks. https://arxiv.org/abs/1701.08734

[15] Serrà, J., Suris, D., Miron, M. and Karatzoglou, A. (2018) Overcoming Catastrophic Forgetting with Hard Attention to the Task. *Proceedings of the Proceedings of the* 35*th International Conference on Machine Learning*, Stockholm, 10-15 July 2018, 4555-4564.

[16] Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R. and Hadsell, R. (2016) Progressive Neural Networks. https://arxiv.org/abs/1606.04671

[17] Xu, J. and Zhu, Z. (2018) Reinforced Continual Learning. *Proceedings of the Advances in Neural Information Processing Systems*, Montréal, 3-8 December 2018, 907-916.

[18] Yan, S., Xie, J. and He, X. (2021) DER: Dynamically Expandable Representation for Class Incremental Learning. 2021 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Nashville, TN, 20-25 June 2021, 3013-3022. https://doi.org/10.1109/CVPR46437.2021.00303

[19] Rajasegaran, J., Khan, S., Hayat, M., Khan, F.S. and Shah, M. (2020) iTAML: An Incremental Task-Agnostic Meta-learning Approach. *Proceedings of the* 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, WA, 13-19 June 2020, 13585-13594. https://doi.org/10.1109/CVPR42600.2020.01360

[20] Wang, Q., Fink, O., Van Gool, L., *et al.* (2022) Continual Test-Time Domain Adaptation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, New Orleans, LA, 18-24 June 2022, 7201-7211. https://doi.org/10.1109/CVPR52688.2022.00706

[21] Lee, K., Lee, K., Shin, J. and Lee, H. (2019) Overcoming Catastrophic Forgetting With Unlabeled Data in the Wild. *Proceedings of the* 2019 *IEEE/CVF International Conference on Computer Vision*, Seoul, 27 October-2 November 2019, 312-321. https://doi.org/10.1109/ICCV.2019.00040.

[22] Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K. and Wang, B. (2019) Moment Matching for Multi-Source Domain Adaptation. *Proceedings of the* 2019 *IEEE/CVF International Conference on Computer Vision*, Seoul, 27 October-2 November 2019, 1406-1415. https://doi.org/10.1109/ICCV.2019.00149

[23] Pan, S.J., Tsang, I.W., Kwok, J.T. and Yang, Q. (2011) Domain Adaptation via Transfer Component Analysis. *IEEE Transactions on Neural Networks*, **22**, 199-210. https://doi.org/10.1109/TNN.2010.2091281

[24] Hinton, G.E., Vinyals, O. and Dean, J. (2015) Distilling the Knowledge in a Neural Network. https://arxiv.org/abs/1503.02531

[25] Addepalli, S., Nayak, G.K., Chakraborty, A. and Radhakrishnan, V.B. (2020) DeGAN: Data-Enriching GAN for Retrieving Representative Samples from a Trained Classifier. *Proceedings of the The Thirty-Fourth AAAI Conference on Artificial Intelligence*, New York, NY, 7-12 February 2020, 3130-3137.

[26] Kim, J., Jeong, J. and Shin, J. (2020) M2m: Imbalanced Classification via Major-to-Minor Translation. *Proceedings of the* 2020 *IEEE/*CVF *Conference on Computer Vision and Pattern Recognition*, Seattle, WA, 13-19 June 2020, 13893-13902. https://doi.org/10.1109/CVPR42600.2020.01391.

[27] LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998) Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, **86**, 2278-2324. https://doi.org/10.1109/5.726791

[28] Ganin, Y. and Lempitsky, V.S. (2015) Unsupervised Domain Adaptation by Backpropagation. *Proceedings of the Proceedings of the* 32*nd International Conference on Machine Learning*, Lille, 6-11 July 2015, 1180-1189.

[29] Xu, R., Chen, Z., Zuo, W., Yan, J. and Lin, L. (2018) Deep Cocktail Network: Multi-Source Unsupervised Domain Adaptation with Category Shift. *Proceedings of the* 2018 *IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, 18-22 June 2018, 3964-3973. https://doi.org/10.1109/CVPR.2018.00417

[30] Krizhevsky, A., Hinton, G., *et al.* (2009) Learning Multiple Layers of Features from Tiny Images. http://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf

[31] Deng, J., Dong, W., Socher, R., Li, L., Li, K. and Li, F. (2009) ImageNet: A Large-Scale Hierarchical Image Database. *Proceedings of the* 2009 *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami, 20-25 June 2009, 248-255. https://doi.org/10.1109/CVPR.2009.5206848

[32] Sutskever, I., Martens, J., Dahl, G.E. and Hinton, G.E. (2013) On the Importance of Initialization and Momentum in Deep Learning. *Proceedings of the* 30*th International Conference on Machine Learning*, Atlanta, GA, 16-21 June 2013, 1139-1147.

[33] Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2012) ImageNet Classification with Deep Convolutional Neural Networks. *Proceedings of the Advances in Neural Information Processing Systems* 25: 26*th Annual Conference on Neural Information Processing Systems* 2012, Nevada, 3-6 December 2012, 1106-1114.

[34] He, K., Zhang, X., Ren, S. and Sun, J. (2016) Deep Residual Learning for Image Recognition. *Proceedings of the* 2016 *IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, 27-30 June 2016, 770-778. https://doi.org/10.1109/CVPR.2016.90.

Appendix

In this section, we briefly compare our approach with Meta-DR on digit datasets. MNIST, MNIST-M, SVHN and Synthetic Digits are adopted by Meta-DR. Be consistent with it; we show the final accuracy of each task for comparison. Please refer to Table A1 and Table A2.

Table A1. Experimental setup.

	Meta-DR	P2P-KD
protoco	MNIST → MNIST-M → SYN → SVHN	MNIST → MNIST-M → SVHN → SYN
backbone	Resnet18	Alexnet
augmentation	Color, Contrast and so on (10 in total)	Rotate
training size	10000 per-task	25000 per-task
optimizer	Adam	SGD

Table A2. Comparison P2P-KD with Meta-DR.

	MNIST	MINST-M	SYN	SVHN
Meta-DR	92.0 ± 0.6	75.1 ± 0.5	95.3 ± 0.3	91.9 ± 0.2
P2P-KD	90.3 ± 0.7	97.3 ± 0.4	97.3 ± 0.1	84.7 ± 0.7