Partially Relaxed Masks for Knowledge Transfer without Forgetting in Continual Learning

Tatsuya Konishi[1], Mori Kurokawa[1], Chihiro Ono[1],
Zixuan Ke[2], Gyuhak Kim[2], and Bing Liu[2]

[1] KDDI Research, Inc.
{tt-konishi,mo-kurokawa,ono}@kddi-research.jp
[2] University of Illinois at Chicago
{zke4,gkim87,liub}@uic.edu

Abstract. The existing research on continual learning (CL) has focused mainly on preventing catastrophic forgetting. In the task-incremental learning setting of CL, several approaches have achieved excellent results, with almost no forgetting. The goal of this work is to endow such systems with the additional ability to transfer knowledge when the tasks are similar and have shared knowledge to achieve higher accuracy. Since the existing system HAT is one of most effective task-incremental learning algorithms, this paper extends HAT with the aim of both objectives, i.e., overcoming catastrophic forgetting and transferring knowledge among tasks without introducing additional mechanisms into the architecture of HAT. The current study finds that task similarity, which indicates knowledge sharing and transfer, can be computed via the clustering of task embeddings optimized by HAT. Thus, we propose a new approach, named "partially relaxed masks" (PRM), to exploit HAT's masks to not only keep some parameters from being modified in learning subsequent tasks as much as possible to prevent forgetting but also enable remaining parameters to be updated to facilitate knowledge transfer. Extensive experiments demonstrate that PRM performs competitively compared with the latest baselines while also requiring much less computation time.

Keywords: continual learning, task similarity, catastrophic forgetting

1 Introduction

Continual learning has recently received substantial attention with the increasing popularity of AI-embedded systems, but these systems still struggle to maintain performance without retraining the model from scratch, which consumes a large amount of time. The main issue in continual learning is *catastrophic forgetting*, which refers to the phenomenon in which once a model has learned a new task, its performance is likely to decline drastically on the previously learned data [10]; thus, many studies have proposed approaches that address this issue [7,20]. In particular, HAT [25] proposes a mechanism called hard attention that blocks the gradients of parameters, which are important for previous tasks to overcome

forgetting, and it achieves learning with almost no forgetting. However, consideration of only the forgetting issue is not sufficient for practical applications. There must sometimes be similar tasks that can be exploited for other tasks and dissimilar tasks that are sensitive to forgetting issues at the same time; however, conventional approaches that have focused mainly on forgetting issues have not fully considered task similarity, which can enhance performance. Therefore, another research theme has become the transfer of knowledge into a newly coming task from previous tasks where, as a matter of course, forgetting should be restrained. These challenges have been represented as *task incremental learning* (TIL), which aims at learning a mixed sequence of similar and dissimilar tasks.

Additionally, looking ahead to the realistic use of continual learning, where AI-embedded edge devices that learn continuously but do not have substantial computational resources are commonly utilized, several studies have focused mainly on the efficiency of learning [3,22]. To advance to the next stage, continual learning methods also need to be as efficient as possible. CAT [13], for instance, is the first approach for tackling a mixed sequence of similar and dissimilar tasks. CAT extends HAT by introducing attention mechanisms across similar tasks to enhance knowledge transfer; however, it is very inefficient and takes a much longer time for task similarity detection because it tries every previous task one by one to judge whether each task is worth being transferred to the current learning task. Although it succeeds in task similarity detection and outperforms HAT, it still faces enormous problems in terms of its efficiency and scalability.

To address these issues, the current study extends HAT so that it can enhance knowledge transfer without another mechanism, such as attention, and even with much shorter computation time. Our contributions to this challenge are as follows. First, the current study discovers that task similarity can be computed from task embeddings that are optimized by a HAT-like approach. Second, we propose a new approach named "partially relaxed masks" (PRM) that employs the masks that are accumulated only for dissimilar tasks so it maintains parameters that are important for the dissimilar tasks as much as possible to prevent forgetting, while keeping the remaining parameters, which are useful for the similar tasks, free to be updated for knowledge transfer. Extensive experiments demonstrate that our approach achieves equal or greater performance than state-of-the-art methods and requires much less computation time.

2 Related work

Continual learning methods are categorized into three main types: regularization-based methods [14,16,28], which add another penalty so as not to change the important parameters for previous tasks; replay-based methods [4,5,18,23], which keep the small size of previous tasks' samples and exploit them to alleviate forgetting; and parameter isolation-based [11,24,26,19] methods, which create new branches for new tasks, which are defined by new parameters. EWC [14] is one of the most popular regularization-based methods. It computes the Fisher information matrix that represents the importance of each parameter and adds

Algorithm 1 Learning procedure for PRM

Input: $x_{1:T}, y_{1:T}$, Model M with L layers
  for $t = 1 \cdots T$ **do**
    # Dissimilar task detection (DTD) phase
    state $\leftarrow$ copy(M)
    1st optimization: Freeze feature extractor, train only classifier of M with x_t, y_t
    2nd optimization: Train whole M with x_t, y_t
    Task embeddings $\left\{e_l^{1:t}\right\} \leftarrow M$
    Set of dissimilar tasks $\mathcal{D}_l^t \leftarrow$ Clustering $\left(\left\{e_l^{1:t}\right\}\right)$
    Load back: $M \leftarrow$ state
    # Learning with partially relaxed masks (LwPRM) phase
    Train whole M with $x_t, y_t, \mathcal{D}_l^t$

a regularization term that corresponds to the matrix to prevent forgetting. A-GEM [5] is a typical replay-based method, and it uses an efficient approach to select samples of previous tasks that will be learned together in a current task. A major issue with these approaches is that they require an additional memory buffer for saving past samples. To address this issue, many approaches exploit a data generator inside the model, and the generated samples are used with current learning. One of the latest parameter isolation-based methods is CCLL [26], which prevents forgetting with few additional parameters by introducing calibration modules that convert activation maps for previous tasks to a current task. Recently, several approaches have been combined with the meta-learning paradigm to select more effective samples or parameters [4,12].

Although conventional approaches have focused mainly on catastrophic forgetting, most do not have any mechanism for knowledge transfer across similar tasks, which has become another important topic in TIL. In particular, HAT [25] achieves learning with almost no forgetting by introducing hard attention to block the updating of parameters that are important for previous tasks; however, the mechanism no longer enhances knowledge transfer. CAT [13] is the first approach to deal with a mixed sequence of dissimilar tasks and similar tasks at the same time by extending HAT. CAT introduces additional attention operations into classifiers and judges similar tasks using another network separately. Although CAT achieves state-of-the-art performance in this scenario, it requires substantial computational resources for task similarity detection because it tries to build and train the reference and transfer models per previous task and check whether this transfer actually improves its performance. Therefore, the more previous tasks there are, the longer CAT takes to learn a new task.

3 Proposed PRM

The structure and procedure of our proposed method are presented in Figure 1 and Algorithm 1. The procedure is composed of the two phases: 1) dissimilar task detection (DTD) and 2) learning with partially relaxed masks (LwPRM).

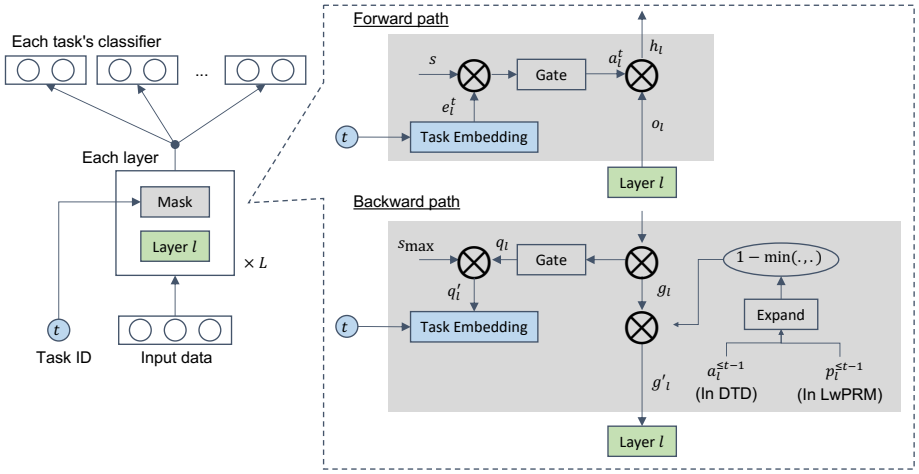

Fig. 1: Structure of PRM for learning task t. PRM follows almost the same procedure as HAT, except for part of the backward path. In dissimilar task detection (DTD), the model uses $a_l^{\leq t-1}$ to block all the parameters. In learning with partially relaxed masks (LwPRM), it uses $p_l^{\leq t-1}$ instead, which blocks only the parameters that are important for the previous dissimilar tasks. In other words, LwPRM aims to relax the masks for some parameters that are not important for previous tasks.

As the proposed method basically follows HAT, we refer to HAT first, and then explain the proposed mechanism in detail.

3.1 Mechanism of HAT

HAT requires every layer to have a task embedding, e_l^t, to control the gradient of the layer's parameters. Each layer's mask, a_l^t, is computed from $a_l^t = \sigma(se_l^t)$, and each layer's output, o_l, is replaced with $h_l = o_l \otimes a_l^t$, where σ is an activation function (e.g., sigmoid), s is a positive scaling parameter, and $\otimes$ denotes elementwise multiplication. To preserve the information obtained in previous tasks, after learning task t, HAT computes an accumulated mask, $a_l^{\leq t}$, as follows:

$$a_l^{\leq t} = \max\left(a_l^t, a_l^{\leq t-1}\right), \tag{1}$$

using elementwise maximum and the all-zero vector for $a^{\leq 0}$. In learning task $(t+1)$, the gradients of parameters, including e_l^t, are computed by a standard back-propagation, and then reduced based on the accumulated mask:

$$g'_{l,ij} = \left[1 - \min\left(a_{l,i}^{\leq t}, a_{l-1,j}^{\leq t}\right)\right] g_{l,ij}, \tag{2}$$

where unit indices i and j denote the l-th and $(l$–$1)$-th layer outputs, respectively. $g_{l,ij}$ denotes its gradient. Additionally, HAT utilizes two more tricks to stabilize

learning. First, in learning, scaling parameter s is linearly annealed as follows:

$$s = \frac{1}{s_{\max}} + \left(s_{\max} - \frac{1}{s_{\max}} \right) \frac{b-1}{B-1}, \tag{3}$$

where $s_{\max}$ is a hyper parameter, the value of which is a large positive number, and b and B denote the batch index and the total number of batches, respectively. In testing, $s_{\max}$ is used instead of s. Second, to alleviate the side effect on embedding gradient compensation, the formula below is used:

$$q'_{l,i} = \frac{s_{\max} \left[\cosh(s e^t_{l,i}) + 1 \right]}{s \left[\cosh(e^t_{l,i}) + 1 \right]} q_{l,i}, \tag{4}$$

where $q_{l,i}$ denotes the gradient that corresponds to $e^t_{l,i}$ and is replaced with $q'_{l,i}$.

3.2 Mechanism of PRM

First, the DTD phase aims to obtain the task similarity from the task embeddings that are optimized in the same way as with HAT. Once the optimized task embeddings, which are represented by $\{e^{1:t}_l\}$, where l and t denote the indices of the layer and task, respectively, are obtained, the set of dissimilar tasks, $\mathcal{D}^t_l$, can be computed via a clustering on the embeddings. The reason that we focus on task embeddings to measure task similarity is that since they are used as the basis for masking the output of each layer, if two tasks emphasize similar parameters and try to pass them without blocking (masking), their task embeddings should be similar. Since HAT utilizes accumulated masks to block the gradients of the model's parameters, the task embeddings are more flexibly updated than the model's parameters; thus, we expect the task embeddings to provide an informative representation for task similarity. Second, in the LwPRM phase, the model's parameters are optimized again using $\mathcal{D}^t_l$ with the intention of not only blocking some parameters to overcome forgetting, as with HAT, but also making remaining parameters free for updating to transfer knowledge.

Dissimilar task detection (DTD) To balance knowledge transfer and the prevention of forgetting, it is important to determine which tasks are similar and can be transferred to the current task and which tasks are dissimilar and should be blocked so that they are not forgotten. To address this issue, we focus on the task embeddings that are learned through the HAT mechanism. Since the mechanism employs an accumulated mask that reduces the gradients of the model's parameters, the task embeddings can be more easily updated than the model's parameters. Therefore, the task embeddings are expected to provide an informative representation of the tasks and their relations with one another. Specifically, we adopt an unsupervised clustering method for judging task similarity.

First, the model follows the approach of HAT in learning task t by reducing the gradients of the parameters according to (1) and (2), namely, as illustrated in Figure 1 at the bottom, $a_l^{\leq t-1}$ is used to reduce the gradients for all the previous tasks in the DTD phase. After learning task t, optimized $\left\{e_l^{1:t}\right\}$ are obtained. Using a clustering method, the set of previous tasks with embeddings that do not belong to the same cluster as task t are regarded as dissimilar tasks, which are represented by $\mathcal{D}_l^t$. Although any clustering method can be used, we exploit X-means [21] since it does not require the number of clusters as input; instead, it searches for the optimal number of clusters based on the Bayesian information criterion by applying K-means recursively.

Two stage optimization We introduce a new optimization that proceeds in two stages on the DTD phase. The model consists of two parts: the first is the feature extractor that is to be shared across tasks, and the other is the classifier that is built for each task. Therefore, we hypothesize that if both the feature extractor and classifier are optimized simultaneously, the information that represents the difference across tasks and can be used as a clue for task similarity comparison may be dispersed both into not only the task embeddings but also the classifier, which may degrade the performance of our approach. To ensure maximum sharing in the task embedding inside the feature extractor, which will facilitate similarity comparison, we first freeze the feature extractor and learn only the classifier (i.e., depicted as "1st optimization" in Algorithm 1); then, we optimize both the feature extractor and classifier simultaneously, from which task embeddings are obtained for the clustering (i.e., "2nd optimization").

Learning with partially relaxed masks (LwPRM) Although the accumulated mask, $a_l^{\leq t}$, in HAT plays a large role in preventing forgetting, it may also restrain knowledge transfer among similar tasks because the gradients of the parameters are reduced, regardless of task similarity. Thus, following HAT, we extend it so that it can promote knowledge transfer by introducing a new mechanism named "partially relaxed masks" (PRM).

PRM employs masks per previous task like HAT; however, not all masks are used to reduce the gradients of the parameters according to the task similarity. Instead, PRM accumulates only the masks that belong to previous dissimilar tasks so that it can prevent forgetting only for dissimilar tasks while maintaining opportunities for improvement for other tasks at the same time. Namely, the accumulated mask focuses only on dissimilar tasks and is partially relaxed to keep the parameters for other tasks updatable, which can enhance knowledge transfer. Given that we know which tasks are dissimilar to the current new task by clustering, (1) and (2) are replaced as follows:

$$p_l^{\leq t} = \max\left(\left\{a_l^i | i \in \mathcal{D}_l^t, i \leq t\right\}\right), \quad g'_{l,ij} = \left[1 - \min\left(p_{l,i}^{\leq t}, p_{l-1,j}^{\leq t}\right)\right] g_{l,ij} \tag{5}$$

4 Experiments

Table 1: The statistics of each task.

Dataset	# Tasks	# Classes	# Trainings	# Validations	# Tests
CIFAR100-10T	10	10	4500	500	1000
EMNIST-10T	10	5 (Last three: 4)	500	200	200
F-CelebA-10T	10	2	400	40	80
F-EMNIST-10T	10	62	1240	310	310

4.1 Datasets

We use the following four kinds of datasets to evaluate the performance in terms of both prevention of forgetting and knowledge transfer. The datasets are split into multiple tasks, and the statistics of each task are presented in Table 1.

Dissimilar tasks: CIFAR100 [15], which contains 100 classes, is split into 10 tasks, each of which has 10 classes; the dataset is named CIFAR100-10T. EMNIST [6], which contains 47 classes, is split into 10 tasks, each of which has 5 (the last three tasks have 4) classes; the dataset is named EMNIST-10T. These datasets are expected to be sensitive to forgetting as each task has different classes and there are few relations or similarities across tasks.

Similar tasks: F-CelebA [17] is a dataset that contains face images of celebrities and labels that indicate whether or not they are smiling. Different celebrities correspond to different tasks, and 10 celebrities are used in the experiments; the dataset is named F-CelebA-10T. F-EMNIST [17] is a dataset that contains 62 classes of character images handwritten by different users. We use the images that correspond to 10 writers; the dataset is named F-EMNIST-10T. These tasks are supposed to have shared knowledge across tasks as each task has the same set of labels and the data that are naturally similar.

We conduct experiments with three kinds of sequences that combine at most two different tasks in random order, as presented in Table 2, Table 3, and Table 4: **only dissimilar tasks** - #1 and #2, **only similar tasks** - #3 and #4, and **mixed of dissimilar and similar tasks** - #5 and #6.

4.2 Baselines

We compare PRM with classic and latest continual learning methods that can work as TIL systems, namely, EWC [14] in the HAT package (EHAT), ACL [8], PathNet [9] (PNT), SupSup [27] (SS), HyperNet [19] (HYP), HAT [25] and CAT [13]. Since HAT focuses only on preventing forgetting and does not have any mechanism for knowledge transfer, it is expected to perform poorly on similar tasks. To the best of our knowledge, CAT is the only approach that focuses on a mixed sequence of similar and dissimilar tasks; however, CAT takes much longer to learn. Also, we prepare two reference methods: naive continual learning (NCL) and single-task learning (STL). NCL learns a new task without considering previous tasks; thus, severe forgetting is expected to occur for dissimilar

tasks. STL learns all the tasks at once. Although it does not follow the continual learning scenario, it is expected to be the upper bound, only for dissimilar tasks.

4.3 Implementation details

The input go through two fully connected layers passing ReLU and dropout layers. The networks are optimized by minimizing the last classifier's cross-entropy loss using SGD. The learning rate starts from 0.025 and is gradually reduced until it reaches 0.001. With no improvement in the validation loss for 5 epochs, the training stops. The batch size and $s_{\max}$ are set to 64 and 400, respectively. Other hyper parameters, such as the dropout rate and the weight of regularization, are searched over 20 trials using Tree-structured Parzen Estimator (TPE) [2], which is implemented in Optuna [1]. The baselines are evaluated on the original code with modifications while aligning with our setting as much as possible.

4.4 Metrics

Accuracy (Acc): The average accuracy for all tasks after learning them, where the model is optimized with the best hyper parameters that are found in the search. **Parameter Sensitivity (PS)**: The standard deviation of "Acc" with varied hyper parameters over 20 searches. **Forward Transfer (FWT)**: The test accuracy of task i just after learning task i is compared to the accuracy for task i by STL, which is expressed as $1/T\sum_t^T(\alpha_t^t - \tilde{\alpha}_t)$, where α_i^j denotes the evaluated accuracy of task i after learning task j, $\tilde{\alpha}_i$ denotes the test accuracy for task i by STL, and T denotes the total number of tasks. **Backward Transfer (BWT)**: The average of improvement from each task's initial accuracy to the final accuracy, which is represented by $1/T\sum_t^T(\alpha_t^{\mathrm{T}} - \alpha_t^t)$ [18]. Negative values represent that forgetting occurs. **Computation Time (CT)**: The total computation time for learning all the tasks, which is measured in seconds.

4.5 Results

The results are presented in Table 2, Table 3, and Table 4. Each row presents to the average results over the same set of three random task sequences.

Only dissimilar tasks (#1 and #2): NCL causes severe forgetting (-3.4% and -5.5%, as presented in BWT). PRM achieves competitive accuracy compared to the baselines (especially PNT and CAT) without much forgetting. Also, PRM requires only one-tenth the computation time of CAT in #2. Among the baselines, PRM achieves almost the best score in the second shortest time.

Only similar tasks (#3 and #4): As the tasks are similar, NCL does not cause much forgetting, and even improves task by task, as presented in BWT. In both sequences, PRM outperforms all baselines without forgetting and even with significant backward transfer, as shown in #4. As in the case of only dissimilar tasks, which is presented in Table 2, PRM's efficiency is only behind PNT, but PNT's performance is markedly lower than PRM in these only similar tasks.

Table 2: Results for only dissimilar tasks sequences.

	(#1) EMNIST-10T				(#2) CIFAR100-10T			
	Acc	PS	CT	FWT / BWT	Acc	PS	CT	FWT / BWT
(STL)	0.929	0.6%	45	- / -	0.612	1.4%	142	- / -
NCL	0.879	0.3%	26	-1.7% / -3.4%	0.534	1.1%	123	-2.3% / -5.5%
ACL	0.902	0.5%	916	-2.6% / -0.1%	0.508	0.6%	6328	-10.3% / -0.1%
PNT	**0.910**	0.3%	43	-2.0% / 0.0%	0.571	0.5%	394	-4.1% / 0.0%
SS	0.829	11.9%	172	-5.6% / -4.4%	0.462	11.2%	2800	-10.8% / -4.2%
HYP	0.822	10.9%	1105	-10.7% / -0.1%	0.219	3.4%	10825	-38.8% / -0.5%
HAT	0.905	0.3%	101	-2.4% / 0.0%	0.582	0.8%	912	-3.0% / 0.0%
EHAT	0.899	0.3%	187	-3.1% / 0.0%	0.578	0.6%	1011	-3.4% / 0.0%
CAT	0.907	0.3%	1276	-2.2% / 0.0%	**0.587**	0.7%	6018	-2.5% / 0.0%
PRM	0.897	0.4%	80	-2.3% / -1.0%	0.582	0.9%	635	-2.8% / -0.1%

Table 3: Results for only similar tasks sequences.

	(#3) F-EMNIST-10T				(#4) F-CelebA-10T			
	Acc	PS	CT	FWT / BWT	Acc	PS	CT	FWT / BWT
(STL)	0.717	3.2%	126	- / -	0.823	0.6%	15	- / -
NCL	0.654	10.0%	48	-5.3% / -0.9%	0.820	1.2%	14	-3.7% / 3.4%
ACL	0.043	0.9%	1633	-66.4% / -0.9%	0.695	2.4%	628	-14.3% / 1.5%
PNT	0.572	15.7%	178	-14.5% / 0.0%	0.716	1.1%	32	-10.6% / 0.0%
SS	0.456	14.6%	1022	-14.3% / -11.7%	0.780	9.5%	440	-5.0% / 0.8%
HYP	0.060	1.5%	3313	-65.5% / -0.2%	0.525	1.5%	1495	-26.6% / -3.2%
HAT	0.655	3.7%	245	-6.2% / 0.0%	0.759	1.1%	70	-6.3% / 0.0%
EHAT	0.655	3.8%	497	-6.1% / 0.0%	0.769	0.8%	110	-5.3% / 0.0%
CAT	0.643	2.1%	2171	-7.4% / 0.0%	0.781	0.7%	707	-4.1% / 0.0%
PRM	**0.657**	3.3%	176	-5.7% / -0.2%	**0.796**	1.4%	54	-5.5% / 2.8%

Mixed of dissimilar and similar tasks (#5 and #6): While PRM achieves the best performance in #6, it underperforms CAT in #5. However, the performance of CAT is highly sensitive to the hyper parameters, according to the PS value of 7.2%, while PRM has 3.4% PS, which is the most stable among the baselines. Moreover, CAT requires a long computation time as shown in Figure 2, where the computation time for each task and the accumulated time for all tasks in #6 are plotted. According to these figures, CAT takes much longer than the others, and learning more tasks requires more time. Based on these observations, in practice, it is difficult to use CAT when there is a large amount of data or many tasks, and due to its unstable performance, tuning is essential. In contrast, PRM needs much less computation time, e.g., 1/35 that of CAT when learning 20 tasks, and its performance is also stable with varied hyper parameters.

Table 4: Results for mixed of dissimilar and similar tasks sequences.

	(#5) EMNIST-10T & F-EMNIST-10T				(#6) CIFAR100-10T & F-CelebA-10T			
	Acc	PS	CT	FWT / BWT	Acc	PS	CT	FWT / BWT
(STL)	0.791	1.4%	113	- / -	0.629	1.1%	254	- / -
NCL	0.728	5.0%	87	-5.6% / -0.6%	0.540	1.0%	154	-3.1% / -5.9%
ACL	0.370	9.3%	5437	-41.6% / -0.5%	0.521	0.4%	7515	-10.9% / 0.0%
PNT	0.628	4.7%	281	-16.3% / 0.0%	0.556	0.7%	425	-7.3% / 0.0%
SS	0.572	16.2%	586	-7.6% / -14.3%	0.461	10.5%	1865	-11.1% / -5.7%
HYP	0.322	6.0%	6439	-46.2% / -0.7%	0.199	1.7%	19973	-42.2% / -0.8%
HAT	0.721	3.6%	357	-7.0% / 0.0%	0.572	0.5%	891	-5.8% / 0.0%
EHAT	0.728	6.9%	854	-6.3% / 0.0%	0.572	0.6%	1801	-5.8% / 0.0%
CAT	**0.737**	7.2%	9851	-5.3% / 0.0%	0.581	0.8%	22810	-4.8% / 0.0%
PRM	0.720	3.4%	310	-6.8% / -0.3%	**0.582**	0.6%	642	-4.2% / -0.6%

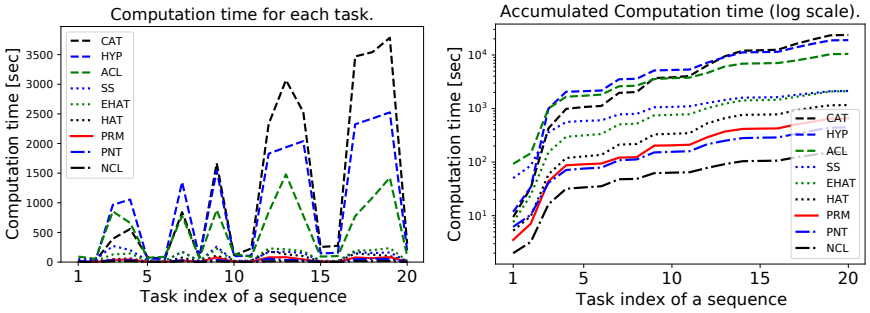

Fig. 2: Computation times for one sequence of #6. The right figure is plotted in log scale. CAT, HYP and ACL take more time, while PRM requires less time.

4.6 Ablation study

We check how much the DTD phase influences the total performance, as Lw-PRM completely depends on its behavior. The results are presented in Table 5. PRM(S) and PRM(D) indicate the cases in which all previous tasks are regarded as similar and dissimilar, respectively, instead of actual clustering. PRM(T) represents the cases in which the types of tasks are given and used as a replacement for clustering (e.g., when learning a new task of CIFAR100-10T, the model can tell which previous tasks come from CIFAR100-10T, and only these tasks are regarded as similar). When the sequence of tasks consists only of tasks from the same dataset (#1 to #4), the behavior of PRM(T) is the same as that of PRM(S). Additionally, we check the effect of the two stage optimization, as shown in the column of "w/o 2SO", where both the feature extractor and classifier are optimized simultaneously in the DTD phase.

Notably, PRM(T), where the types of tasks are given, does not always show the best performance. Instead, PRM, which employs task embeddings through clustering, can utilize not explicit but implicit relation across tasks, thereby

Table 5: Results of ablation experiments.

	PRM(S)	PRM(D)	PRM(T)	PRM	w/o 2SO
(#1) EMNIST-10T	0.882	**0.904**	-	0.897	0.897
(#2) CIFAR100-10T	0.562	0.581	-	**0.582**	0.579
(#3) F-EMNIST-10T	0.633	0.654	-	**0.657**	0.658
(#4) F-CelebA-10T	**0.825**	0.759	-	0.796	0.775
(#5) EMNIST-10T & F-EMNIST-10T	0.713	**0.731**	0.720	0.720	0.721
(#6) CIFAR100-10T & F-CelebA-10T	0.568	0.571	0.567	**0.582**	0.586

resulting in similar or better performance than PRM(T). Conversely, it is reasonable that PRM(D) has the best performance in #1 and PRM(S) performs the best in #4. Although PRM(D) performs the best in #5, the performance differences among other the types of PRM are small. Generally, PRM treats well task embeddings via clustering to handle various types of data sequences. Moreover, it is demonstrated by comparing "PRM" and "w/o 2SO" that the two stage optimization contributes PRM's performance especially in #4, which is consistent with our hypothesis that it can facilitate the similarity comparison.

4.7 Limitations

While PRM tries to open up masks for knowledge transfer and achieves better transfer in similar tasks as shown in Table 3, there is room to employ another explicit mechanism, such as attention. However, it is still unclear which combinations perform best. Additionally, as presented in Table 5, the current PRM is sometimes outperformed by PRM(S) and PRM(D), which may indicate that with a more effective method for utilizing task embeddings, it will be possible to improve its performance. The effectiveness of exploiting task embeddings is proven in most cases; however, a more effective approach needs to be developed.

5 Conclusion

To extend HAT so that it can not only overcome catastrophic forgetting but also transfer knowledge, the current study makes two contributions. First, we discover that the task embeddings optimized by parameter masking approaches, such as HAT, provide an informative representation for the task similarity. Second, we propose a new approach, namely PRM, that controls which parameters should be blocked or relaxed based on the task similarity that is obtained via clustering on the task embeddings. The experiments show that PRM achieves at least competitive performance in terms of both prevention of forgetting and knowledge transfer compared to the latest baselines with much less computation time.

References

1. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A Next-generation Hyperparameter Optimization Framework. In: Proc. of SIGKDD (2019)
2. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization (2011)
3. Borsos, Z., Mutný, M., Krause, A.: Coresets via Bilevel Optimization for Continual Learning and Streaming. In: Proc. of NeurIPS (2020)
4. Chaudhry, A., Gordo, A., Dokania, P.K., Torr, P., Lopez-Paz, D.: Using Hindsight to Anchor Past Knowledge in Continual Learning. In: Proc. of AAAI (2021)
5. Chaudhry, A., Ranzato, M., Rohrbach, M., Elhoseiny, M.: Efficient Lifelong Learning with A-GEM. In: Proc. of ICLR (2019)
6. Cohen, G., Afshar, S., Tapson, J., van Schaik, A.: EMNIST: Extending MNIST to handwritten letters. In: Proc. of IJCNN (2017)
7. Delange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T.: A Continual Learning Survey: Defying Forgetting in Classification Tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
8. Ebrahimi, S., Meier, F., Calandra, R., Darrell, T., Rohrbach, M.: Adversarial Continual Learning. In: Proc. of ECCV (2020)
9. Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A.A., Pritzel, A., Wierstra, D.: PathNet: Evolution Channels Gradient Descent in Super Neural Networks (2017)
10. Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks. In: Proc. of ICLR (2014)
11. Hu, Wenpeng and Lin, Zhou and Liu, Bing and Tao, Chongyang and Tao, Zhengwei and Ma, Jinwen and Zhao, Dongyan and Yan, Rui: Overcoming Catastrophic Forgetting for Continual Learning via Model Adaptation. In: Proc. of ICLR (2018)
12. Javed, K., White, M.: Meta-Learning Representations for Continual Learning. In: Proc. of NeurIPS (2019)
13. Ke, Z., Liu, B., Huang, X.: Continual Learning of a Mixed Sequence of Similar and Dissimilar Tasks. In: Proc. of NeurIPS (2020)
14. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., Hadsell, R.: Overcoming Catastrophic Forgetting in Neural Networks. Proc. of National Academy of Sciences (2017)
15. Krizhevsky, A., Hinton, G., et al.: Learning Multiple Layers of Features from Tiny Images. Tech. rep. (2009)
16. Li, Z., Hoiem, D.: Learning without Forgetting. In: Proc. of ECCV (2016)
17. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep Learning Face Attributes in the Wild. In: Proc. of ICCV (2015)
18. Lopez-Paz, D., Ranzato, M.: Gradient Episodic Memory for Continual Learning. In: Proc. of NeurIPS (2017)
19. von Oswald, J., Henning, C., ao Sacramento, J., Grewe, B.F.: Continual Learning with Hypernetworks. In: Proc. of ICLR (2020)
20. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual Lifelong Learning with Neural Networks: A Review. Neural Networks (2019)
21. Pelleg, D., Moore, A.W., et al.: X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In: Proc. of ICML (2000)

22. Pellegrini, L., Graffieti, G., Lomonaco, V., Maltoni, D.: Latent Replay for Real-Time Continual Learning. In: Proc. of IROS (2020)
23. Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., Tesauro, G.: Learning to Learn without Forgetting by Maximizing Transfer and Minimizing Interference. In: Proc. of ICLR (2019)
24. Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R.: Progressive Neural Networks (2016)
25. Serra, J., Suris, D., Miron, M., Karatzoglou, A.: Overcoming Catastrophic Forgetting with Hard Attention to the Task. In: Proc. of ICML (2018)
26. Singh, P., Verma, V.K., Mazumder, P., Carin, L., Rai, P.: Calibrating CNNs for Lifelong Learning. In: Proc. of NeurIPS (2020)
27. Wortsman, M., Ramanujan, V., Liu, R., Kembhavi, A., Rastegari, M., Yosinski, J., Farhadi, A.: Supermasks in Superposition. In: Proc. of NeurIPS (2020)
28. Zenke, F., Poole, B., Ganguli, S.: Continual Learning Through Synaptic Intelligence. In: Proc. of ICML (2017)