Unsupervised Model Personalization while Preserving Privacy and Scalability: An Open Problem

Matthias De Lange[1] Xu Jia[2] Sarah Parisot[2,3]
Ales Leonardis[2] Gregory Slabaugh[2] Tinne Tuytelaars[1]

[1]KU Leuven [2]Huawei, Noah's Ark Lab [3]Mila

{firstname.lastname}@kuleuven.be {firstname.lastname}@huawei.com

Abstract

This work investigates the task of unsupervised model personalization, adapted to continually evolving, unlabeled local user images. We consider the practical scenario where a high capacity server interacts with a myriad of resource-limited edge devices, imposing strong requirements on scalability and local data privacy. We aim to address this challenge within the continual learning paradigm and provide a novel Dual User-Adaptation framework (DUA) to explore the problem. This framework flexibly disentangles user-adaptation into model personalization on the server and local data regularization on the user device, with desirable properties regarding scalability and privacy constraints. First, on the server, we introduce incremental learning of task-specific expert models, subsequently aggregated using a concealed unsupervised user prior. Aggregation avoids retraining, whereas the user prior conceals sensitive raw user data, and grants unsupervised adaptation. Second, local user-adaptation incorporates a domain adaptation point of view, adapting regularizing batch normalization parameters to the user data. We explore various empirical user configurations with different priors in categories and a tenfold of transforms for MIT Indoor Scene recognition, and classify numbers in a combined MNIST and SVHN setup. Extensive experiments yield promising results for data-driven local adaptation and elicit user priors for server adaptation to depend on the model rather than user data. Hence, although user-adaptation remains a challenging open problem, the DUA framework formalizes a principled foundation for personalizing both on server and user device, while maintaining privacy and scalability.

1. Introduction

Data availability and increased hardware efficiency have made neural networks thrive in a wide range of tasks, competing human-level performance in a variety of tasks [13].

However, high performing deep neural network models lead to considerable data requirements, with high capacity models trained on large amounts of labeled data. Additionally, performance could be significantly increased by personalizing models to user-specific data. Nonetheless, user data cannot be shared directly due to rigorous privacy constraints. This motivates the need to separate supervised model training on the server from local adaptation to a user's unlabeled personal data. Furthermore, the personalized user model performing tasks locally has the additional benefit of alleviated connectivity requirements.

In this work, we explore this challenge in a class incremental learning setting, relying on the assumption that a user's personal data evolves over time. We define a pragmatic distributed setup comprising a central server connected with a large number of user devices. We assume the server to be a high-end machine endowed with extensive storage capacity and computational resources. By contrast, the compact user device has limited resources for both storage and computation. In practice, the number of users may be very high, hence imposing the need for scalable user-adaptation. A naive approach consists of training a new server model from scratch for each user, resulting in a linearly increasing demand for computational resources. Another infeasible route is to locally finetune user-models, as user devices are confined by limited computational capabilities. This paper explores a more realistic solution, training a single ensemble of models on the server, subsequently aggregated leveraging user-specific priors. Besides scalability, two additional hurdles must be tackled: the server is precluded access to local user data by pressing privacy requirements, and user data is typically unlabeled, raising the need for unsupervised adaptation.

In order to tackle these rigorous constraints, we introduce a new Dual User-Adaptation framework (DUA) for model personalization using a task incremental setup. In this setting, tasks are defined as clearly delineated batches of independent and identically distributed (i.i.d.) data, en-

abling network optimization for a task through multiple iterations over its data. The server trains on the sequence of tasks and provides the user either a general or personalized model. This model can be prone to further local adaptation on the user device. Hence, DUA disentangles user-adaptation in two phases: 1) the server exploits a model adaptation strategy, with model weight importance as a proxy for user data, and 2) the user device directly adapts to local data using domain adaptation tools.

In more detail, server adaptation relies on two main components for task incremental learning that are well suited to fulfill our constraints for unsupervised and scalable user adaptation. First, Incremental Moment Matching (IMM) [15] yields a sequence of task-specific models, restraining new task models to reside close to the previously learned model. Averaging these models presumes a convex-like search space of the loss function, aiming for a single merged model optimal for all tasks. Although weighted averaging provides scalability in the number of tasks, IMM presumes fully supervised model weighting. Therefore, to overcome this need for supervised user data, we incorporate Memory Aware Synapses (MAS) [1] deriving parameter importance from unlabeled data, to attain scalable and fully unsupervised *Remote Adaptive Continual Learning* (RACL).

Secondly, DUA establishes further local user adaptation of the model obtained from the server. Our domain adaptive approach considers the fact that user and server data distributions resemble two different domains, where we want to transfer domain knowledge from server to user domain. A particularly suitable domain adaptation approach is Adaptive Batch Normalization (AdaBN) [16], which simply collects Batch Normalization (BN) statistics from the target user data. Domain knowledge of the user is assumed to reside in these BN statistics, which can be retrieved at low computational cost and without any supervision. Consequently, this unsupervised setup can enhance any method to become (more) user-adaptive.

The scope of this paper includes user adaptation within the continual learning paradigm, with threefold contributions in this unexplored setup:

- We establish an inherently scalable and privacy-preserving Dual User-Adaptation framework (DUA), flexibly disentangling user adaptation from the server and local user device.

- We introduce a novel benchmark, specifically designed for the evaluation of locally adaptive models in an incremental learning setting.

- We provide empirical evidence supporting the validity of IMM mode-merging with unsupervised MAS importance weights, and find importance weights to depend on the model rather than data.

Adapting to user data with RACL culminates in several advantages for both server and user. The server establishes a single sequence of N task-specific models using IMM. As a consequence, independence is imposed on the typically excessive amount of users L. Further, the continual learning setup enables the server to accumulate its knowledge with new arriving batches of data. Hence, this evades rebuilding the server knowledge base from scratch, which would impose time-consuming retraining and storage of all seen data. Moreover, storing one model per task instead of its training dataset results in enhanced storage requirements, especially when task data greatly exceeds model size. Additionally, users only share model parameter importance instead of their raw data, confining shared information to model-specific gradients. On top of that, users are not required to label local data as importance is measured in an unsupervised manner. However, when a subset of labeled user data is available, performance can be increased further by local user adaptation, as we will show later.

2. Related Work

The DUA framework introduces a new paradigm for user adaptation on the server, resembling federated learning [23], although completely overturning the purpose. Federated learning updates a common server model with an aggregated gradient from a distributed database, wherein each user constitutes a node providing local gradients. Similarly, DUA solely uses user-specific gradients to acquire better models, but attains decentralized user-personalized models, instead of a general trend-following model. Our framework invigorates profound overall user privacy, ensuring no sensible raw user data has to be shared, and additionally tackles the challenging issue of scalability for millions of personalized neural networks.

Further, sequentially learning multiple tasks by finetuning a neural network results in significant loss of previously acquired knowledge. Literature on continual learning largely addresses coping with this catastrophic forgetting [5, 25]. Nonetheless, recent works mainly focus on supervised data, leaving the richness of available unsupervised user data unused. Following [5], these methods can be subdivided into three main categories. First, *parameter-isolation* methods preserve task knowledge by obtaining task-specific masks [22, 21, 31], or dynamically extending the architecture [30]. *Replay* methods preserve a subset of representative samples of the previous tasks, replayed during training of new tasks. These exemplars can be raw images [20, 2, 29], or virtual samples retrieved from task-specific generative models [32]. Rao *et al.* [28] extend virtual replay to a completely unsupervised setting based on variational autoencoders. However, this would require exhaustive training on the low capacity edge-device of the user, with only a limited set of available user data, hence infea-

sible for user personalization. Finally, *regularization-based* methods impose a prior in the loss function when training the new task. Learning without forgetting (LwF) [17] minimizes a KL divergence prior to remain close to the new sample's output on the previous task model, hence distilling previous task knowledge [8]. Further work [27] extends this idea with task-specific autoencoders, additionally penalizing new task features to drift away from features deemed important for previous tasks. Elastic Weight Consolidation (EWC) [11] introduces a prior on previous-task parameters in a sequential Bayesian framework, Laplace approximated by a Gaussian with diagonally assumed Fisher information matrix (FIM) as precision. As the FIM is estimated in the task optimum, Zenke *et al.* [35] propose an online approach to estimate precision during training instead. Furthermore, the FIM relies on the loss gradient $\nabla \mathcal{L}$, whereas MAS [1] sidesteps this supervised loss dependency by relying on the output gradient ∇F instead. IMM [15] differs from previously discussed methods in first preserving trained task models, which are subsequently merged using FIM importance weights or by averaging.

For server user-adaptation in our DUA framework, multiple task-specific models are compressed into a single model. This is in the same vein to several other works. Chou *et al.* [4] merge two task-specific networks, subsequently finetuned with both task data. Although reduced training time is targeted, it remains unfit for scalable personalization, requiring raw user data and data of all tasks. Cheung *et al.* [3] superpose different models into a single one from which task-specific parameters can be retrieved. However, adapting models to users linearly increases training time. Another compression route distills knowledge [8] from teacher networks into smaller nets. Nevertheless, the focus of this work is model compression to achieve a smaller model for deployment, without addressing scalability to employ user personalization.

Finally, deep domain adaptation introduces several unsupervised back-propagation based techniques [6, 19], with state-of-the-art introducing an adversarial loss during training [34, 9]. The unsupervised setting befits these methods for adaptation to unlabeled user data. However, user data is required during training, and therefore unscalable as each personalized model would require training from scratch.

3. Methodology

3.1. Dual User-Adaptation Framework

A flexible framework should enable user-personalization both on the server and locally on the user edge-device. For this purpose, the novel Dual User-Adaptation framework (DUA) respectively divides user-personalization in two adaptation functions ψ and ϕ. The key to optimal preservation of user privacy is that no raw user data is trans-

mitted. To achieve this in practice, interactions between user and server are typically encrypted. However, when encryption fails, our framework provides additional concealment of explicit user-data with ψ. Further, adaptation on the server with ψ should be scalable, as training personalized models from scratch would require tremendous resource time for the vast amount of interacting users. Ideally, both adaptation functions ψ and ϕ favor unsupervised local adaptation to limit required user interactions. The DUA framework is surveyed in Figure 1, with its two user-adaptation phases elaborately discussed in the following.

Server user-adaptation. Server S has a set of task-specific expert models $\mathcal{M} = \{M_1, \ldots, M_N\}$, trained sequentially on a sequence of N tasks from its typically labeled data d_S. The Markov assumption holds for $\mathcal{M}$, with each model depending only on current task data and previous task model, as parameters θ_{t+1} of model M_{t+1} are initialized with θ_t. $\mathcal{M}$ is continuously extendable with new task models M_{N+1}.

L users interact with S, for which a user l provides for each M_t an obscured implicit prior $\psi(d_l, M_t)$ to the server, based on the local raw user-specific data d_l. To prevent transmission of raw user data, the ψ function uses a task-specific model M_t and corresponding local data d_l to extract concealed user-specific information. This ensures additional privacy safety, as the conveyed user prior is expressed by an implicit proxy, rather than the user data itself. Ideally, ψ does this without local supervision of the user, as we discuss in Section 3.2.

Once the server convokes all information in set $\Psi_l = \{\psi(d_l, M_t) | \forall M_t \in \mathcal{M}\}$ for user l, aggregating function χ delivers the ultimate user-personalized model on the server $\hat{M}_l = \chi(\Psi_l, \mathcal{M})$.

Local user-adaptation. User l receives $\hat{M}_l$ from the server on the edge device, which can be prone to further local adaptation function ϕ to accomplish the final model $M_l^* = \phi(d_l, \hat{M}_l)$. As no data needs to be transmitted, the user can fully exploit its raw local data d_l. However, local adaptation is limited in resources, inhibiting exhaustive training procedures for the model. Flexibility in this framework facilitates ϕ to further process an already personalized $\hat{M}_l$ as in the first DUA phase, or any general model without serverside personalization. This endows DUA to extend any method delivering a single model, with adaptation to the local user data d_l as in Section 3.3.

3.2. Unsupervised Server User-adaptation

To enable the server to perform user-adaptation in the DUA framework, we first have to define how to constitute task experts in $\mathcal{M}$, the user-adaptation function ψ, and the aggregating function χ to establish the final personalized model $\hat{M}_l$. This section explores opportunities in the challenging setup of task incremental continual learning, with

Figure 1: The Dual User-Adaptation framework (DUA): (1) server user-adaptation involves adaptation to local user data d_l with ψ for each model in $\mathcal{M}$. Aggregating function χ incorporates all models $\mathcal{M}$ and resulting user priors Ψ_l into single model $\hat{M}_l$. (2) Local user-adaptation consists of adaptation function ϕ mapping $\hat{M}_l$ to the final personalized model M_l^*.
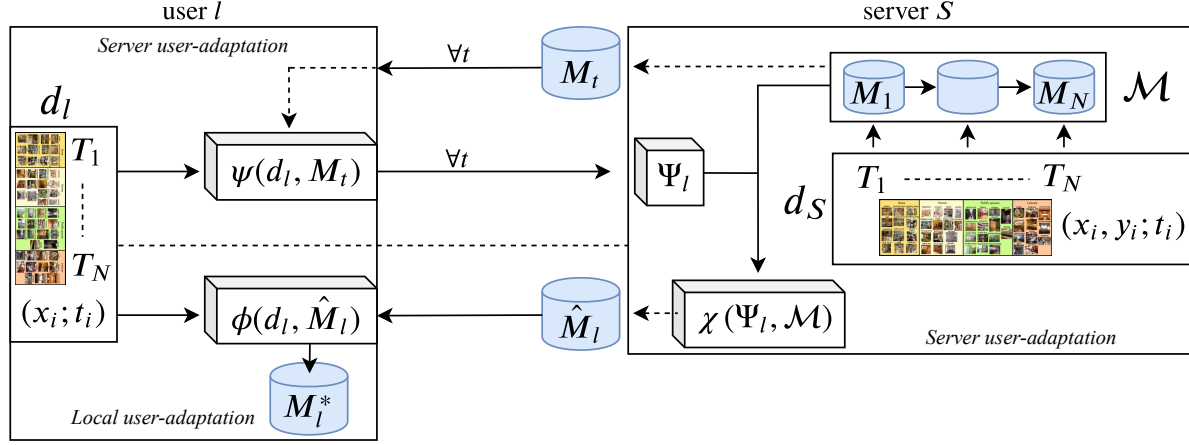
only data of the new task T_n available, and inhibiting data access for previously learned tasks. This is particularly suited for the server, which can learn a new task, discard the new task data D_n, and keep only the model M_n to enable incremental learning for further tasks. Thus, server data d_S at any point comprises only new task data distribution D_n, from which $(x_i, y_i; t_n)$ is sampled, respectively constituting the image x_i, label y_i, and task index t_n.

The task experts in $\mathcal{M}$ and the aggregating function χ are defined following Lee *et al.* [15] with Incremental Moment Matching (IMM). Parameter uncertainty is introduced using the Bayesian framework, wherein incremental training of tasks results in a new task posterior after training. Task posteriors are presumed Gaussian, with the task sequence posterior aggregating these components in a mixture of Gaussians. Mode-IMM [15] Laplace approximates the mixture with a single Gaussian, and for this assumption to hold, a smooth and convex loss search space is required between the posterior means of the mixture components. Therefore, we adopt the weight and L2 transfer techniques proposed in [15], respectively initializing the network with previous task weights, and urging the new task optimum to remain close to the previous task optimum by L2-regularization. The aggregating function χ constitutes the mode of the final Laplace approximation of the Gaussian mixture with N components, and is formalized by its mean

$$\hat{\theta}_l = \frac{1}{\hat{\Omega}_l} \sum_t^N \alpha_t \Omega_t \theta_t \,, \tag{1}$$

and precision

$$\hat{\Omega}_l = \sum_t^N \alpha_t \Omega_t \,. \tag{2}$$

for user l, with precision Ω_t of task T_t. Mixing ratio α_t weighs importance of task T_t, subject to $\sum_t^N \alpha_t = 1$. The

balanced tasks in our experiments are deemed equally important.

Further, the user adaptation function ψ should produce an unsupervised implicit user prior from both the raw user data d_l and a given model. Precision indicates the degree of parameter certainty, and therefore resembles a parameter importance measure. As this resembles an implicit prior, we define $\psi(d_l, M_t) = \Omega_t$. Nonetheless, to estimate task precision Ω_t, mode-IMM employs the Fisher information matrix (FIM) similar to [11]. The FIM is constituted by second-order derivatives of the loss function, thus requiring labeled data. In contrast, to enable unsupervised importance measures, we exert MAS importance weights [1] based on the expected gradient of the L2-norm of the output function with respect to parameter θ_t^k,

$$\Omega_t^k = \mathbb{E}_{x \sim D_t} \big[\frac{\nabla \|F(x; \theta)\|_2^2}{\delta \theta_t^k} \big] \,, \tag{3}$$

with $D_t \in d_l$ the unlabeled user data distribution for which importance is measured. Aggregating mode-IMM and MAS importance weights constitutes *Remote Adaptive Continual Learning* (RACL) for server-side user-adaptation in the DUA framework.

3.3. Unsupervised Local Domain Adaptation

The second component enabling dual user-adaptation in the DUA framework is local adaptation function $\phi(d_l, \hat{M}_l)$, constrained by limited resources on the edge device. For lightweight adaptation with ϕ, we could adapt to the user data statistics with Batch Normalization (BN) [10]. During training, each input feature x_k of the BN layer is normalized to $\hat{x}_k$ using current batch statistics. Subsequently, scaling γ_k and shift β_k parameters are learned, producing the

normalized output y_k, with

$$\hat{x}_k = \frac{x_k - \mathbb{E}\left[x_k\right]}{\sqrt{\text{Var}\left[x_k\right]}}, \qquad (4)$$

$$y_k = \gamma_k \hat{x}_k + \beta_k. \qquad (5)$$

Whereas BN obtains global batch statistics of the training data for inference, Adaptive BN (AdaBN) [16] introduces an unsupervised scheme gathering batch statistics of the target domain data instead. The main idea is for domain knowledge to reside in the batch statistics, rather than the parameters to optimize. In our setup, the target domain is task-specific user data d_l, enabling unsupervised user adaptation with AdaBN.

Relaxing the constraint for unsupervised adaptation, we can assume a labeled subset in the user data. Although this would facilitate finetuning on the user device, the computation and storage limitations both restrain us from computing gradients for all network parameters. Alternatively, we extend AdaBN to this supervised setting (AdaBN-S), additionally training BN layer parameters γ and β for a few epochs, while freezing all remaining network parameters. This approach significantly reduces the number of trainable parameters compared to finetuning, scaling down computational effort by faster convergence and diminishing storage requirements for the gradients.

4. User Personalization Benchmarks

In order to evaluate the DUA framework, we need datasets mimicking user-specific data. Our experiments comprise three different data setups[1]. In all setups, the server data is split into training and validation sets with a ratio of $80/20$, and user-data is split into equally sized evaluation and user-validation sets. User adaptation techniques such as importance weight estimation or user-specific finetuning solely access the user-validation subset, evading overfitting to the evaluation set or tuning on test data.

Two setups are based on the MIT Indoor Scene recognition dataset (MITIS) [26], divided into tasks according to the five scene supercategories. Omitting supercategory 'work' as its extra data is too limited, the final task sequence is defined as {home, leisure, public, store}. All images have a minimal resolution of 200 pixels on the smallest axis, are randomly cropped and horizontally flipped during training, and then resized to 224×224. MITIS training data is available to the server, following the continual learning paradigm in only providing access to current task data. Evaluation and extra MITIS data are divided over users in the following two schemes:

1. **Category Prior (CatPrior)**. Five users each prefer a random subset of 3 categories per task, acquiring for

---

[1]Code available at: `https://github.com/mattdl/DUA`

Figure 2: TransPrior user transformations following [7] with severeness three, including: spatters, elastic transformation, saturation, defocus blur, Gaussian noise, brightness, Gaussian blur, jpeg compression, contrast and impulse noise.

each preference 250 extra MITIS images. The 20 evaluation images per category are equally divided over users. All user data is mutually exclusive.

2. **Transform Prior (TransPrior)**. Ten users each exhibit a different type of transform following [7] with perturbation severeness 3 (in range 1 to 5), permuting 1000 randomly sampled images from extra MITIS data, and all MITIS evaluation data. All user data prior to transformation is identical. See Figure 2 for examples of the ten types of transformed MITIS images.

For all users, Monte Carlo Cross-validation over 5 iterations is performed over the extra MITIS user data, with priors remaining fixed.

The third setup (**Numbers**) comprises handwritten digits from MNIST [14] and Street View House Numbers (SVHN) [24] data, divided into five tasks of two subsequent numbers as in {0,1} to {8,9}. MNIST comprises 28×28 images, with 32×32 SVHN images center-cropped to match this resolution. Server data is constituted by both MNIST and SVHN training data, and two users are each characterized by the evaluation data of MNIST and SVHN respectively. Results are averaged over three trained models, initialized with different seeds.

5. Experiments

5.1. Evaluation Setup

Models. Experiments with both MITIS setups use AlexNet [12] and VGG11 [33] models, except for the BatchNorm experiments which are only applied to VGG11. Due to the small input size and simplicity of the Numbers benchmark, a small three-layer MLP with two hidden layers of 100 units suffices for this setup. For AlexNet and VGG11, we start from models pretrained on ImageNet, while the MLP model is trained from scratch.

Table 1: Qualitatively comparing features: user-adaptive (Adapt.), unsupervised (Unsup.), scalable (Scal.) and privacy-preserving (Priv.). DUA subdivides adaptation on the server (ψ) and local user device (ϕ), with MAS importance weights discarding supervision. Scalability for user-adaptive methods implies training independent of the number of users L. Shared user-data can be raw d_l, gradients of the output function $F(x; \theta)$ or loss $\mathcal{L}(x, y; \theta)$. All methods can be extended with unsupervised (AdaBN) and supervised (AdaBN-S) local user adaptation ϕ.

Method	Adapt.	Unsup.	Scal.	Priv.
MAS-RACL	$\checkmark(\psi)$	$\checkmark$	N	∇F
FIM-RACL	$\checkmark(\psi)$		N	$\nabla \mathcal{L}$
Task Experts	$\checkmark$		$N \cdot L$	d_l
MAS-IMM		$\checkmark$	N	
FIM-IMM			N	
MAS		$\checkmark$	N	
EWC			N	
LWF			N	
Joint			N	
+ AdaBN	$\checkmark(\phi)$	$\checkmark$		
+ AdaBN-S	$\checkmark(\phi)$			

Evaluation. For all experiments, we report average accuracy and forgetting on the final model after training all tasks. Depending on the method, this final post-merging model is either user-specific or the general server model. Results are averaged over all users.

Methods can be subdivided into user-specific and user-agnostic approaches. Table 1 summarizes all method features in our user-adaptive setting.

User-Specific methods adapt to the local user-validation set of the user, resulting in a personalized model.

1. **MAS-RACL** is our server user-adaptation method discussed in Section 3.2, merging task-specific IMM models on the server with unsupervised MAS importance weights obtained from the user-validation set.

2. **FIM-RACL** is a variant of MAS-RACL using the FIM as importance measure to merge the task-specific IMM models. The FIM is constituted by gradients of the loss, hence requiring labeled user data. This baseline serves as a performance reference for the MAS importance weights in MAS-RACL.

3. **AdaBN** adapts to the user BN statistics in an unsupervised fashion, only demanding one forward pass for all user-validation data (details in Section 3.3).

4. **AdaBN-S** collects batch statistics while training BN

parameters for a few epochs, hence requiring supervision in the user-validation data (details in Section 3.3).

5. **Task Experts** are obtained by finetuning each task-specific IMM server model on the raw user validation data for several epochs with low learning rate. This results in an ensemble of task-specific expert networks for each user. This is not scalable and should be regarded an upper bound for user-specific models.

User-Agnostic methods only access server training and validation data, and therefore are not adapted to the user.

1. **FIM-IMM** trains a model per task, subsequently merged using a per-parameter importance measure estimated on the server validation data. Identical to mode-IMM in [15].

2. **MAS-IMM** is a variant of FIM-IMM using MAS importance weights.

3. **Joint training** optimizes all tasks at once, accessing all task data simultaneously. This violates the continual learning setup, and is considered as a weak upper bound for performance.

4. **EWC** [11] preserves previous task knowledge using FIM-based importance weights.

5. **MAS** [1] instead uses the gradient of L2 norm of the output function to measure importance.

6. **LwF** [18] uses knowledge distillation with new task data outputs obtained from the previous task network.

No gridsearches are performed for forgetting-related hyperparameters as the previous task data is assumed unavailable in the continual learning paradigm. Therefore, the recommended setting of the original works is used. Other hyperparameters are determined from best performance on the Joint baseline, yielding 30 and 10 epochs with a learning rate of $1e^{-3}$, and batch size 30 and 20 for MITIS and Numbers respectively. After five epochs of unimproved validation accuracy, the learning rate anneals with a factor of 0.1, stopping early after five more subsequent inferior epochs.

Table 2: Reporting average accuracy (forgetting) for IMM mode-merging with both unsupervised (MAS) and supervised (FIM) importance weights.

Data Setup	Model	MAS-IMM	FIM-IMM
CatPrior	AlexNet	67.39 (0.73)	67.42 (0.23)
	VGG11	76.77 (0.30)	76.29 (0.43)
TransPrior	AlexNet	46.51 (-0.14)	46.68 (-0.35)
	VGG11	53.49 (-0.17)	53.14 (0.07)
Numbers	MLP	84.36 (-0.40)	87.68 (0.07)

Table 3: *Left*: Average accuracy (forgetting) for the three data setups and models, comparing user-specific (RACL) and user-agnostic (IMM) importance weights, both unsupervised (MAS-) and supervised (FIM-). RACL outperforming the corresponding IMM variant is indicated in bold. *Right*: Qualitatively comparing features user-adaptive (Adapt.), unsupervised (Unsup.), scalable (Scal.) and privacy-preserving (Priv.).

Method	Alexnet		VGG11		MLP	Adapt.	Unsup.	Scal.	Priv.
	CatPrior	*TransPrior*	*CatPrior*	*TransPrior*	*Numbers*				
MAS-RACL	66.97 (0.88)	**47.04 (-0.27)**	**77.32 (0.77)**	**53.59 (-0.14)**	84.01 (-0.22)	✓	✓	✓	✓
MAS-IMM	67.39 (0.73)	46.51 (-0.14)	76.77 (0.30)	53.49 (-0.17)	84.36 (-0.40)	✗	✓	✓	✓
FIM-RACL	67.20 (0.73)	**47.32 (-0.51)**	76.53 (0.68)	**53.73 (-0.13)**	**87.83 (0.30)**	✓	✗	✓	✓
FIM-IMM	67.42 (0.23)	46.68 (-0.35)	76.29 (0.43)	53.14 (0.07)	87.68 (0.07)	✗	✗	✓	✓

5.2. Unsupervised Moment Matching

The first experiment studies similarity in IMM mode-merging performance of the original supervised FIM, and the proposed unsupervised MAS importance weights. The results in Table 2 show similar performance for MAS-IMM and FIM-IMM. However, we observe a more salient discrepancy for the Numbers setup, where FIM attains 3.32% higher average accuracy, despite 0.47% increased forgetting. Analyzing importance weights, the first Numbers task $\{0,1\}$ results are an order of magnitude higher compared to subsequent tasks in the sequence. By contrast, FIM importance weights obtain the same order of magnitude over all tasks. As a consequence, MAS importance outweighs stability in first task knowledge, but deteriorating adaptation to new tasks. Further, the Numbers MLP model with few parameters is trained from scratch, with the first task learning only a limited set of discriminating features. The output function magnitude depends only on the features learned for this binary task, and especially the limited MLP network size implies greater output function sensitivity to changes in the intermediate features, which may be substantial when learning the second task. In contrast, the AlexNet and VGG networks initialize the network with an output function depending on vast Imagenet pretraining. In conclusion, the MAS importance measure provides competitive outcomes, especially for the setups with pretrained networks.

5.3. Locally Adapting to the User

As the MAS importance weights empirically prove a valid alternative precision measure for mode-merging in IMM, we can now adapt to local user data d_l in an unsupervised fashion. Table 3 reports average accuracy over all users for both user-specific and user-agnostic importance weights. The majority of locally estimated importance weights of RACL results in small improvements. The benefit of locally adapting to the user seems minimal.

To better understand why this is the case, we further scrutinize data dependency of importance weights measuring Pearson correlation ρ. In Figure 3 we consider user-validation data of two CatPrior tasks 'home' (D_1) and 'leisure' (D_2). Both tasks have corresponding optimized server models M_1 and M_2, for which our original task incremental setup calculates the importance weights. In contrast, for this analysis we compare importance weight correlation for both datasets on the same model, resulting in correlation coefficients 0.82 and 0.73 for M_1 and M_2 respectively (see Figure 3 (a) and (b)). This high correlation implies limited dependence of the importance weights on the data. Next, we compare correlation for the same dataset, yet calculated on the two different models. Correlations for D_1 and D_2 yield 0.55 and 0.58, which is significantly lower, hence indicating higher dependence on the model (see Table 3 (c) and (d)). In conclusion, importance weights represent parameter importance within the specific model rather than the data they are estimated from. As a result, there is little to be gained by estimating them on user-specific data.

5.4. Adapting to the User Domain

Borrowing ideas from domain adaptation, we can extend any method to become user-specific (see Table 1). For this experiment, we employ the VGG11 model from previous experiments interspersed with BN layers after each block of convolutional and ReLU activation layers (VGG11-BN). Table 4 shows results for the CatPrior and TransPrior setups. Note that Task Experts inherently adapts BN parameters of the VGG model, as it finetunes to the user data d_l. In general, the unsupervised AdaBN mainly exhibits limited gain for RACL, and seems ineffective for the remaining continual learning methods. In contrast, the supervised variant AdaBN-S consistently outperforms user-agnostic BN, with an average gain of 2.64% and 3.44% accuracy in Cat-Prior and TransPrior setups respectively. The discrepancy between AdaBN and AdaBN-S performance discloses unsupervised adaptation to remain a challenging open problem. Remarkably, Joint training with VGG11-BN performs worse than VGG11 results without BN in Table 3. This seems related to overfitting with an observed 10% increase in discrepancy between training and validation accuracy.

Figure 3: Visualizing user importance weight correlation ρ in the CatPrior setup, for the first two tasks. Blue and orange represent weight and bias importance weights respectively. (a) and (b) each compare importance weights of different task data on the same task-specific model, whereas (c) and (d) each use the same data on the two task-specific models.

(a) $M_1 - D_1$ vs D_2
$\rho = 0.82$

(b) $M_2 - D_1$ vs D_2
$\rho = 0.73$

(c) M_1 vs $M_2 - D_1$
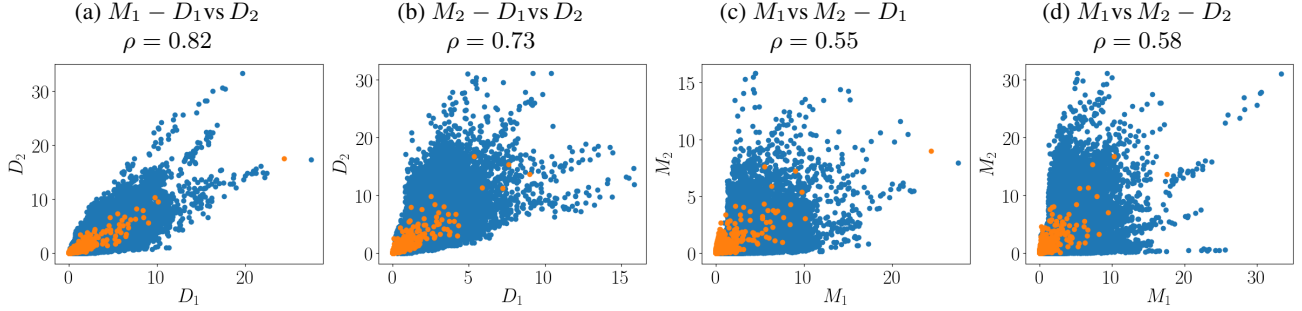$\rho = 0.55$

(d) M_1 vs $M_2 - D_2$
$\rho = 0.58$

Table 4: Results in the CatPrior and TransPrior setups with model VGG11-BN, comparing batch normalization on the server data (BN) with unsupervised (AdaBN) and supervised (AdaBN-S) user-adaptive variants.

Method	CatPrior			TransPrior		
	BN	*AdaBN*	*AdaBN-S*	*BN*	*AdaBN*	*AdaBN-S*
MAS-RACL	58.05 (2.74)	58.30 (2.34)	60.68 (2.67)	30.14 (2.69)	30.19 (2.50)	32.82 (3.25)
FIM-RACL	59.58 (2.14)	59.71 (1.61)	62.43 (1.84)	32.15 (1.53)	32.04 (1.33)	34.80 (2.13)
Task Experts	80.78 (5.61)	n/a	n/a	68.22 (11.35)	n/a	n/a
MAS-IMM	55.55 (2.69)	55.89 (2.69)	58.87 (2.81)	29.36 (2.63)	29.15 (2.45)	31.73 (3.22)
FIM-IMM	61.50 (-0.03)	61.35 (-0.46)	63.99 (-0.16)	32.08 (1.32)	31.86 (1.21)	34.48 (2.05)
MAS	65.58 (3.96)	64.15 (4.04)	67.10 (4.66)	37.32 (2.64)	35.64 (2.88)	40.51 (2.69)
EWC	66.20 (2.88)	64.03 (3.43)	67.54 (3.90)	37.16 (2.85)	35.44 (3.12)	40.05 (3.18)
LWF	70.76 (0.73)	70.37 (0.43)	72.73 (1.03)	40.22 (0.43)	39.51 (0.12)	43.07 (0.52)
Joint	75.75 (n/a)	72.13 (n/a)	76.39 (n/a)	46.53 (n/a)	41.18 (n/a)	48.50 (n/a)

By normalizing over batch statistics, BN layers alleviate the internal covariance shift in the network. For a network prone to overfitting due to few data, this internal covariance shift might instead introduce regularizing noise in the batch, interfering optimization to overfit the training data. In this respect, even though the server presents a plausibly overfitted model, adaptation to the user domain remains efficacious with AdaBN-S. Furthermore, the effects of BN on continual learning methods still urge further elicitation, as it is mainly disregarded in current state-of-the-art [5, 25].

In conclusion, unsupervised adaptation with AdaBN exhibits cumbersome adaptation to the user domain, although gaining significant improvement with a subset of labeled data in AdaBN-S.

6. Conclusion

In this work, we proposed a practical Dual User-Adaptation framework (DUA) to tackle incremental domain adaptation to real-life scenarios with numerous users. This novel user-adaptation paradigm disentangles personalization to both the server and local user device, and combines

desirable user privacy and scalability properties, which remain highly unexplored in literature. We devised benchmarks to scrutinize both types of user-adaptation.

First, adapting models on the server following RACL incurs these scalability, privacy, and additional supervision properties, yet in practice yielded only marginal improvement over a user-agnostic model, due to gradient-based importance weights being largely data independent.

Second, local user-adaptation with a data regularization approach based on adaptive Batch Normalization (AdaBN), and especially its supervised variant (AdaBN-S), seem more promising, leading to systematic improvements when taking advantage of labeled user-specific data.

User privacy and experience are of major concern, for which our DUA framework forges a principled foundation for dual user-adaptation, aspiring to promote further research in this direction.

Acknowledgements

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018. 2, 3, 4, 6

[2] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *ICLR*, 2019. 2

[3] Brian Cheung, Alex Terekhov, Yubei Chen, Pulkit Agrawal, and Bruno Olshausen. Superposition of many models into one. *arXiv preprint arXiv:1902.05522*, 2019. 3

[4] Yi-Min Chou, Yi-Ming Chan, Jia-Hong Lee, Chih-Yi Chiu, and Chu-Song Chen. Unifying and merging well-trained deep neural networks for inference stage. *arXiv preprint arXiv:1805.04980*, 2018. 3

[5] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. *arXiv preprint arXiv:1909.08383*, 2019. 2, 8

[6] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014. 3

[7] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. 5

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 3

[9] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017. 3

[10] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 4

[11] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 3, 4, 6

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 5

[13] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. 1

[14] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. 5

[15] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in neural information processing systems*, pages 4652–4662, 2017. 2, 3, 4, 6

[16] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016. 2, 5

[17] Zhizhong Li and Derek Hoiem. Learning without forgetting. In *ECCV*, pages 614–629. Springer, 2016. 3

[18] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 6

[19] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pages 136–144, 2016. 3

[20] David Lopez-Paz et al. Gradient episodic memory for continual learning. In *NeurIPS*, pages 6470–6479, 2017. 2

[21] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *ECCV*, pages 67–82, 2018. 2

[22] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *CVPR*, pages 7765–7773, 2018. 2

[23] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016. 2

[24] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 5

[25] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019. 2, 8

[26] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–420. IEEE, 2009. 5

[27] Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *ICCV*, pages 1320–1328, 2017. 3

[28] Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. In *Advances in Neural Information Processing Systems*, pages 7645–7655, 2019. 2

[29] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017. 2

[30] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 2

[31] Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. *arXiv preprint arXiv:1801.01423*, 2018. 2

[32] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *NeurIPS*, pages 2990–2999, 2017. 2

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[34] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017. 3

[35] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3987–3995. JMLR. org, 2017. 3