FORGET TO LEARN (F2L): RETHINKING REPLAY LOSS IN UNSUPERVISED CONTINUOUS DOMAIN ADAPTATION.

Anonymous authors
Paper under double-blind review

ABSTRACT

Although continuous unsupervised domain adaptation (CUDA) has shown success in dealing with non-stationary data, catastrophic forgetting is still a challenge hindering its full potential. The current state-of-the-art (SOTA) focuses on training a single model to simultaneously perform adaptation (e.g., domain alignment) and knowledge retention (i.e., minimizing replay loss). However, the two conflicting objectives result in a hyper-parameter, which is difficult to tune yet significantly affecting model performance. Therefore, we propose to use two separate models so that one model is dedicated to the retention of historical knowledge (i.e., high stability) while the other to the adaptation to future domains (i.e., high plasticity). This allows the algorithm to forget to achieve better overall performance: dubbed as Forget to Learn (F2L), Specifically, F2L decomposes the training process into specialist model and generalist model, and uses knowledge distillation to transfer knowledge between the two models. We demonstrate the superiority of F2L compared to current CUDA trends (i.e., multi-task learning and single-task constrained learning) on different continuous unsupervised domain adaptation datasets.

1 INTRODUCTION

Continuous Unsupervised Domain Adaptation (CUDA) is a generalization of the traditional Domain Adaptation (DA) problem. However, in (CUDA), instead of dealing with a single target domain, the model must continuously adapt to sequentially and smoothly varying target domains. One famous realization of the CUDA problem is computer vision for self-driving cars under continuously and smoothly varying environments. A Deep learning vision algorithm pre-trained on images taken during the noon needs to continuously adapt to new lightning conditions due to the variation of sun position throughout the day. CUDA can deal with these kinds of problems without requiring extra labeling efforts. Hoffman et al. (2014) and Wulfmeier et al. (2018) represent early successful attempts to solve this problem. Hoffman et al. (2014) crafted domain invariant kernels using subspace alignment (SA) or Geodesic flow kernel (GFK) for each target domain. During inference and for each target, the corresponding learned kernel is used in any kernel machine for classification or regression. In recent years, more focus is made on directly training deep learning models to extract domain invariant features. Wulfmeier et al. (2018), inspired by Tzeng et al. (2017), have proposed an adversarial framework that breaks down CUDA continuous shifts into smaller incremental shifts. By adapting to each incremental shift and employing Adversarial Discriminative Domain Adaptation (ADDA) by Tzeng et al. (2017), they were able to adapt to the continuous shift. However, despite the success of these early attempts, they shared a significant drawback: the inability to handle catastrophic forgetting.

Catastrophic forgetting is a problem that presents itself in machine learning models both in a supervised setting (i.e., continuous learning) and in unsupervised settings such as CUDA. In case of Wulfmeier et al. (2018), the model M_t is adapted to the new current target domain D_T^t without any consideration of its performance on the previous domains $[D_0, ..., D_T^{t-2}, D_T^{t-1}]$. Accordingly, the model forgets most of its knowledge on the previous domain and performs very poorly when tested on them. This is a direct consequence of catastrophic forgetting. Many have extending the work done Wulfmeier et al. (2018) to find a way around such problem. Kim et al. (2020) proposed storing a few convolution layers (i.e., Target-specific Memory) from each model at each adaptation step,

and later during inference on a certain domain, the corresponding Target-specific Memory should be used. Although this is an effective approach, it assumes that the domain identity is given during inference, which is not necessarily true. To solve this issue, Schutera et al. (2021) proposed training a domain classifier. This domain classifier is trained to predict the domain index during inference. Once the domain index is known, and since all previous models are stored, $[M_0, M_1, M_2, ..., M_t]$ the right model can be selected, and thus catastrophic forgetting is avoided. One drawback of such an approach is that training of such domain classifier requires access to all future targets. Wang et al. (2020) proposed another approach that effectively can deal with catastrophic forgetting but also conditioned on the access to all future targets' data. They proposed training a regressor as the discriminator instead of the traditional binary classifier in the traditional DA approaches. In their approach, the regressor attempts to predict the domain index instead of just differentiating the source from the current target domain. Although these methods can be useful in alleviating catastrophic forgetting, they do not comply with the streaming nature of the general CUDA problem, where target data are assumed to arrive sequentially over time.

Multi-task approach has been the mainstream approach to addressing the problem of catastrophic forgetting. Bobu et al. (2018), proposed extending Wulfmeier et al. (2018) by utilizing an extra loss term denoted the replay loss. The usage of replay loss has been extensively studied in continuous-supervised learning settings Yoon et al. (2021). This replay loss encourages the model to retain previous experiences and stores previous pseudo-labeled samples in a memory buffer. Later in the adaptation step, the model simultaneously tries to be consistent with the previous experience (i.e., minimize the replay loss) as well as align the new target features with the source features. Similar to Wulfmeier et al. (2018), the alignment is done via training the discriminator and encoder in a two-player zero-sum game. The success of the multi-task approach (i.e., replay loss and adversarial alignment), proposed by Bobu et al. (2018), motivated utilizing the same technique in different CUDA settings. Rostami (2021) used the same multi-task objective function proposed by Bobu et al. (2018). However, instead of assuming access to the source data during training, they fitted a Gaussian mixture model to approximate the source data and source encoder distribution. They relied on a memory bank to store samples from previous domains to resist forgetting. Rakshit et al. (2022) slightly modified Bobu et al. (2018) objective. Instead of aligning the source distribution with the current target, they proposed aligning an auxiliary source domain with the current target. The auxiliary source domain is a union of the source domain and pseudo-labeled samples from the previous domain. Although approaches based on multi-task training (e.g., Bobu et al. (2018)) can effectively reduce catastrophic forgetting, it still relies on finding an optimum trade-off between the two tasks at hand. In the original article Bobu et al. (2018), the authors multiplied the replay loss by weighing factor λ that was found using grid search on a labeled validation target dataset, which does not comply with the unsupervised nature of CUDA.

Huang et al. (2021) and Tang et al. (2021) treat the replay loss as a constraint to turn the multi-task problem into a constrained single task problem. This approach eliminates the hyper-parameter balancing between two conflicting objectives. By doing so, and during adapting, the model parameters are only allowed to change if the replay loss term does not increase. Note, in this formulation, the need for deciding the optimum λ value is eliminated. However, the downside of this approach is (1) the complexity of solving such a constrained optimization problem (i.e., quadratic programming problem in the number of the model parameters, refer to the original article for more details), (2) this constrained optimization approach can limit model plasticity. This is especially critical due to machine learning systems' vulnerability to the stability-plasticity dilemma Delange et al. (2021), and Mermillod et al. (2013). Stability refers to the neural network's ability to remember the previous experience, while plasticity refers to the neural network's capacity to acquire new knowledge continuously. Hard constrained optimization in Huang et al. (2021) or overly emphasized replay loss in Bobu et al. (2018) can significantly deteriorate model plasticity.

In the above-reviewed literature, two main approaches for solving catastrophic forgetting in the CUDA setting have been proposed, unconstrained multi-task optimization and single-task constrained optimization. Both approaches follow different techniques in an attempt to train a single model that can simultaneously adapt to future domains as well as remember previous domains. Unfortunately, neural networks are susceptible to the stability-plasticity dilemma. Accordingly, the two conflicting objectives - stability (i.e., remembering previous domains) and plasticity (i.e., acquiring new knowledge) -are unrealistic goals for a single model, and therefore we propose a fundamentally new approach that trains two different models to overcome this dilemma. The first model, which

we refer to as the domain specialist, is permitted to forget in order to give it a better chance to acquire new knowledge. Hence, we call our approach as Forget to Learn (F2L). The second model is continuously learning and accumulating knowledge from the domain specialist model. Since the second model is accumulating knowledge, it resists forgetting and can be used on any previous domain at any point in time; thus, it is called the generalist. F2L is as simple as Bobu et al. (2018), requires no λ hyper-parameter tuning as Huang et al. (2021) and achieves a better stability-plasticity trade-off than both. To the best of our knowledge, F2L is the first attempt to address the issue of the plasticity-elasticity dilemma in continuous unsupervised domain adaptation.

2 THE METHOD

We propose to train two separate models to tackle the stability-plasticity dilemma. This is done by disjointing the learning objective. To do so, we train two encoders, which we refer to as Domain-Specialist (E) and Generalist ($\widehat{E}$). By allowing the Domain-Specialist to forget, we can achieve superior performance in terms of both stability and plasticity.

2.1 LOSS FUNCTIONS

The existing works, such as Bobu et al. (2018) and Huang et al. (2021), attempt to balance between the domain alignment task and the knowledge retention task. The two tasks are defined with respective loss functions. The domain alignment task trains the current target encoder ($\widehat{E_T^t}$) to adapt from the source domain (D_0) to the new current target domain (D_T^t) by minimizing the discrepancy between the source domain representation and the target domain representation $\widehat{E_T^t}(X_t^t)$ by minimizing the following loss function:

$$d\left[\widehat{\mathrm{E}_0}\left(\mathrm{X}_0\right), \widehat{E_T^{\mathrm{t}}}\left(X_t^{\mathrm{t}}\right)\right] \tag{1}$$

where $\widehat{E_0}$ is the pre-trained source encoder and d is a distance metric that represents the misalignment between the two domains. Bobu et al. (2018) used ADDA approach proposed by.Tzeng et al. (2017) for the evaluation and minimization of d. X_T^t is the current target unlabeled streaming dataset $(X_T^t) \sim p_{D_T^t}(x)$ while X_0 is the source domain dataset $X_0 \sim p_{D_0}(x, y)$. $p_{D_0}(x, y)$ and $p_{D_T^t}(x, y)$ is the joint probability distribution in the input and label space for the source (D_0) and target (D_T^t) domains respectively. (X_0, Y_0), and (X_t^t) are defined as follow. $(X_0, Y_0) \coloneqq \left\{(x_1, y_1), \ldots, (x_N, y_N) \mid (x_i, y_i) \overset{iid}{\sim} p_{D_0}(x, y), \forall i\right\}$ and $(X_t^t) \coloneqq \left\{(x_1), \ldots, (x_N) \mid (x_i) \overset{iid}{\sim} p_{D_T^t}(x), \forall i\right\}$.

On the other hand, the knowledge retention task tries to retain knowledge from previously trained tasks by assuring consistency between previous models $\{E_T^\tau \mid \tau \in [0 \rightarrow t-1]\}$ predictions $Y_p^t = C_0(\widehat{E_T^\tau}(X_p^t))$ and current model predictions $C_0(\widehat{E_T^t}(X_p^t))$. $(X_p^t, Y_p^t) \in M_t$ are previous experiences from a memory buffer M_t. $X_p^t \subset X_T^t$, $X_p^t \overset{K}{\sim} p_{D_T^t}(x)$ and K is the memory buffer size per domain. Cross-entropy loss, as given below, is commonly used to train the model for the memory retention task:

$$\mathrm{CE}\left[C_0\left(\widehat{E_T^{\mathrm{t}}}\left(X_p^{\mathrm{t}}\right)\right), Y_p^{\mathrm{t}}\right], \tag{2}$$

where C_0 is the source domain classifier, CE is cross-entropy loss,

Bobu et al. (2018) try to balance between the two objectives by combining the two loss functions with a hyper-parameter λ, and minimize the following multi-task loss:

$$\widehat{E_T^{\mathrm{t}*}} = \mathrm{argmin}_{\widehat{\mathrm{E}}_{\mathrm{T}}}\left\{d\left[\widehat{\mathrm{E}_0}\left(\mathrm{X}_0\right), \widehat{E_T^{\mathrm{t}}}\left(X_t^{\mathrm{t}}\right)\right] + \lambda * \mathrm{CE}\left[C_0\left(\widehat{E_T^{\mathrm{t}}}\left(X_p^{\mathrm{t}}\right)\right), Y_p^{\mathrm{t}}\right]\right\}. \tag{3}$$

The performance of this approach turned out overly sensitive to the hyper-parameter λ as discussed in the following sections. An alternative approach is to treat one of the two objectives as a constraint. Tang et al. (2021) proposed two major modifications: (1) for domain alignment, they used a unified contrastive loss instead of the adversarial ADDA-based alignment and (2) they dropped the retention

loss term from the multi-task loss and instead used a target memorization (TM) constraint as follows [1]:

$$\begin{aligned} \min_{g_w} \quad & \tfrac{1}{2}\left\|g_w - g_t\right\|_2^2 \\ \text{subject to} \quad & \langle g_w, g_{dm}\rangle := \left\langle g_w, \frac{\partial \mathrm{CE}\left[C_0\left(\widehat{E_T^t}\left(X_p^t\right)\right), Y_p^t\right]}{\partial \theta_t}\right\rangle \geq 0, \end{aligned} \tag{4}$$

where g_t is the gradient for the alignment loss (i.e., contrastive loss in the original paper), g_w is the gradient to update the network, g_{dm} is the gradient of replay loss, (i.e., antiforgetting term), and θ_t is the parameterization of the encoder $\widehat{E_T^t}$.

2.2 Forget to Learn (F2L)

As explained earlier, the current trend in the literature is to simultaneously train a single encoder to fulfill the following two conflicting objectives: (1) adapt to future domains and (2) retain knowledge from previous domains. Instead, in the present study, we propose a disjoint training approach (i.e., adaptation-distillation training). We train two encoders: the domain specialist E_T^t and the generalist encoder $\widehat{E_T^t}$ for each target domain (D_t). The training can be divided into three steps, as shown in Figure 1. These steps are: (1) the domain specialist encoder training at the top (i.e., the adaptation step), (2) memory update at the middle, and at the bottom, (3) the generalist encoder training (i.e., the distillation step).

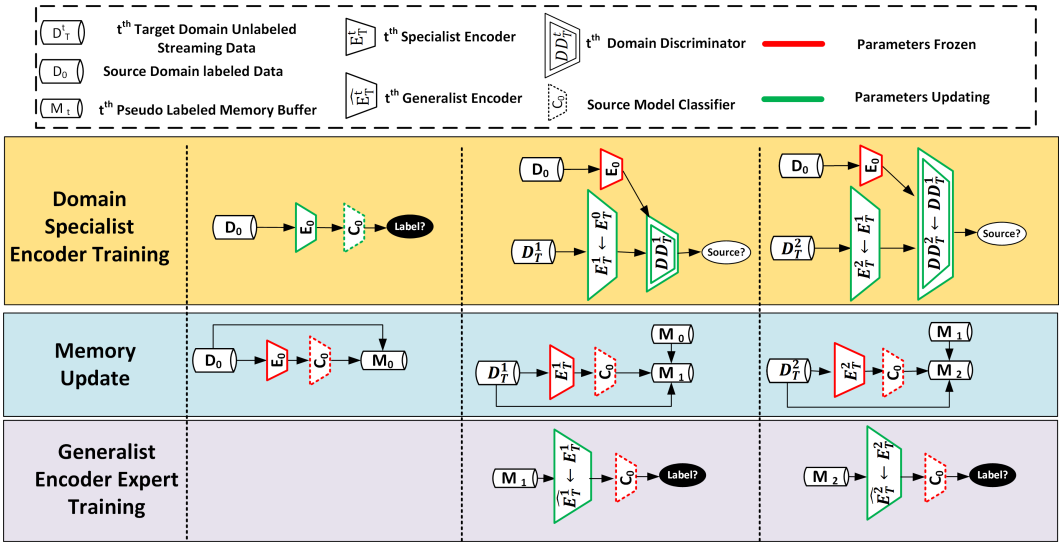


Figure 1: The proposed F2L disjoint (adaptation-distillation) training

As in Bobu et al. (2018), the training starts with training the classifier C_0 and source specialist encoder E_0 on the labeled source data $(X_0, Y_0) \sim p_{D_0}(x, y)$. Once C_0 is trained, it is frozen and used as it is. For next adaptation steps, and as in Wulfmeier et al. (2018), the domain-specialist encoder is trained with only plasticity in mind; thus, such an encoder has a high capability of adapting to infinite adaptation steps. The training of such an encoder is equivalent to setting λ to zero in Eq. 3.This Leads to the objective in Eq. 5.

$$E_T^{t^*} = \mathrm{argmin}_{E_T^t}\left\{d\left[\mathrm{E}_0\left(\mathrm{X}_0\right), E_T^t\left(X_t^t\right)\right]\right\} \tag{5}$$

For the distance metric d, and following Wulfmeier et al. (2018), we use Tzeng et al. (2017) ADDA approach to calculate and minimize GAN loss. However, fundamentally different traditional unsupervised domain adaptation UDA can be utilized (e.g., MMD Tzeng et al. (2014), Jensen–Shannon

[1]The notation and gradient shown here are slightly different from the original paper due to difference in the memory buffer construction. Also, in the original paper, there is an extra constraint called Source Discriminative Constraint. Such constrain is omitted for this discussion for brevity.

divergenceShui et al. (2020), Wasserstein distanceDamodaran et al. (2018) or CORAL distanceSun & Saenko (2016) among others Kashyap et al. (2020)).

Once the domain specialist encoder is trained, it is used in the memory update step.In the memory update step, a representative subset of the training data (X_p^{t}) of the current domain along with their crosponding pseudo labels are stored in the memory buffer (M_t).

$$M_{\mathrm{t}} \leftarrow \left\{ M_{\mathrm{t}-1} \cup \left(X_p^{\mathrm{t}}, Y_p^{\mathrm{t}} \right) \right\} \tag{6}$$

In the present study, we randomly select a fixed-size subset of K samples from the current target Domain $X_p^t \overset{K}{\sim} p_{D_T^t}(x)$. Afterward, the pseudo-labels (Y_p^{t}) are obtained with the trained specialist.

$$Y_p^{\mathrm{t}} = \operatorname{argmax} \left[C_0 \left(E_T^{\mathrm{t}^*} \left(X_p^{\mathrm{t}} \right) \right) \right], . \tag{7}$$

K is the memory buffer size per domain and is determined according to the memory budget allowed. However we use random sampling in this paper we use random sampling, more advanced selection (i.e., coreset) techniques can be utilized to optimize the memory budget utilization Guo et al. (2022).

After the memory update, the generalist encoder is trained on all the samples in the updated memory as follows:

$$\widehat{E_T^{\mathrm{t}}}^* = \operatorname{argmin} \left(\mathrm{CE} \left[C_0 \left(\widehat{E_T^{\mathrm{t}}} \left(X_p^{\mathrm{t}} \right) \right), Y_p^{\mathrm{t}} \right] \right), \tag{8}$$

where $(X_p^t, Y_p^t) \in M_t$ and $\widehat{C_0}$ is the pre-trained and frozen source classifier. Equation 8 can be considered as a knowledge distillation step where the current generalist ($\widehat{E_T^t}$) is a student of multiple teachers (i.e., all the previous trained domain specialist encoders $\left[E_T^{0^*}, E_T^{1^*}, \ldots, E_T^{\mathrm{t}^*} \right]$). The distillation loss used in this study is the usual cross-entropy, as indicated in Eq. 8 It is worth noticing that Eq. 8 is the second term in Eq. 3 without the trade-off parameter λ.

Figure 1 illustrates the overall F2L training procedures and Figure 7 in the Appendix illustrates a side-by-side comparison with our proposed approach and Bobu et al. (2018) approach. Algorithm 1 summarizes the entire training procedure of our proposed approach.

Since the Generalist model in F2L is trained with pseudo-labeled samples from all previous domains, stability is no longer an issue. The only issue that should be taken into consideration is the quality of pseudo labels. However, since the pseudo labels in the current approach are generated from the domain-specialist model (i.e., the high plasticity encoder), the quality of the pseudo is expected to be at its highest. It is critical to notice that pseudo labels' quality depends not only on the model's plasticity but also on the traditional UDA's success in line 9 Alg. 1. Accordingly, the success of F2L or any rehearsal-based CUDA is conditioned on defining the right distance metric in Eq. 5 and the ability to minimize it.

There is no weight sharing between E_0 and E_T^t. DD_T^{t-1}, E_T^{t-1} and E_T^{t} are used to initialize DD_T^t, E_T^t and $\widehat{E_T^{\mathrm{t}}}$ respectively. DD_T^t is the domain discriminator that is used for calculating and minimizing the distance (d) in Equation 5 as proposed in Tzeng et al. (2017).

3 Related Work

3.1 Continuous Learning

Continuous learning is concerned with designing training techniques that allow the model to learn continuously from new experiences without forgetting previous experiences. These training techniques can be categorized into (1) rehearsal, (2) regularization (3) parameters isolation/expanding-based techniques Kemker et al. (2018). However, these techniques are designed for supervised settings, mostly class incremental settings van de Ven & Tolias (2019), and thus cannot be directly applied to the CUDA problem.

3.2 Knowledge Distillation

Knowledge distillation deals with transferring knowledge from one model (i.e., teacher), to another model (i.e., student). Such transfer could be desired for model compression in cases where the

Algorithm 1 : illustrates the complete training procedures for (N) adaptation steps.

Randomly Initialize E_0 and C_0
$(X_0, Y_0) \sim p_{D_0}(x, y)$
Train E_0, C_0 using cross-entropy loss $\{C_0\left[\left(E_0\left(X_{0,}\right)\right], Y_0\right\}$
Update the Memory. $M_0 = \left\{\left(X_p^0, Y_p^0\right)\right\}$ where $Y_p^0 = \operatorname{argmax}\left\{C_0\left[\left(E_0\left(X_0\right)\right]\right]\right\}$
For t=1→N
    $X_T^t \sim p_{D_T^t}(x)$
    Initialize $E_T^{\mathrm{t}} \leftarrow E_T^{\mathrm{t}-1}$
    Initialize $DD_T^{\mathrm{t}} \leftarrow DD_T^{\mathrm{t}-1}$
    Train E_T^t using the discriminator discrepancy loss $d[E_0(X_0), E_T^t(X_t^t)]$
    $X_T^t \overset{K}{\sim} p_{D_T^t}(x)$
    Update the pseudo bank. $M_{\mathrm{t}} = \{M_{t-1} \cup (X_t^{\mathrm{t}}, Y_T^{\mathrm{t}})\}$ where $Y_T^{\mathrm{t}} = \operatorname{argmax}\left\{C_0\left[E_T^{\mathrm{t}}\left(X_t^{\mathrm{t}}\right)\right]\right\}$
    $(X_p^{\mathrm{t}}, Y_p^{\mathrm{t}}) \in M_{\mathrm{t}}$
    Initialize $\widehat{E_T^{\mathrm{t}}} \leftarrow E_T^{\mathrm{t}}$
    Train $\widehat{E_T^{\mathrm{t}}}$ using cross-entropy loss $\mathrm{CE}\left[C_0\left(\widehat{E_T^{\mathrm{t}}}\left(X_p^{\mathrm{t}}\right)\right), Y_p^{\mathrm{t}}\right]$
End For

student model is smaller in capacity. It also could be for improving generalization Furlanello et al. (2018) if the models were of equivalent capacity, or model interpretability or explainability. These techniques require the availability of labeled training data on which the teacher model was trained. However, in F2L, such data is not available. Instead, unsupervised domain adaptation is used to train the teacher, and then the teacher distills its knowledge to the student.

4 Numerical Studies

4.1 Rotating MNIST Data Set

To validate the proposed approach's effectiveness, we tested it on the Rotated MNIST dataset (RotMNIST). In this experiment, Rotated MNIST consisted of seven domains, $D_0, D_1, \ldots . D_6$. D_0 is the only labeled source domain, and it contains digits rotated uniformly between 0 and 45 degrees. Each consecutive domain has an increment of 45 digits rotation angle (e.g D_1 contains digits rotated uniformly between 45 and 90 degrees). For domain alignment, 10,000 images are randomly sampled and rotated from the original MNIST training data set for each domain. The same is done for the testing dataset. For rehearsal, 5000 samples are randomly sampled per domain and stored in the memory buffer along with their corresponding pseudo labels. The same data configuration is used for the other SOTA methods. Figure 2 shows the overall performance of F2L and Bobu et al. (2018) approaches after each incremental adaptation step. Figure 2a shows the classification test accuracy from the domain specialist encoder, and Figure 2b shows the ones from the generalist model. Each row presents the performance of the i^{th} Model (i.e., after adapting to the i^{th} Domain) when tested on past ([0 →t-1]), present (t) and future ([t+1 →6]) domains. The values on the diagonal represent the accuracy of a domain after adapting to it, which is a good indicator of the model's plasticity. On the other hand, the values in the lower triangle under the diagonal show the accuracy of the current model in previously seen domains and are a good measure for memory retention. In general, Figure 2 shows the effectiveness of the F2L approach. Figure 2a shows the encoder specialist's good plasticity as evident from consistently high accuracy score on the diagonal. However, since, as described in Eq. 5, the specialist encoder training does not take into consideration memory retention, the previous experience is very volatile and significantly forgotten after a few adaptation steps. For example, observe how the accuracy on the D_1 column drops significantly below row M_1. On the other hand, the generalist encoder in Figure 2b[2] shows successful knowledge transfer between the generalist and specialist encoders. To show the superiority of F2L, Figure (2c- 2e) show the performance of Bobu et al. (2018) using the exact baseline model and memory resources with three different λ values. At λ=0.01, little emphasis is given to the anti-forgetting term in Eq. 3, so the model shows good plasticity, as evident from the high accuracy values on the diagonal. However, At

[2]In F2L, and during inference/testing, only the performance of the generalist is relevant.

a slightly higher value of λ (i.e., λ=0.1), a higher priority is given for the replay loss, significantly improving knowledge retention, as evident from the values under the diagonal, but at the expense of model plasticity. For the extreme case of λ=10, the model entirely loses its plasticity and does not acquire any new knowledge after the second adaptation step.

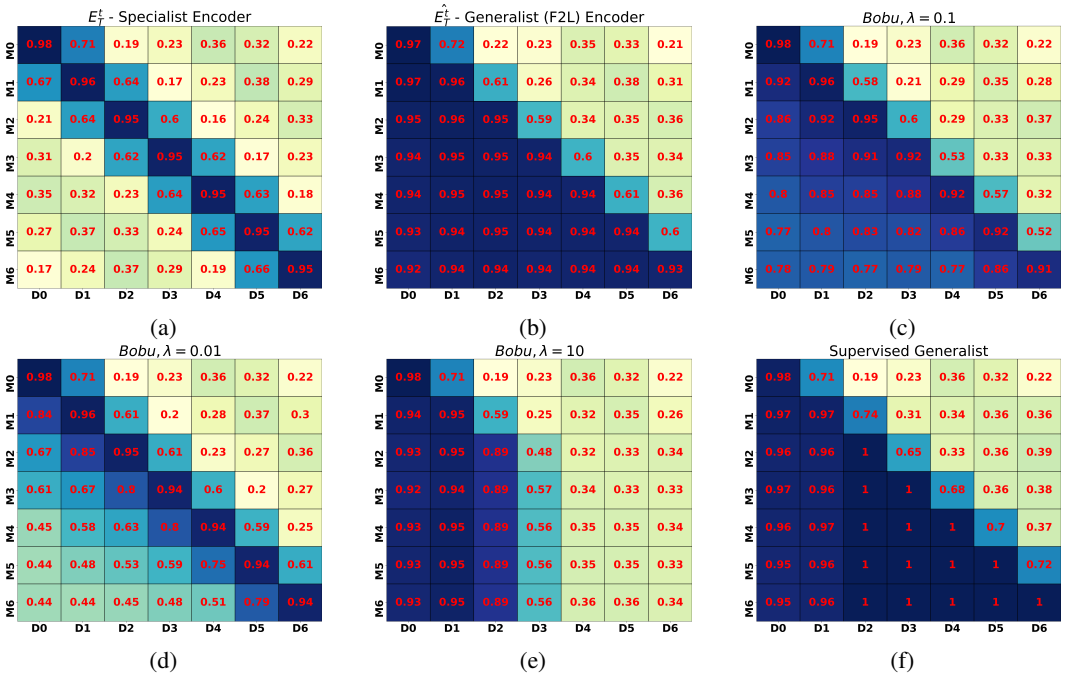


Figure 2: Performance heat map after each incremental adaptation step for (a,b): The proposed F2L disjoint (adaptation-distillation) training, (c-e): The joint multi-task training proposed by Bobu et al. (2018). and (f): Supervised (upperbound) generalist. The remaining case of Table 1 are presented in the appendix in Figure 8

We use the following three evaluation metrics to quantify the influence of λ in Bobu et al. (2018). Average Peak Accuracy (APA)

$$APA = \frac{\sum_{t=0}^{N} \mathrm{acc}_{t,t}}{\mathrm{N}} \tag{9}$$

Average Latest Accuracy (ALA)

$$ALA = \frac{\sum_{t=0}^{N} \mathrm{acc}_{N,t}}{\mathrm{N}} \tag{10}$$

Average Historical Accuracy (AHA) (known as Backward transfer BWT in Díaz-Rodríguez et al. (2018))

$$AHA = \frac{2 * \sum_{j=1}^{N} \sum_{t=0}^{j-1} \mathrm{acc}_{j,t}}{\mathrm{N} * (\mathrm{N}+1)} \tag{11}$$

Average Drop in Accuracy (ADiA) (also known as Backward transfer BWT in (Lopez-Paz & Ranzato, 2017))

$$\mathrm{ADiA} = \frac{\sum_{t=0}^{N} \mathrm{acc}_{N,i} - \mathrm{acc}_{t,t}}{\mathrm{N}}, \tag{12}$$

where N is the number of target domains and $acc_{x,y}$ is the accuracy of the model M_x after adapting to D_x when tested on the D_y test dataset. Intuitively, APA is the average of the diagonal in Figure 2, and a high value of APA indicates high model plasticity. AHA is the average lower triangular value of Figure 2, excluding the diagonal. ADiA is the difference between the diagonal values and the last row values. A higher value of AHA or ADiA indicates better stability. Note for the F2L approach, APA and AHA are calculated solely from the generalist accuracies in Figure 2b. However, for the sake of fair comparison, when calculating the ADiA, the diagonal values (i.e., $acc_{i,i}$) are taken from

the specialist Figure 2a , which are higher than the values for the generalist. On the other hand, the last row values are taken from the generalist accuracies in Figure 2b as it is the actual model to be used in deployment.

Figure 3 shows the trade-off between plasticity, measured by APA, and stability, measured by AHA, at different values of λ. The figure clearly shows the realization of elasticity-plasticity dilemma in the multi-task approach. It also shows why approaches like Huang et al. (2021) and Tang et al. (2021) that try to formulate the problem as single objective constrained optimization problems would not fundamentally solve the plasticity-elasticity dilemma. This is because the fundamental reason why the multi-task approach does not work is that the two tasks are conflicting in nature, not because the replay loss is not effective enough to guarantee knowledge retention, as proposed by Tang et al. (2021). Thus any attempt to promote knowledge retention must lead to a different degree of plasticity drop. Muti-task techniques, as well as constrained single-task techniques, at their best hyperparameter, can only help avoid unnecessary sacrifice in stability when the gain in plasticity is insignificantly infinitesimal. This also raises another issue of how to select the best λ values. This issue is solved by F2L since there is no need for this λ hyperparameter.

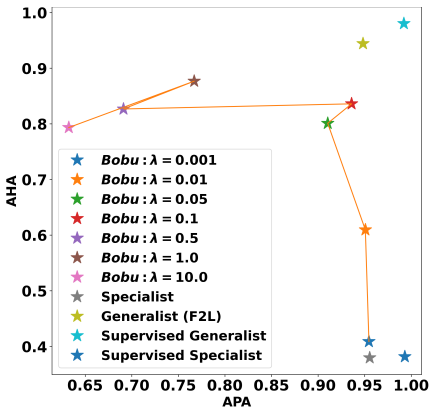


Figure 3: Stability-Plasticity trade-off at different values of λ

The same results in Figure 3 are shown again in Table 1, along with the ADiA. Interestingly, Bobu $\lambda = 10$ is showing a higher ADiA of -0.00613 compared to F2L with only -0.01211. This misleadingly indicates that Bobu $\lambda = 10$ is doing a better job in retaining knowledge as it is showing less drop in accuracy. However, as hinted while discussing Figure 2e earlier, this high stability is achieved at the expense of extremely low plasticity; i.e., the model is not learning any new information after the third adaptation step. This is why Bobu $\lambda = 10$ shows the lowest APA with 0.632429. This also reveals the superiority of AHA as a stability metric relative to the ADiA. Two upper bounds are also presented in Figure 3 and Table1. The supervised specialist is trained in the same manner as the specialist in the F2L framework, but instead of minimizing Eq. 5, it is trained in a supervised manner with access to the current target domain labels. Similarly, the supervised generalist is trained similarly to the generalist, but instead of distilling knowledge using Eq. 8, it is trained in a supervised manner jointly on the current domain, and all historical domains, training labeled data. From Figure3, it is evident that the F2L generalist is the closest to the supervised generalist. On the other side, Bobu $\lambda = 0.1$ achieves the best trade-off for the Bobu et al. (2018) multi-task approach. So far, the memory buffer size per domain was fixed at K=5000. In Appendix A.4, we show that the F2L maintains its superiority at lower memory sizes. Also, it is worth mentioning that although F2L requires training two encoders, the specialist and generalist, the overall training time is smaller, as demonstrated in Appendix A.3.

Table 1: Evaluation metrics on the Rot-MNIST dataset

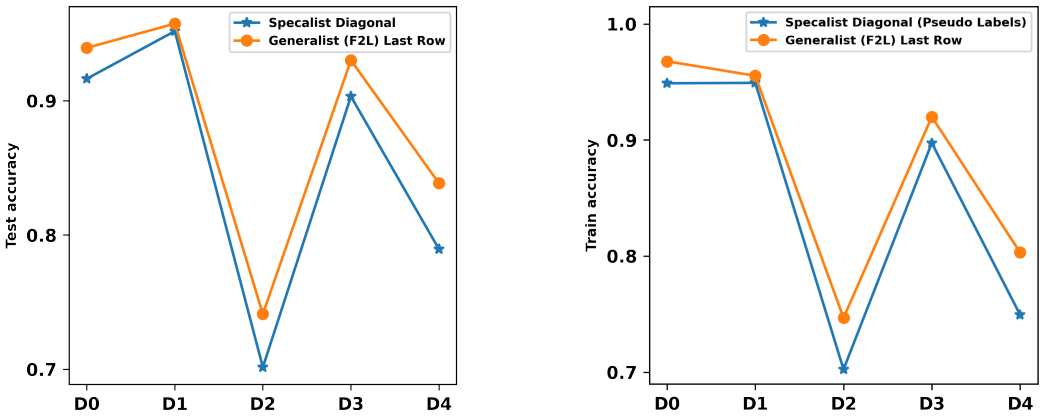
Method	ADiA	APA	AHA	ALA
Bobu λ=0.001	-0.53534	0.954654	0.408581	0.419317
Bobu λ=0.01	-0.37038	0.95082	0.609346	0.580437
Bobu λ=0.05	-0.13351	0.910406	0.800896	0.776891
Bobu λ=0.5	-0.01708	0.691146	0.82663	0.674066
Bobu λ=1	-0.02214	0.767134	0.87666	0.744997
Bobu λ=10	**-0.00613**	0.632429	0.793251	0.626303
Specialist (Wulfmeier et al. (2018))	-0.54485	**0.955474**	0.379436	0.410626
F2L (Generalist)	-0.01211	0.948386	**0.944057**	**0.93628**
Supervised Specialist	-0.556697	0.993126	0.381601	0.436429
Supervised Generalist	-0.004754	0.992234	0.980084	0.98748

4.2 5 DIGITS DATA SET

5 Digits dataset is another domain adaptation dataset consisting of one source domain and four target domains. We follow the same adaptation task solved in Tang et al. (2021), where we start pretraining on the source domain of SYN, then we adapt to MNIST, MNIST-M, USPS, and SVHN sequentially. Similar to Tang et al. (2021), 1024 samples per domain are stored in the memory buffer. Unlike the RotMNIST dataset, this task consists of a shorter sequence and relatively more challenging classification task. Again, Table 2 shows the superiority of F2L both in terms of ADiA and ALA. Noticeably, the F2L was the only method to achieve positive ADiA. To further explain the meaning behind the positive ADiA, Figure 4 compares the specialist peak accuracies (i.e., values on the diagonal of the performance heat map) and the generalist performance after the latest adaptation step (i.e., last row of the performance heat map). The figure shows that the generalist (i.e. the student) can outperform the specialist (i.e. the teacher). This is interesting because, the pseudo labels, used in training the generalist, are the training data labeled using the specialist predictions.

Table 2: Evaluation metrics on the 5-Digit dataset

Method	ADiA	ALA
Bobu et al. (2018)	-0.061	0.8212
Tang et al. (2021)	-0.010	0.8534
Specialist	-0.078	0.7746
Generalist (F2L)	**0.029**	**0.8813**
Supervised Specialist	-0.294	0.6310
Supervised Generalist	-0.003	0.9259



(a) (b)

Figure 4: Performance compassion between the specialist and the generalist on the (a) testing, and (b) training datasets.

This indicates that, instead of just copying the current specialist's knowledge and not forgetting the historical specialists' knowledge, the generalist also scans historical knowledge and selects the relevant ones for the current task. The generalist fuses such selected knowledge with the knowledge distilled from the current specialist. Consequently, the generalist corrected the specialist's miss-labeled pseudo-labeled training data by applying such fused knowledge. This explains why the generalist can outperform its teachers not only on testing but also on the training data on which it was originally trained, as shown in Figure 4. Examples of these cases for D_3 are shown in Figure 5. These are training data points that were given wrong pseudo labels by the specialist and then used to train the generalist. However, after the generalist was trained on such pseudo labels, it was able to classify these mislabeled pseudo labels correctly. Such data points are the reason for the gap between the orange and blue lines in Figure 4. The full performance heat maps of the relevant models are available in the Appendix in Figure 10. Beside testing our approach on Digits dataset we also show the superiority of F2L on office-caltech dataset in Appendix A.6.

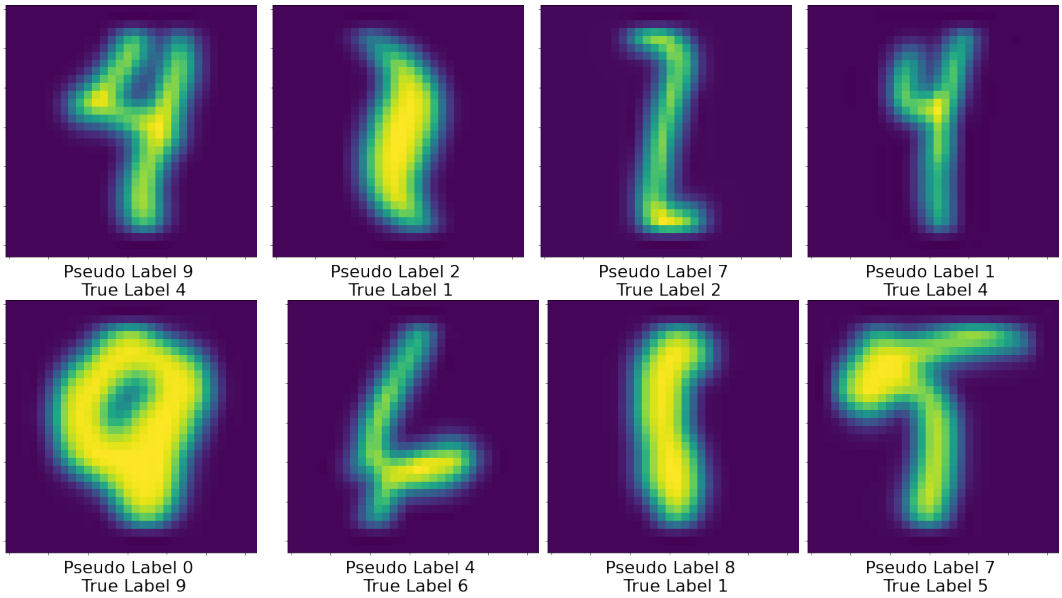


Figure 5: Examples of specialist's wrong predicted pseudo-labels in USPS training data that are correctly classified by the generalist.

5 Conclusion

The present study investigated the stability–plasticity dilemma of CUDA. We have demonstrated that the two current trends do not address such a dilemma adequately. In our approach, Forget to Learn (F2L), instead of attempting to train a single model that can achieve an optimum trade-off between stability and plasticity, we proposed training two models, the specialist and the generalist. The specialist shows high plasticity and transfers their knowledge to the generalist via knowledge distillation. We have tested our framework on different CUDA datasets, and the results indicate the effectiveness of the proposed approach in simultaneously achieving better plasticity and stability. The results on the rotated MNIST dataset indicate that no hyperparameter values can make the multi-task approach superior to F2L. The F2L achieved ADiA of -0.01211 and APA of 0.948386, while the best competing SOTA achieved ADiA -0.12548 and APA of 0.93606 at λ= 0.1. On Digit 5 Dataset, F2L approach continued to show superior performance compared to other SOTA CUDA methods. F2L was the only method showing a positive ADiA of 0.029, whereas the second highest SOTA showed a value of -0.010.

REFERENCES

Andreea Bobu, Eric Tzeng, Judy Hoffman, and Trevor Darrell. Adapting to continuously shifting domains. *6th International Conference on Learning Representations, ICLR 2018 - Workshop Track Proceedings*, (Icml), 2018.

Bharath Bhushan Damodaran, Benjamin Kellenberger, Rémi Flamary, Devis Tuia, and Nicolas Courty. Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 447–463, 2018.

Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (c):1–29, 2021. ISSN 19393539. doi: 10.1109/TPAMI.2021.3057446.

Natalia Díaz-Rodríguez, Vincenzo Lomonaco, David Filliat, and Davide Maltoni. Don't forget, there is more than forgetting: new metrics for Continual Learning. 2018. URL http://arxiv.org/abs/1810.13166.

Tommaso Furlanello, Zachary C. Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born-Again Neural Networks. *35th International Conference on Machine Learning, ICML 2018*, 4:2615–2624, 2018.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *32nd International Conference on Machine Learning, ICML 2015*, 2(i):1180–1189, 2015.

Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning. *arXiv preprint arXiv:2204.08499*, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Judy Hoffman, Trevor Darrell, and Kate Saenko. Continuous Manifold Based Adaptation for Evolving Visual Domains. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 867–874. IEEE, jun 2014. ISBN 978-1-4799-5118-5. doi: 10.1109/CVPR.2014.116. URL https://ieeexplore.ieee.org/document/6909511.

Zhipeng Huang, Zhizheng Zhang, Cuiling Lan, Wenjun Zeng, Peng Chu, Quanzeng You, Jiang Wang, Zicheng Liu, and Zheng-jun Zha. Lifelong Unsupervised Domain Adaptive Person Re-identification with Coordinated Anti-forgetting and Adaptation. (December), 2021. URL http://arxiv.org/abs/2112.06632.

Abhinav Ramesh Kashyap, Devamanyu Hazarika, Min-Yen Kan, and Roger Zimmermann. Domain divergences: a survey and empirical analysis. *arXiv preprint arXiv:2010.12198*, 2020.

Ronald Kemker, Marc McClure, Angelina Abitino, Tyler L. Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pp. 3390–3398, 2018.

Joonhyuk Kim, Sahng-Min Yoo, Gyeong-Moon Park, and Jong-Hwan Kim. Continual Unsupervised Domain Adaptation with Adversarial Learning. 2020. URL http://arxiv.org/abs/2010.09236.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in Neural Information Processing Systems*, 2017-Decem(Nips):6468–6477, 2017. ISSN 10495258.

Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in Psychology*, 4, 2013. ISSN 1664-1078. doi: 10.3389/fpsyg.2013.00504. URL http://journal.frontiersin.org/article/10.3389/fpsyg.2013.00504/abstract.

Sayan Rakshit, Anwesh Mohanty, Ruchika Chavhan, Biplab Banerjee, Gemma Roig, and Subhasis Chaudhuri. FRIDA — Generative feature replay for incremental domain adaptation. *Computer Vision and Image Understanding*, 217(January):103367, 2022. ISSN 1090235X. doi: 10.1016/j.cviu.2022.103367. URL https://doi.org/10.1016/j.cviu.2022.103367.

Mohammad Rostami. Lifelong Domain Adaptation via Consolidated Internal Distribution. *Advances in Neural Information Processing Systems*, 14(NeurIPS):11172–11183, 2021. ISSN 10495258.

Mark Schutera, Frank M. Hafner, Jochen Abhau, Veit Hagenmeyer, Ralf Mikut, and Markus Reischl. Cuepervision: self-supervised learning for continuous domain adaptation without catastrophic forgetting. *Image and Vision Computing*, 106: 104079, feb 2021. ISSN 02628856. doi: 10.1016/j.imavis.2020.104079. URL https://doi.org/10.1016/j.imavis.2020.104079https://linkinghub.elsevier.com/retrieve/pii/S0262885620302110.

Changjian Shui, Qi Chen, Jun Wen, Fan Zhou, Christian Gagné, and Boyu Wang. Beyond h-divergence: Domain adaptation theory with jensen-shannon divergence. *ArXiv*, abs/2007.15567, 2020.

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pp. 443–450. Springer, 2016.

Shixiang Tang, Dapeng Chen, Jinguo Zhu, Shijie Yu, and Wanli Ouyang. Layerwise Optimization by Gradient Decomposition for Continual Learning. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 9629–9638, 2021. ISSN 10636919. doi: 10.1109/CVPR46437.2021.00951.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:2962–2971, 2017. doi: 10.1109/CVPR.2017.316.

Gido M. van de Ven and Andreas S. Tolias. Three scenarios for continual learning. pp. 1–18, 2019. URL http://arxiv.org/abs/1904.07734.

Zeya Wang, Baoyu Jing, Yang Ni, Nanqing Dong, Pengtao Xie, and Eric Xing. Adversarial domain adaptation being aware of class relationships. *Frontiers in Artificial Intelligence and Applications*, 325:1579–1586, 2020. ISSN 09226389. doi: 10.3233/FAIA200267.

Markus Wulfmeier, Alex Bewley, and Ingmar Posner. Incremental adversarial domain adaptation for continually changing environments. *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 4489–4495, 2018. ISSN 10504729. doi: 10.1109/ICRA.2018.8460982.

Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. *arXiv preprint arXiv:2106.01085*, 2021.

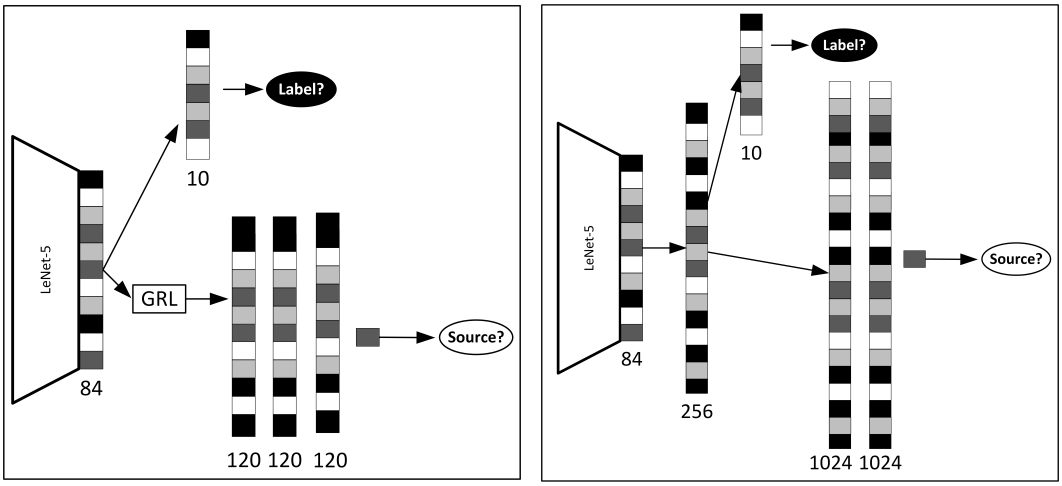
A APPENDIX

A.1 THE IMPLEMENTATION DETAILS

The proposed approach in section 2 is general and can be implemented in different configurations. Fundamentally different choices for the distance (d) in Eq. 8 can be used. As a matter of fact, any traditional domain adaptation algorithm can be used for the evaluation and minimization of (d). This includes the contrastive loss proposed in Tang et al. (2021). In the present study, and for all experiments, ADDA from Tzeng et al. (2017) was utilized. It is GAN inspired adversarial domain adaptation approach where a discriminator is trained to align the source and target distribution by fooling the feature extractor.

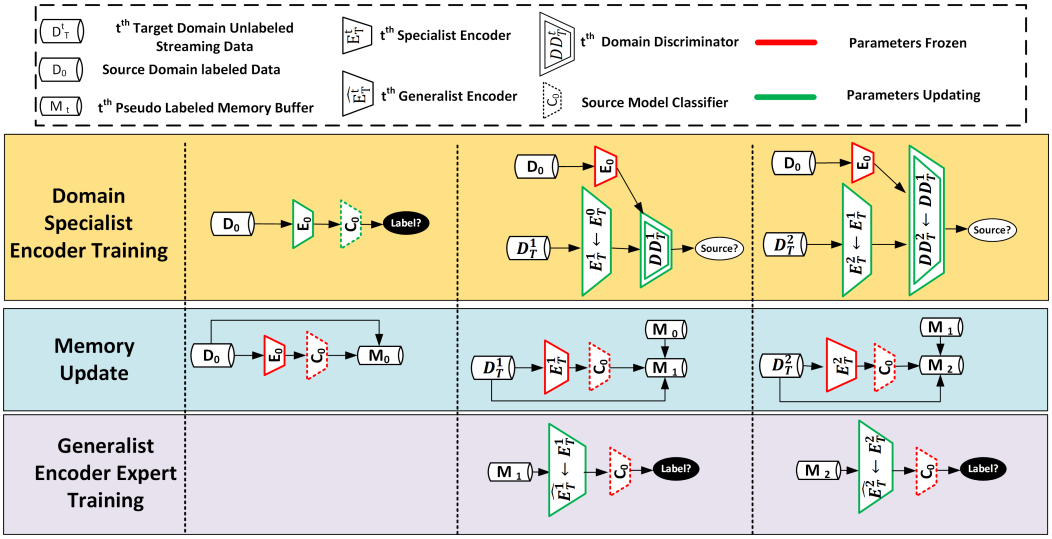
For the RotMNIST dataset, LeNet-5 Lecun et al. (1998) was used as the backbone (i.e., generalist and specialist encoder). A discriminator with three hidden layers, each with 120 neurons, was used right after the feature extractor of LeNet-5, as shown in Figure 6a. A batch size of 512 and a constant learning rate of 1^{-4} were used. A GRL reversal layer with a scaling factor of 1 was used for in the two-player zero-sum game training of the discriminator and encoder Ganin & Lempitsky (2015)

In the case of the 5 Digit dataset, following Tang et al. (2021), LeNet-5 was also used as the backbone, with an additional projection linear layer of 256 neurons. The output of the linear layer was projected to 10 output nodes for the classifier. Also, the output of the projection layers was used as an input to the discriminator, as shown in Figure 6b. The discriminator is 2-layer FCNs, each with 1024 neurons, batch normalization, and ReLU activation. A batch size of 512 with an initial learning rate of 0.01 with a decay factor and gamma of 0.75 and 0.001, respectively, were used.

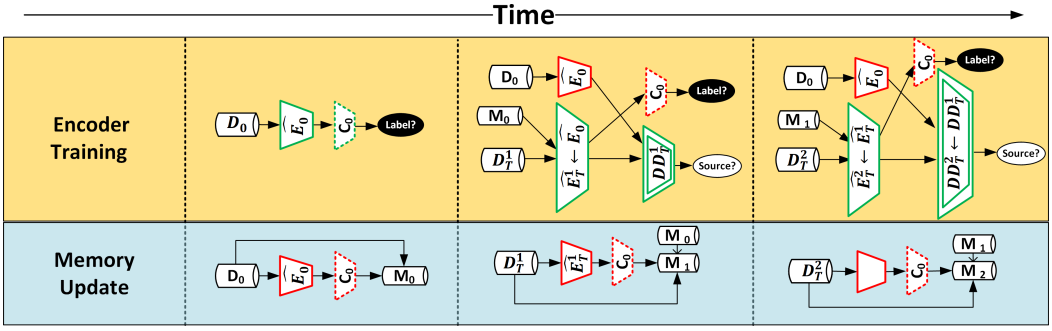


(a) (b)

Figure 6: The used architecture for (a) RotMNist and, (b) 5 Digit, datasets



(a)



(b)

Figure 7: (a) The proposed F2L disjoint (adaptation-distillation) training, (b) the joint multi-task training proposed by Bobu et al. (2018).

A.2 More Results From RotMNIST Dataset

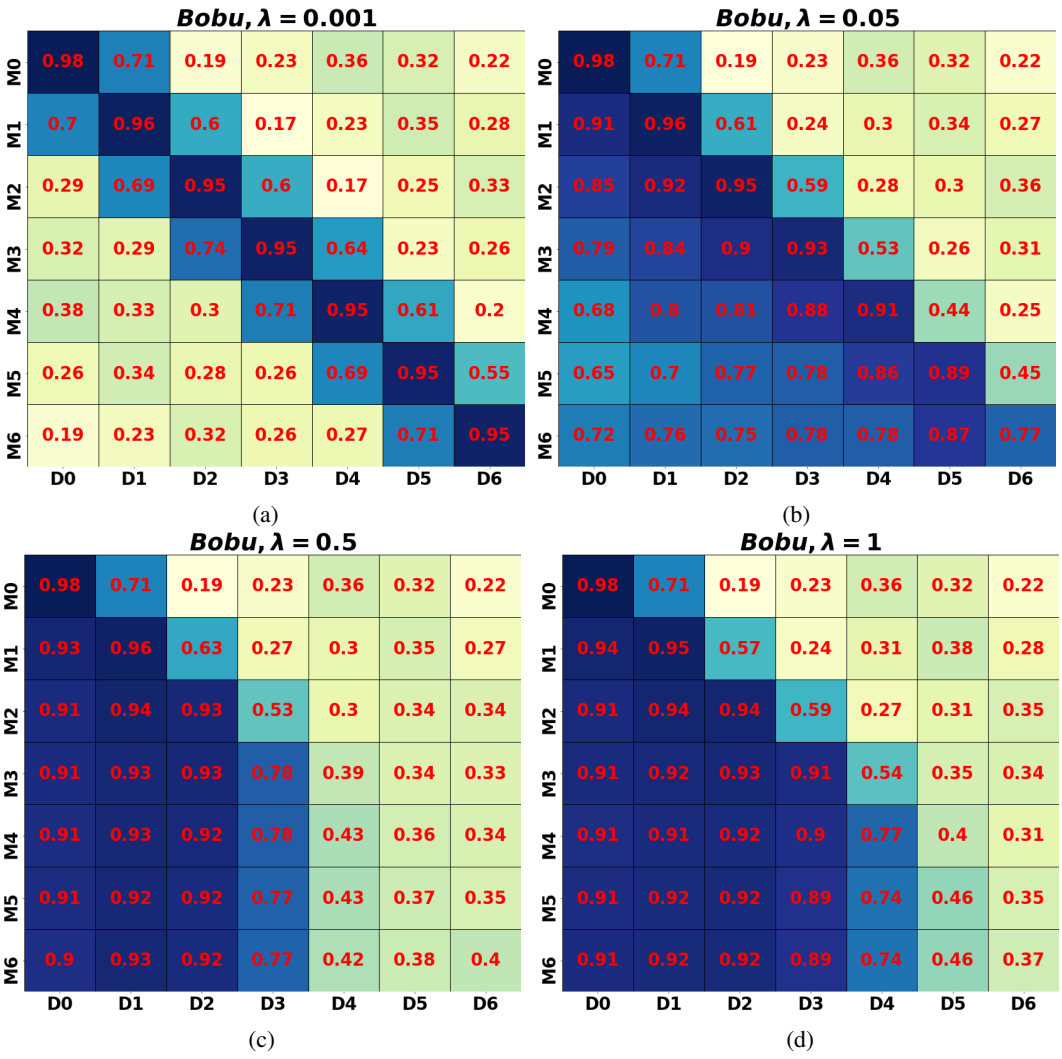


Figure 8: continuation of Figure2 at different values of λ

A.3 CONVERGENCE AND COMPUTATION REQUIREMENTS

Although F2L involves training two encoders of equal capacity, this section shows that this does not increase computational requirement, even reduces it. F2L (1): converges in fewer total number epochse[3], and (2): the number of optimizer update steps (i.e., number of gradient calculations) per epoch is less.

For the first point, training the F2L (i.e., both Generalist and Specialist) took 72% fewer epochs to converge[4] than Bobu. This faster convergence is due to the simpler, non-conflicting, single objective function used for each stage of F2L.

For the second point, only the source and current target domain data are needed when training the specialist, which accounts for 76% of the training time in the F2L approach. Accordingly, 76% of the time, there is no need to feed forward/backward the samples from the memory buffer to calculate the loss/gradient of Equation 5. Accordingly, for the same batch size, a single epoch will involve fewer gradient calculations, and update steps. Note that for Bobu and other methods, all data in the memory buffer, source, and current target data must be processed during every optimizer step, see Equation 3.

[3]For Specialist, 1 Epoch= Processing all the source and current target data once. For Generalist, 1 Epoch= Processing all the source, current target, and memory buffer data once For Bobu. 1 Epoch= Processing all the source, current target, and memory buffer data once

[4]Convergence is assumed when the accuracy shows non-increasing values for ten consecutive epochs.

A.4 MEMORY REQUIREMENTS

To further investigate the memory requirement of our method, in 9, we show the influence of limiting the memory buffer size on both F2L and Bobu et al. (2018). Both approaches show less stability by lowering the buffer size. However, overall, F2L still maintains its superiority. At 500 samples per domain, F2L and Bobu show AHA of 0.835 and 0.764, respectively. It is worth mentioning that, at a memory-free setup, both Bobu and F2L converge to the F2L specialist (i.e., Wulfmeier et al. (2018)), which has an AHA of 0.379.

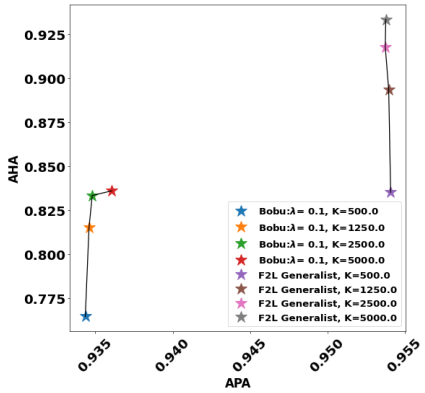


Figure 9: Effect of memory buffer size per domain on F2L and Bobu

Another critical aspect of memory requirement is the memory needed to store the model-trained para maters. Inspired by Wulfmeier et al. (2018), in Algorithm 1 line 7, we use the specialist from the last time step E_T^{t-1} to initialize the new specialist. This necessitates storing the specialist in the memory buffer, which might be memory-demanding, especially for large models. Another alternative is to use the most recent generalist for specialist initialization purposes. In other words, line 7 in Algorithm 1, becomes Initialize $E_T^{\mathrm{t}} \leftarrow \widehat{E_T^{\mathrm{t}-1}}$. This is useful because $\widehat{E_T^t}$ is already stored in the buffer to be used during inference. Thus no extra memory is required for specialist initialization. Table 3 indicates that the alternative initialization shows an insignificant effect on the overall performance.

Table 3: Evaluation metrics on the Rot-MNIST dataset with alternative initialization Strategy

Method	ADiA	APA	AHA	ALA
Specialist	-0.54485	0.955474	0.379436	0.410626
Generalist	-0.01211	0.948386	0.944057	0.93628
Specialist (Alternative Initialization)	-0.522097	0.951143	0.394990	0.429046
Generalist (Alternative Initialization)	-0.018314	0.952186	0.935750	0.933871

A.5 More Results From 5 Digit Dataset

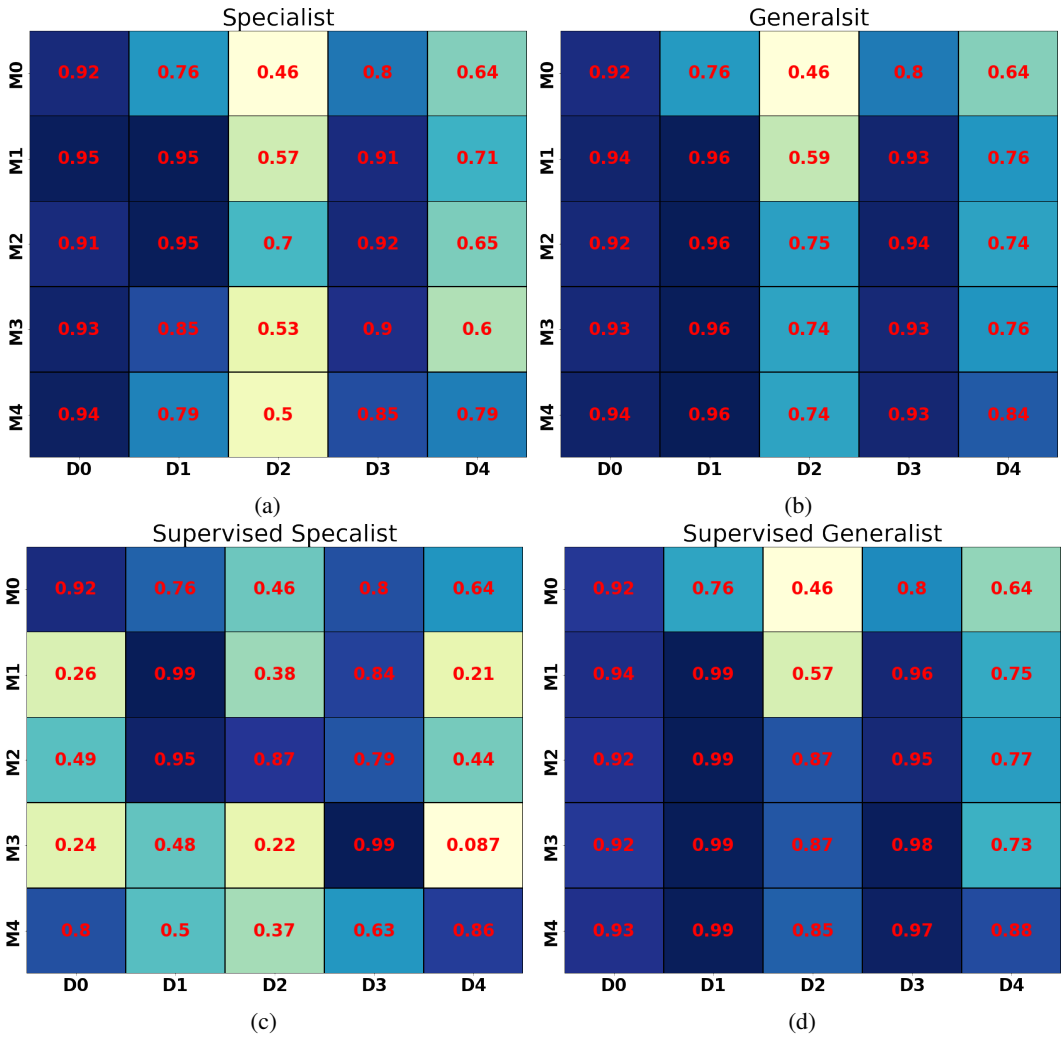


Figure 10: Performance hear map on Digit 5 Dataset of the (a) and (b) specialist and generalist of F2L (c) supervised specialist and (d) supervised generalist, where SYN, MNIST, MNIST-M, USPS, and SVHN are D_0, D_1, D_2, D_3 and D_4 respectively.

A.6 OFFICE-CALTECH DATASET

RotMnist and 5-Digit Dataset both share the same input dimensionality. This section shows the same previous test procedures on the 224*224 Office-Catech Dataset. Similar to the 5-Digit dataset, Office-Caltech consists of three discrete DA steps. We set the source domain to be DSLR, and we adapt to Amazon, Webcam then Caltech targets sequentially. Table 4 shows the evaluation metrics for the Office-Caltech dataset. F2L still shows the best performance. The performance of the other approaches shows significantly lower performance. It is worth mentioning that, due to the large similarity between the domains, a model trained on the source domain only in this task can achieve up to 0.9125 average accuracy over all domains, which is significantly higher than the values reported by Tang et al. (2021). Furthermore, it is noticeable that the Generalist in (F2L) is showing a positive ADiA with a higher value for the F2L Generalist of 0.0087. For this dataset, following Tang et al. (2021), ResNet-18 He et al. (2016) was used as a backbone structure. The same projection head, discriminator, and training details described in section A.1 for the 5 Digit dataset were utilized.

Table 4: Evaluation metrics on the Office-Caltech dataset

Method	ADiA	ALA
Bobu et al. (2018)	-0.0465	0.8483
Tang et al. (2021)	0.0005	0.8723
Specialist	-0.020	0.938
Generalist (F2L)	0.0087	0.952