RESEARCH ARTICLE OPEN ACCESS

Transferability Enhanced Deep Neural Networks through Unsupervised Domain Adaptation

Karunarathna Chandrathilaka Appuhamilage Asiri Iroshan*, Qing Tian**

*(School of Computer & Software, Nanjing University of Information Science & Technology, 210044, Nanjing, China Email:asiriiroshan@hotmail.com)

**(School of Computer & Software, Nanjing University of Information Science & Technology, 210044, Nanjing, China Email:tianqing@nuist.edu.cn)

----------**********----------

Abstract:

Unsupervised Domain Adaptation (UDA) is the utilization of a model that is trained on data from a source domain, on a target domain via adaptation without using labelled data in the target domain. The knowledge learned from a source with labelled data is transferred to a target domain where there is no labelled data available.The processof UDA involves model training on source domaindata enabling the learning of underlyingpatterns and relationships in the train data set. The trained model is then adopted the targetdomain by minimizing the discrepancy between the source and target domain. The discrepancycan be measured by using Domain Adversarial Training or Domain Alignment methods. Thisresearch provides a review of a series of research works related to UDA and provides athorough understanding of Transferability Enhanced UDA Networks. The research drawsinsights from the reviewed existing literature and conducts experiments and analysis onMNIST and MINST-M dataset where the model is trained on MNIST dataset and adapted tounlabelled MNIST-M dataset. The research is primarily based on the work of Ganin et al., 2016,where a novel representation learning approach is presented for Domain Adaptation (DA), theexperiments are conducted and results are produced. The approach proposed has drawn insightsfrom the theory of DA which suggests that in order to achieve domain transfer effectively, thefeatures that are incapable of discriminating between source and target domains should be usedas the basis for predictions. This idea is implemented in the context of neural networks whichare trained on labelled source domain data and unlabelled target domain data. To enhance theaccuracy of the results and to ensure that the domain confusion and class loss in the labelpredictor are maximized and minimized respectively, $-\lambda$ factor is applied to the gradient duringthe backward pass. Wepropose a function to increase this λ factor gradually from 0 to 1 to ensure that thedomain confusion is not backpropagated to the CNN layers in the beginning. The idea is toignore the domain loss in the beginning as it is going to be noisy because it is based on badconvolutional features. During the progress of training, the features that are discriminative forthe main learning task on the source domain emerge along with the features that areindiscriminative with regards to the domain shift. The experiments conducted has resulted inexcellent performance in adapting the model trained on MNIST dataset to predict on MNIST-M dataset.

Keywords **—Unsupervised Domain Adaptation, Domain Adaptation, Deep Neural Networks, Convolutional Neural Networks.**

----------**********----------

I. INTRODUCTION

Deep learning is incredibly versatile and has good performance in various computer vision tasks. As a result, deep learning has gained a lot of popularity in recent years. Deep networks, however, have a lot of different parameters. Massive networks pose computational difficulties and make it impossible for humans to understand the reasoning behind the networks' judgments. The model must be able to recognize and react appropriately to numerous environmental changes in order to allow networks to be deployed to delicate jobs like autonomous driving.Network performance may suffer when making inferences about a target domain because they are trained using data from the source domain (deployment). In order to ensure stable behaviour, domain adaptation (DA) involves adjusting networks to novel settings. DA focuses on agile methodologies that use data from a source domain to produce models with excellent performance on a target domain rather than creating universal models that should function in any situation. By not requiring annotations from the target domain, unsupervised domain adaptation (UDA) goes beyond conventional adaptation. Alternatively, by using unlabelled data from the target domain, the adaptation is made possible.The target (environment) where the approach is to be applied in real-world applications differs from the source (annotated data) utilized for training. Moreover, labelled data from the deployable distribution is typically unavailable. DA techniques have gained favour in applications where labelling is expensive, and sample quality significantly relies on the hardware configuration such as camera setup and viewing angle. Due to the laborious nature of the annotation process and the task's steadfast reliance on the hardware and positioning of the camera, UDA has given semantic segmentation a great deal of attention. For instance, labelling the pixels of a single Cityscapes image takes on average ninety minutes, at which rate it would take three person years to annotate the whole dataset [1].

Over the years, UDA has grown significantly, although it is still only used in a few specific situations. Real-world data are so varied that even humans and machine learning models struggle to adapt to well-known tasks when faced with novel situations. Performance improvement in a defined and static target domain is the fundamental objective of current UDA approaches. But in the actual world, tasks frequently experience several domain transitions. To create an accurate model, a conventional supervised machine learner would need a lot of labelled data. Large volumes of training data must be manually annotated, which takes a lot of effort and is impractical. Furthermore, there is no assurance that the annotated data is impartially representing all of the data[2], [3]. For instance, due to the disparities in cost of living, statistics gathered from a local market in Colombo may be biased when compared to marketplaces around the world.The data from the Sri Lankan market might be used to assist a model generalize to the Indian market instead of having to generalize to the entire global market. When using supervised learning, the available data is often divided into two sets: a training set for building a model and a test set for assessing how well the model works. The model makes an effort to reduce the training error during training.The generalization error on the test set, the target set that was not observed during training, is what we are genuinely concerned with minimizing. Usually, it is assumed that the distributions of the training and test sets are independent and identical.A feature space and its probability distribution construct a domain. Different to that, a task is made up of a label space and a decision function to teach the test set the desired probability distribution[4]. The training set is used in conventional supervised learning techniques to develop an independent model, which is then tested against data from each target domain for each target task.

For instance, if there is a domain X and we want to train a model M_X for a task T_0 in the said domain X, we must provide labelled data for the task T and domain X. If the given data is from a different task T_1 or comes from a different domain Y, we want to train a model M_Y to have very good performance on domain Y. When there is no sufficient labelled data for the target task or domain, in this case, if sufficient labelled data for target domain Y does not

exist, the supervised models will be unsuccessful at learning a model that is accurate on the target domain.

When it comes to Transfer Learning, there are notable differences when compared to Traditional Supervised Learning paradigms. In transfer learning, knowledge is transferred from a task or a domain that is related to the target task or domain to obtain a trained model that works accurately on the target task or the domain[5]. While the source and target domains in transfer learning can differ, they should have some things in common. The process can be carried out in different stages, such as feature extraction, fine-tuning, or complete retraining.Transfer learning and supervised DA differ primarily in that transfer learning can be carried out even in the absence of labelled data in the target domain, whereas supervised DA necessitates labelled data in the target domain. The pre-trained model serves as a feature extractor in transfer learning, whereas in supervised DA, the model is adjusted on the target domain to fit its properties. There are two subfields in transfer learning; Multi Task Learning (MTL), DA.

In MTL, the transfer happens across the tasks where as in DA, the transfer happens across the domains. The goal of MTL is to train a model for a task by simultaneously learning several related tasks[6]. Learning tasks simultaneously has proven to improve the performance significantly more than when each and every task is trained independently in cases where the availability of data per each task is limited. Through MTL, the available data could be utilized across multiple tasks that are related to each other [6]. MTL is also called as learning with auxiliary tasks as defined in [7]. DA deals with the problem of adapting models trained on a source domain to perform well on a target domain that is related but has different characteristics to that of the source domain. Through DA, the amount of labelled data needed to train a model to either perform well on a new domain, or to improve its performance on a new domain, is expected to be reduced. In DA, both the source and target domains have the same task. In order to bring the two domains closer together and make it simpler to transition from the source domain to the target domain, DA aims to identify a common space representation for the two domains.

This paper aims to evaluate a set of previous work with regards to different types of UDA solutions and present a review of those work and conducting experimentation based on the previous work. Accordingly, an UDA Task of adapting a model trained on MNIST labelled source dataset to MNIST-M target unlabeled dataset is performed based on the work of[8]and the results are provided. The paper introduces a gradual lambda reversing functionality to the gradient reversal layer of the model proposed in [8].

The Gradient Reversal Layer of model proposed in [8] is utilized to achieve the maximization of loss in the domain classifier and minimization of loss in the label predictor. When the domain label is computed and it then starts to propagate backwards, the gradient reversal layer flips the sign of the gradient. As the model is trained with MNIST data with labels, the input MNIST data goes all the way through the label predictor and obtain a class loss. It will also go through the domain classifier and obtain a domain loss. Then it will backpropagate from both. When a data from MNIST-M is input where no labels exist, it would only go through the domain classifier where the convolutional part of the model is confused such that it cannot create features that would allow the domain classifier to work while it still can create features that would allow the label predictor to work. If the model is confused about what domain the data comes from but still able to classify the digits correctly, then we can assume that it does not use any domain related information of any sort. In the gradient reversal layer, the reversal is done by changing a parameter value which is changed from 0 to 1. But doing this directly could cause issues in the accuracy. In the beginning, convolutional features are not extracting anything as they do not have sufficient knowledge to know what to extract. Such convolution features, which can be called “bad convolutional features”, should not be used as the basis for predictions. Therefore, we have to get to a point where our convolutional features are good enough before we start to confuse them. The idea is therefore to ignore the domain loss in the beginning as it is going to be

noisy and we do not want to backpropagate domain confusion into the CNN layers in the beginning. To achieve this, we propose a function to gradually increase the Lambda parameter value from 0 to 1 during the course of training.Good performance results are obtained through the conducted experiments and the results are visualized.

II. LITERATURE REVIEW

This section analyses a series of related concepts and prior work related to the research.

A. Transfer Learning

Deep architectures use a series of layers to convert inputs into the necessary outputs. While the last layers carry out the task-specific inference, the initial layers improve the raw data's feature representations[9]. Throughout their progress of development,with increasing availability of a wealth of data, deep learning models have become more complicated and as a result, transfer learning[5], which examines "knowledge transfer" across machine learning models, has become increasingly popular in recent years.Deep models may be trained on big data to create robust feature representations, which can then be adjusted for the task at hand. With the use of transfer learning, big parameter models can be trained on enormous, generic datasets and subsequently fine-tuned for a particular purpose. Strong feature representations and a noticeably quicker convergence time are made possible via transfer learning. Based on the real world experiments[10], the estimated training time for the GPT-3 with 175 billion parameters utilizing 1024 A100 GPUs is just over a month. However, fine-tuning these models could be done with limited resource and within hours. Therefore, it is important to highlight the scales and proportions of time and resource that could be saved by utilizing existing already trained models to new tasks via Transfer Learning. Transfer Learning is not just utilized for transferring of tasks. Techniques for transfer learning can be used to transfer knowledge from one domain to another. Customizing a segmentation model for a new location or environment can be stated as an example. Even when the purpose is the same, the data or domain may be different. As models usually struggle to generalize to new domains, it is beneficial to transfer current information and employ adaptation procedures.

B. Semantic Segmentation

Semantic segmentation is a computer vision task that involves assigning a semantic label to every pixel in an image, thereby dividing the image into meaningful regions. This method is frequently employed in numerous fields, including autonomous vehicle technology, image analysis in medicine, and video monitoring. The objective of semantic segmentation is to categorize every pixel in an image into one of many pre-defined classes which can be done using Convolutional neural networks (CNNs), a type of deep learning technique that has been proven to be very successful for image processing tasks. The basic process of semantic segmentation is as follows:

- Input Image: The input image is fed into the deep learning model, which processes it through a series of convolutional layers to extract meaningful features.
- Encoding: The extracted features are then encoded into a compact representation using techniques such as pooling or down-sampling.
- Decoding: The encoded features are then decoded using techniques such as up-sampling or transposed convolution to reconstruct the original image.
- Classification: Finally, each pixel in the reconstructed image is classified into one of several pre-defined classes based on the features extracted in the earlier stages.

Applications of Semantic Segmentation includes Autonomous Driving, Medical Image Analysis, Video Surveillance, Object Detection and Tracking, Robotics, Augmented Reality and Environment Monitoring. In Autonomous driving, Semantic Segmentation can be used to identify and classify objects such as vehicles, people, roadway blocks, traffic signs, barriers etc. to enable safe driving of the vehicle for passengers as well as pedestrians [11]. Semantic Segmentation is used in medical image analysis to separately identify different tissues and organs in medical images for diagnosing

various diseases [12]. In Video Surveillance, Semantic Segmentation is used for real-time object detection and tracking which is crucial for security and monitoring applications [13]. In Augmented Reality, Semantic Segmentation is used to detect and track objects in real-time video streams accurately. Augmented Reality applications overlay virtual objects in to the real world on a live video stream [14]. Using Semantic Segmentation, the real-world scene is segmented into meaningful regions and the objects of interest are identified. This segmentation information is then used to track and overlay virtual objects on top of real-world objects in the video stream in an accurate as well as realistic manner [15]. One of the key challenges in Augmented Reality is occlusion, where virtual objects are partially or completely blocked by real-world objects. Using semantic segmentation, the occluding objects can be identified. Then the background can be modified as necessary for a realistic experience by hiding or modifying the occluding objects. Additionally, semantic segmentation is further employed to enhance user experience in Augmented Reality by enabling more interactive and realistic virtual objects[16]. Multiple functions and equations are utilized to achieve semantic segmentation.

1) Cross-entropy Loss Function:*Cross-entropy loss function is commonly used in semantic segmentation to measure the difference between the predicted class probabilities and the ground truth class labels for each pixel in the image.In this equation, L is the cross-entropy loss, y_{ij} is the ground-truth label for pixel(i,j) and p_{ij} is the predicted probability of the pixel that belongs to the same class as y_{ij}[17].*

$$L = -\sum_{i}\sum_{j} y_{ij} \log p_{ij} \quad (1)$$

2) Intersection over Union (IoU):*IoU is commonly used metric to evaluate the performance of a semantic segmentation model.In this equation, TP is the number of true positive pixels, FP is the number of false positive pixels and FN is the number of false negative pixels [18].*

$$IoU = TP / (TP + FP + FN) \quad (2)$$

3) Dice Coefficient: *Dice Coefficient is a similarity metric. It measures the agreement between the predicted segmentation and the ground truth.In this equation, TP is the number of true positive pixels, FP is the number of false positive pixels and FN is the number of false negative pixels [19].*

$$D = 2 * TP/(2 * TP + FP + FN) \quad (3)$$

C. Unsupervised Domain Adaptation

Unsupervised Domain Adaptation (UDA) aims to improve the performance of a model on a target domain by leveraging knowledge from a related but different source domain, without the need for labelled data in the target domain. UDA is particularly useful in scenarios where labeled data in the target domain is scarce or expensive to obtain[20]. UDA in general approaches DA by learning a domain-invariant feature representation that can capture the shared information between the source and target domains, while minimizing the differences between the two domains. This is achieved by using an UDA loss, such as the Maximum Mean Discrepancy (MMD) or the Domain Adversarial Neural Network (DANN) loss[8], which encourages the feature distributions of the source and target domains to be similar. Then, the learned domain-invariant features are utilized to train a classifier on the labeled source domain, which can be applied to the target domain with improved performance.

For UDA, a number of techniques have been developed, including MMD-based techniques, adversarial-based techniques, and reconstruction-based techniques. Deep Adaptation Networks (DAN)[21]. and Joint Adaptation Networks (JAN), two MMD-based techniques, employ MMD to reduce the domain disparity between the source and destination domains. With adversarial-based approaches like DANN and Adversarial Discriminative Domain Adaptation (ADDA), the feature extractor seeks to reduce the domain classification loss while the domain classifier attempts to discriminate between the source and target domains. Reconstruction-based techniques can produce realistic samples in the target domain by learning a mapping between the source and target domains.Examples of these techniques

include Pixel-Level Adaptation Networks (PLAN) and Image-to-Image Translation with Conditional Adversarial Networks (CycleGAN)[22]–[24]. Following are some of the commonly used functions in UDA techniques.

1) Maximum Mean Discrepancy (MMD): *MMD is a distance metric that measures the difference between the distributions of the source and target domain data in the feature space. The aim here is to minimize the distance between the two domains. Here,* P_S *and* P_T *are the source and target domain distributions,* x_i^S *and* x_j^T *are samples from the source and target domains.* ϕ *is a feature mapping function, and* n_S *and* n_T *respectively are the number of samples in the source and target domains.*

$$D_{mmd}(P_S, P_T) = \| \frac{1}{n_S} \sum_{i=1}^{n_S} \phi(x_i^S) - \frac{1}{n_T} \sum_{j=1}^{n_T} \phi(x_j^T) \|^2 \quad (4)$$

2) Domain Adversarial Neural Networks (DANN): *MMD is DANNinvolves adding a domain classifier to a neural network model and training it to simultaneously minimize the task loss and maximize the domain confusion. The domain confusion can be measured using the adversarial lossformula (5). In formula (5),* F *and* G *respectively are the feature extractor and task predictor and* x^S *and* x^T *respectively are the source and target domain samples.Domain classifier is denoted as* D*. The aim here is to minimize the task loss and maximize the domain confusion simultaneously, which is formulated as in formula (6).*

$$L_{adv} = -log(D(G(F(x^S)))) - log(1 - D(G(F(x^T)))) \quad (5)$$

$$L_{total} = L_{task} + \lambda * L_{adv} \quad (6)$$

3) Conditional Domain Adversarial Neural Networks (CDAN): *CDAN is based on DANNbut takes into account the conditional information in the target domain data. It involves adding a conditional domain classifier to the model and training it to minimize the task loss and maximize the conditional domain confusion, which can be measured using the adversarial loss formula (7) [25]. Here,* y_i^S *and* y_j^T *are the class labels of the source and target domain samples. The rest of the variables are the same as in the DANN equation. The aim here is to minimize the task loss and maximize the conditional domain confusion simultaneously, which can be formulated as in formula (8). Here,* λ *is a hyperparameter that controls the trade-off between the task loss and the conditional domain confusion.*

$$L_{adv} = -\sum_i \log(D(G(F(x_i^S), y_i^S))) - \sum_j \log(1 - D(G(F(x_j^T), y_j^T))) \quad (7)$$

$$L_{total} = L_{task} + \lambda * L_{adv} \quad (8)$$

D. Task Specific Boundaries

A key aspect of DA is identifying and defining the boundaries between source and target domains. These boundaries are called Task-specific boundaries. The specific aspects of the problem or task at hand that separate the source domain from the target domain are considered Task-specific boundaries. To guarantee that the model is properly adapted to the target domain, the adaptation process is guided by task-specific boundaries. Task-specific limits might be specified in terms of the input space, the output space, or both[26].

Task-specific boundaries in the input space describe characteristics or input variables that are pertinent to the task in the target domain but not in the source domain. For instance, if the source domain is pictures of cats and dogs and the target domain is pictures of wild animals, task-specific boundaries in the input space can include characteristics relating to the presence of trees, rocks, or other natural aspects in the image. Task-specific boundaries in the output space describe the labels or outputs that are relevant to the task in the target domain but not in the source domain. For instance, if the source domain is a dataset containing English texts and the target domain is a dataset containing Spanish texts, task-specific boundaries in the output space can include differences in grammar or syntax that have an effect on the labelling parts of speech or named entities.Task-specific boundaries can further be defined in terms of the loss function used to train the model. As Different tasks may require different loss functions, and adapting a model from one

domain to another may require modifying the loss function to account for differences in the target domain[27].

E. Feature Based Unsupervised Domain Adaptation

There are many different approaches for feature-based UDA methods. Projection based methods is one of them where an embedded space is learned. In this embedded space, the difference between feature distributions of source and target domains is minimized. This results in the learning of a common feature space [28], [29]. In the process of learning this common feature space, pivot selection plays a vital role. Furthermore, it should be noted that various forms of feature augmentations have been utilized in DA [30], [31]. Feature sparseness is a problem that exists in text classification. Feature expansion is a solution that is proposed to overcome this problem [32]. This solution predicts the mission features and expands the instances. The problem of feature sparseness is encountered in UDA when there is a small intersection of feature spaces in the source and target domain data distributions. Utilization of Diagnosing Encoders is another approach. These encoders do not rely on manual pivot selection [29], [33].

Pivot Selection has different strategies such as frequency of a pivot in a domain, Mutual Information (MI) or Pointwise Mutual Information (PMI) between a pivot and a domain label. Pivots can be defined as features with similar behaviour for discriminative learning in source and target domains [28]. In the work of [28], features that frequently appear in both domains (source and target) are selected as pivots. This pivot selection strategy does not require labelled data and has good performance for sequence labelling tasks. The work [28] has further shown that for discriminative classification tasks such as sentiment classification, MI is better than utilizing frequency. In the MI strategy, it is expected that the features discriminating the sentiment in the source domain will also discriminate the sentiment in the target domain. To that end, reviews with positive and negative sentiment labels are generated using MI between a feature and source domain. Pivotal features are those that have a high MI with either positively or negatively labelled reviews. This approach however required labelled data in the source domain for selecting pivots. A different strategy proposes an alternative definition for pivots where the features that are common to both source and target domain are considered as domain-independent features [34]. Other features are considered as domain-specific features. Domain-independent features are selected as pivots. In the proposed solution, MI between a feature and unlabelled training instances in each domain is used as a pivot selection strategy. It is assumed that features with low MI between source and target domain are likely domain-independent features. In UDA settings, the amount of unlabelled data in the source domains are usually significantly larger than that of labelled data. Therefore, it is possible to make better estimations of MI using unlabelled data. But it is not possible to solely utilize unlabelled data to select pivots that discriminate the classes related to the target prediction task. Another literature [35] proposes PMI [36] as a pivot selection strategy. It is a well-established measure for accurate work association. It has been utilized in multiple NLP tasks such as collocation detection [37], word similarity measurement [38] and relational similarity measurement [39]. A variant of PMI called Positive Pointwise Mutual Information (PPMI)[40]replaces all negative PMI values with zero and is proposed as another pivot selection strategy for UDA [41].

When adapting from one domain to another, the two domains are represented in domain-specific feature spaces, which is a fundamental challenge [3]. For learning transferable features between domains, learning a projection from the two domains to a domain-invariant space is the best option. During DA, it is important to reduce the mismatch of features between the source domain and the target domain. To achieve that, a technique called structural correspondence learning (SCL) is used to automatically determine the correspondence between a source domain and a target domain [28]. This approach was first presented for POS tagging and then expanded to include sentiment analysis [29]. To obtain the outcome, SCL first chooses k features using a particular selection technique. The

association between pivot features and non-pivot features is then modelled using *k* binary predictors that have been trained. The weight matrix is next subjected to Singular Value Decomposition (SVD) in order to learn a lower-dimensional projection for the pivot predictors. Finally, the labelled data represented as the concatenation of the original features and the predicted pivot features. This labelled data is used to train a logistic regression classifier.

An approach called Spectral Feature Alignment (SFA) has been developed for the classification of sentiment across domains[34]. In this approach, firstly, all features are separated into the domain-specific and domain-independent groups. These two groups are mutually exclusive. To obtain a bipartite graph between domain-specific and domain-independent features, SFA first counts how many times each feature occurs in both domains in the same document. When projecting domain-specific features, SFA modifies spectral clustering to produce a lower dimensional representation using top eigenvectors. Using labelled data from the source domain by the original features and the projected domain-specific features, a logistic regression model is learned in the last stage, which is similar to SCL.

A multi-source method for cross-domain sentiment classification is Sentiment Domain Adaptation Method from Multiple Sources (SDAMS) [42]. It is consisted of two components called a sentiment graph and a domain similarity metric. The sentiment polarity relationships between word pairs are used to extract the sentiment graph from unlabeled data. The word-pairs of domains in the sentiment graph serve as the foundation for the domain similarity metric. The sentiment graph of each domain is then used to learn the projection.

When it comes to feature expansion methods, a variety of solutions are available. Frequent Term Sets (FTS) [32]approaches feature extraction by computing the feature co-occurrence followed by selecting the expansion candidates under a pre-defined frequency threshold. The frequently co-occurring features are then used to expand the features in the original feature vectors. An improvement to this method is proposed with the introduction of support and confidence to the co-occurrence relationship [43] where the support and confidence are introduced at the phase when the frequent term sets are created for expansion. Sentiment Classification Thesaurus (SST) automatically creates a thesaurus to group various features that express the same sentiments to achieve cross-domain sentimentclassification [35].In the beginning, a set of features that co-occur with each feature and a set of sentiment features by the source labelled instances where the feature appears combine to form a feature vector for each feature. In order to generate a thesaurus, SST then ranks the traits according to how closely they are connected to one another. The tagged data in the source instances where the feature occurs is also to construct sentiment features for the thesaurus. The top *k* related features from the thesaurus generated in the preceding phase are then introduced, expanding the instance vector of the document. Finally, extended document vectors are used to learn a classifier.

There are various approaches utilizing DL for feature-based UDA. Stacked Denoising Autoencoders (SDA) could be considered as one of the earliest methods [29]. Autoencoder based methods are trained to minimize the loss between the original inputs and their reconstructions. In order to identify an invariant feature space on the source and target domain data, SDA first employs stacked denoising autoencoders. The in-variant characteristics that the encoder extracted and the original features from the source domain are then used to train a Support Vector Machine (SVM) classifier.SDA with Domain Supervision (SDA-DS) and SDA with Sentiment Supervision (SDA-SS) are two SDA variants suggested by latter research works [44]. SDA-DS adds a layer to SDA to its the input reconstruction process to predict the distribution of the domain whereas SDA-SS adds a layer to incorporate sentiment labels. Another extension to SDA is Marginalized Denoising Autoencoders (mSDA) [45]. This approach allows to compute the reconstruction mapping in close-form without corrupting a single instance [46]. The training is conducted layer by layer across the entire

dataset. During this training, each layer attempts to reconstruct the output of the previous layer. The inputs and hidden layers are then concatenated and used to train a SVM classifier. mSDA reduces the computational cost relatively to SDA while addressing its lack of scalability in the high-dimensional features. This is achieved by only constructing the features that are domain-independent [47]. Another neural approach is World Distribution Prediction (WDP) which is aimed at predicting the domain-to-domain distribution of word changes [41]. To achieve this, WDP begins by selecting pivots and using SVD to create two latent feature spaces. These feature spaces are created separately for the source domain and the target domain. Then it utilizes Partial Least Square Regression (PLSR) [48] to learn a mapping from source latent feature space to target latent feature space. Another approach that work similar to PLSR is Feature Embeddings for Domain Adaptation (FEMA) [49]. Unlike PLSR where pivots are used, FEMA obtains low-dimensional embeddings by utilizing a natural language model. An approach that combines multiple autoencoder based approaches [29], [45], [50]with SCL is Natural Structural Correspondence Learning (AE-SCL)[51]. This method maintains a structure that is similar to an autoencoder. However, it does not reconstruct all the input like many other autoencoder based methods. A prediction function from the non-pivot features to the pivot features is learned. The pivot features can then be rebuilt using the learned representation.Another neural approach is Adversarial Memory Network (AMN) [52]. This method has drawn inspiration from DANN [8]. Similar to DANN, it has two classifiers, each for predicting labels and domains. It is further consisted of two memory networks with shared parameters [53]. The two memory networks are utilized to extract pivots with two attributes. In this approach, pivots are captured automatically using attention and does not employ manual pivot selection like other methods. It further provides a direct visualization of the selected pivots.

F. Instance Based Unsupervised Domain Adaptation

Instance based UDA solutions are popular in computer vision and pattern recognition fields. Instance based methods aim at selecting the most suitable training instances in order to reduce the differences between the source domain and the target domain [54]. This is different from projection-based methods where the source domain and target domains are transferred to a shared subspace. Instance reweighing methods does not drop other training instances. Rather, they assign a new weight to each training instance. This is done to approximate the distribution of the target domain [55], [56]. There are other types of instance-based UDA solutions that are based on self-labelling. Labelled training data are hard to retrieve in classification tasks. When it comes to UDA, the availability of labelled source data is severely limited while a large amount of unlabelled data exists in both source and target domains. Self-labelling addresses this problem by utilizing the unlabelled data available [57].

When it comes to selecting and reweighing instances in instance-based UDA, an approach that is used to solve the problem of covariate shift is weighting or selecting instances based on their importance to the target domain[2]. These methods use the training instances of the source domain to learn a model on the target domain. A weight estimation technique called Maximum Mean Discrepancy (MMD)uses the difference between the means of mapped representations of data distributions in a reproducing kernel Hilbert space (RKHS)[58]. This is done to assess whether the distributions are from the same or separate domains. However, MMD does not generate the samples.It solves an expensive semi-definite program(SDP) to learn a latent space. A different approach that does not use parameters is Kernel Mean Matching (KMM) [56]. It aims to close up the means of training and test features in RKHS by reweighting the training instances. This instance weighting method tries to solve DA by weighting the loss of the source instances. Transfer Component Analysis (TCA) is another method that uses MMD to minimize dissimilarities across source and target domains in order to introduce a shared latent subspace [59]. The subspace preserves the

properties of the original data. TCA has further been applied to multi-source DA [60]. It should however be noted that in TCA, all the source domains are equally weighted in the joint matrix.For the purpose of identifying training instances that are the most useful for DA in the context of cross-domain sentiment classification, Positive and Unlabelled Learning (PU Learning) method is proposed [54].In this approach, a binary in-target-domain selector is constructed and in-target-domain probabilities are added to source instances. Two models are proposed in this approach. One model selects the instances with higher in-target-domain probabilities and the other model reweights the instances with in-target-domain probabilities. The two models train a Bayesian classifier weighted with instances. When selecting data, many instance selection methods have utilized similarity measures [61], [62]. Bayesian optimization is employed to weight training instances from several source domains using a set of similarity and diversity indicators where existing similarity and diversity measures are used to define a data selection model [63]. Using a small collection of validation data, Bayesian optimization is used to learn the weights in order to optimize the objective function for the specific task. The work of this research shows that diversity measures are of significant importance in DA even though it does not outperform state-of-the-art methods. It further provides cross-model, cross-domain, and cross-task outcomes for the learned measures.Another approach presents an instance selection method aimed at machine translation [64]. This method utilizes the sum of cross-entropy differences over sentences of the corpus in order to rank source-domain instances with respect to target-domain instances.

In self-labelling, a model is trained on the labelled instances. The trained model is then utilized to assign pseudo labels to unlabelled instances [65]. In UDA, a large amount of unlabeled data is available. By utilizing self labelling methods, these unlabeled data can be given pseudo labels. Then the data with pseudo labels can be used to create the feature space of training data. When deploying a deep learning model that has been trained in a source domain to unlabelled target domains, self-training-based UDA has demonstrated tremendous promise for addressing the issue of domain shift. Self-training has been adopted to many different cross-domain tasks from NLP to image classification [66]–[68]. Another approach is Co-training [69] where the problem of using a large unlabelled sample to boost the performance of an algorithm with finite amount of labelled data is addressed. This method assumes the existence of multiple views of the feature space. Accordingly, in the most basic scenario, two views can be available for the feature space. Here, the labelled instances of the source domain are leveraged to learn a separate classifier. During this process, features from only a particular view are involved per classifier. In the most basic scenario with two views, there would be two classifiers. Then, pseudo labels of the unlabelled instances of the target domain are predicted using the learned dual classifiers. A label is assigned to an unlabelled instance of the target domain when both classifiers agree on that label. Co-training approach has been applied in UDA in scenarios where the source and target domain feature spaces has multiple views [70]. The complementarity of the data gathered by the various feature spaces will determine the effectiveness of co-training. To that end, the importance of constructing feature spaces carefully and properly should be highlighted. Another self-labelling approach that utilizes three classifiers is Tri-Training [71]. The three classifiers are trained with subsets of instances. These instances are sampled from the available labelled instances. Co-training has the requirement for the construction of sufficient features spaces and availability of redundant views [70]. This requirement is relaxed in Tri-training [71]. In the Tri-training approach, only two classifiers out of the three have to agree upon a label for it to be assigned to a particular unlabelled instance. Based on Tri-training, an improved version of the method has been proposed where in order to assign a label to an unlabelled instance, not only two classifiers have to agree on the particular label, but the third classifier has to disagree [72]. This has led to the reduction of the number of additional instances and the

diversification of the sample process. Tri-training is further improved by adding three neural networks [73]. Two of the three neural networks are used to give pseudo labels for the unlabelled data in the target domain. The third neural network uses those pseudo labels to learn a discriminator. Multi-task Tri-training (MT-Tri) [74]is an advanced improved approach based on Tri-training and Bi-LTSM methods. This method reduces the time and space complexity compared to the traditional approaches.

When it comes to neural approaches of instance-based UDA, Deep Domain Confusion Network (DDCN) is a DA method that utilizes CNN and is based on discrepancy [75]. In order to minimize distribution distance (computed using MMD) between the source domain and the target domain this method attempts to learn a representation. The loss function of the method utilizes the learnt representation for classification tasks. The method further focuses on adjusting the confusion between the source domain and the target domain by introducing a hyperparameter for regularization. Then, it adds the MMD loss on top of the layer to standard AlexNet architecture [76]. Modified AlexNet architecture has 8 layers and is used in other methods such as Deep Adaptation Network (DAN) [21]. In DAN, out of the 8 layers in its modified AlexNet architecture, the first 3 layers are utilized for learning general transferable features from the source domain. The middle 2 layers are utilized for learning features that are specific to the domains. The last 3 layers are utilized for learning features that are invariant to the domains. In the last layer, the discrepancy loss takes place. The RKHS distance between the mean embeddings of two distributions is calculated using a multiple kernel variation of MMD (MK-MMD)[21]. In order to perform feature learning, DA, and classifier learning concurrently in the model architecture by utilizing backpropagation for UDA, a method called Domain Adversarial Neural Networks (DANN) has been proposed with the aim of learning features that are both discriminative as well as domain-invariant [8]. DANN method is consisted of three separate parameters. The method seeks to maximize the loss in the label-predictor and to minimize the loss in the domain classifier. The label predictor essentially tries to predict the class labels while the domain classifier tries to identify the domain from which the data is coming from. By generalizing DANN, Multiple Source DA with Adversarial Learning (MDAN) has been proposed [77]. It aims to learn features that are independent of the domain but still relevant to the target task. MDAN is consisted of two versions called hard and smooth. In the hard version, gradient reversal backpropagates the source domain that achieves the minimum domain classification error. In the smooth version, all classification errors are adaptively combined and backpropagated. Adversarial Discriminative DA (ADDA) [24]is another method that is based on GAN [78]. In ADDA, labels of the data in the source domain are utilized to learn a discriminator. Then, domain adversarial loss is utilized by an encoder to map the target data to the same space. The ultimate goal here is to be able to utilize the source domain model on the target domain directly. To that end, the method attempts to minimize the distance of mapping distributions between the two domains. Another multiple-source DA method is Mixture of Experts (MoE)[79]. MoE utilizes a point-to-set metric to model the relations in the domains. The relations are modeled to an encoded training matrix for source domains. This is followed by the application of meta-training to update the model parameters by conducting a joint training over all the available domain-pairs [80].

III. METHOD

A. Problem

The problem is related to different domains but identical tasks. In this particular case, it is the identification of digits from non-labelled digit images in the MNIST-M dataset using a model trained on labelled MNIST dataset.

Fig. 1 MNIST and MNIST-M Datasets

In the UDA setting, we assume that there are no labelled data available for the target domain. The intuition is that we need a way to force our Convolutional Neural Network to learn features of the digit-outline-shapes only, ignoring the color distributions. The approach is based on the model proposed by [8] which utilizes the concept of Domain Confusion Loss.

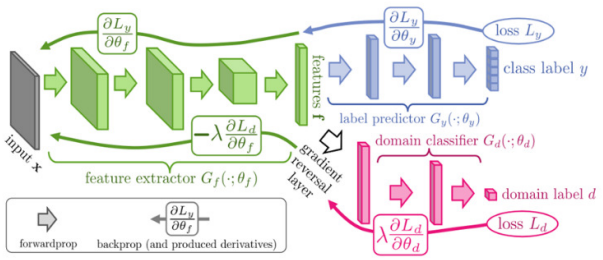


Fig. 2 DANN

The model is consisted of a deep feature extractor highlighted in the figure in green color and a deep label predictor highlighted in blue color. These two form a feed-forward architecture. A domain classifier as highlighted in red color is added to achieve UDA. A Gradient Reversal Layer (GRL) exists between the feature extractor and the domain classifier. During the backpropagation, the GRL multiplies the gradient by a negative constant.

The idea here is to use two different classification heads that are placed above the convolutional feature extractor. When a data X is input, which, in our case, an image of a digit from MNIST or MNIST-M datasets, it forward passes through the feature extractor where its convolutional features are extracted. These extracted features are composed in to a vector which goes through the subsequent fully connected layers. The domain classifier, which is the second classification head that has been added to the model, takes in the exact same features extracted by the feature extractor and passes them through a different set of fully connected layers attempting to predict the domain from the features. If we assume that MNIST as Domain 0 and MNIST-M as Domain 1, the domain classifier would try to predict if a given set of convolutional features belongs to either Domain 0 or Domain 1. There are losses associated with both the label predictor and domain classifier. In the case of the domain classifier, its loss is treated as confusion loss. The reasoning behind is that we want the domain classifier to be confused about domain of which the features are passing through the layers of the domain classifier. In order to achieve this, the loss in the domain classifier is maximized. At the same time, in order to improve the accuracy of prediction, the loss of the label predictor is minimized. This essentially leads to a point where the model is capable of correctly identifying the digit of a given input digit image without having an idea of which domain it comes from, meaning that the model is trained with a certain level of accuracy to identify data on the target domain.

B. Gradient Reversal Layer

A Gradient Reversal Layer (GRL) is utilized to achieve the maximization of loss in the domain classifier and minimization of loss in the label predictor. When the domain label is computed and it then starts to propagate backwards, the gradient reversal layer flips the sign of the gradient. As the model is trained with MNIST data with labels, the input MNIST data goes all the way through the label predictor and obtain a class loss. It will also go through the domain classifier and obtain a domain loss. Then it will backpropagate from both. When a data from MNIST-M is input where no labels exist, it would only go through the domain classifier where the convolutional part of the model is confused such that it cannot create features that would allow the domain classifier to work while it still can create features that would allow the label predictor to work. If the model is confused about what domain the data comes from but still able to classify the digits correctly, then we can assume that it does not use any domain related information of any sort.

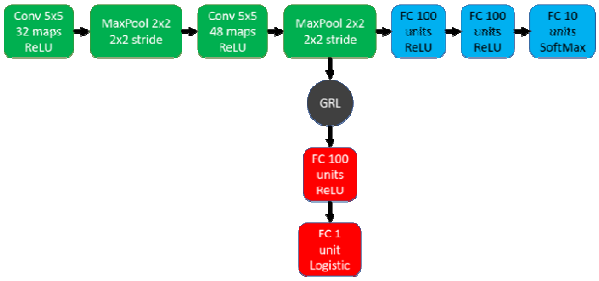


Fig. 3 Architecture

The model is consisted of three parts, as in fig. 3:

- A "deep" CNN for image feature extraction (2x Conv, ReLU, MaxPool) (not- that deep in this case)
- A digit-classification head (3x FC, ReLU)
- A domain classification head (2x FC, ReLU), with GRL

GRL is no-operation in forward pass, but applies a –λ factor to gradient in the backward pass to flip the sign of the gradient.

This MNIST architecture is inspired by the classical LeNet-5[81]. If we do something in the forward pass, such as, multiply it by a certain factor, it would also affect the backward pass. We need something that has no effect in the forward pass but has an effect in the backward pass. To achieve this, a function is created using PytorchAutograd functions. It is a way to define the operation and explicitly say that this operation works on tensors as well as what it needs to do in the forward and backward passes. The implemented function reverses the gradient in the backward pass whereas it does nothing in the forward pass but stores a parameter which is utilized to multiply the gradient. On the backwards pass, the function loads the previously stored data, takes the gradient of the output of the function and computes the gradient of the input. The constant is then applied for that and returned.

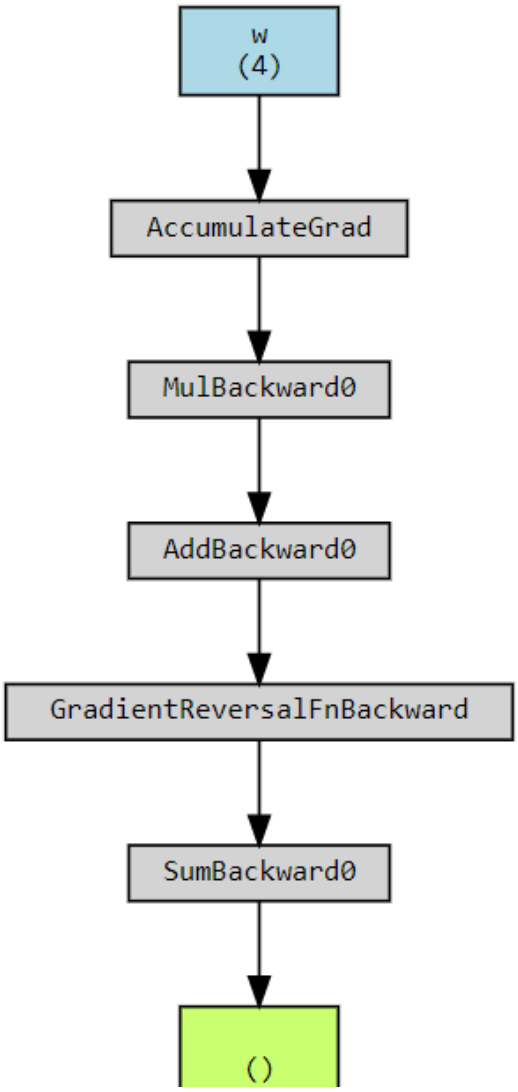


Fig. 4 Computational Graph Generated by TorchViz

C. Lambda Change

$$\lambda_P = \frac{2}{1 + \exp(-10.p)} - 1 \quad (9)$$

Here,$p \in [0,1]$ is the training process. Lambda parameter is changed in the forward function during training, where, it is increased overtime. The reasoning behind this is that in the beginning, the convolutional features are not extracting anything as they do not have sufficient knowledge to know what to extract. Such convolution features, which can be called “bad convolutional features”, cannot be used as the basis for predictions. Therefore, we

have to get to a point where our convolutional features are good enough before we start to confuse them. The idea is therefore to ignore the domain loss in the beginning as it is going to be noisy and we do not want to backpropagate domain confusion into the CNN layers in the beginning. Therefore, the lambda is gradually changed from 0 to 1 in the course of the training.

IV. EXPERIMENTS AND RESULTS

The experiments were conducted using Pytorch and Jupyter Notebook as the primary environment. Google Colab platform was utilized for computing power with Nvidia GPUs.

Initial Training was conducted for 30 epochs. Pytorch Ignite Wrapper was utilized for feature visualizations. Fig. 5 shows the initial feature visualization between the source (MNIST) and target (MNIST-M) domains before training.

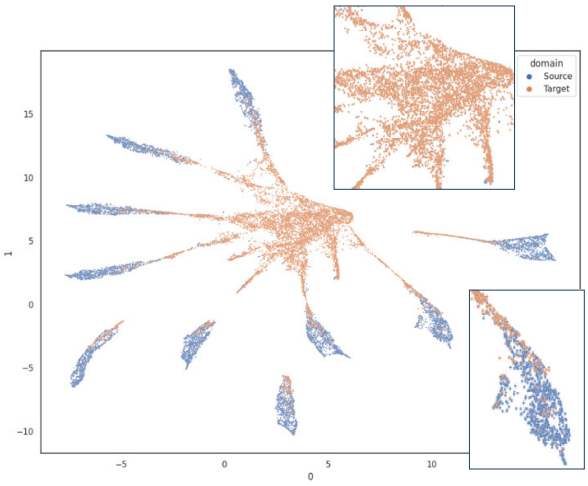


Fig. 5 Initial Feature Visualization

After training for 30 epochs, the feature visualization in fig. 6 was received.

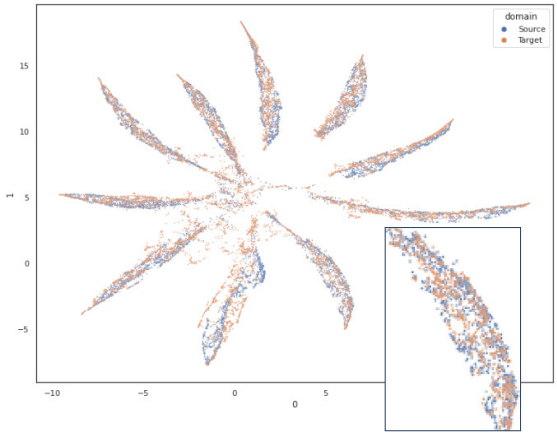


Fig. 6 Feature Visualization after 30 epochs

The best epoch was epoch 28 as depicted in fig. 7.

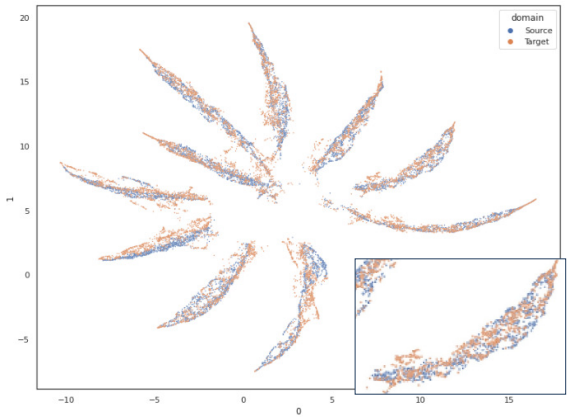
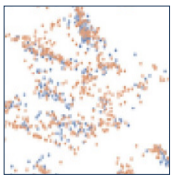
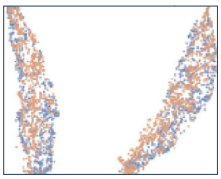


Fig. 7 Epoch 28 Feature Visualization

The accuracy of the trained model on the target domain recached 77% during the experimentation upon testing it on the target domain MNIST-M.

Further training was conducted up to 50 epochs and the feature visualization in fig. 8 was received. The accuracy on the target data reached 80.9%. The results of the experiment are compared with other methods in Table 1.





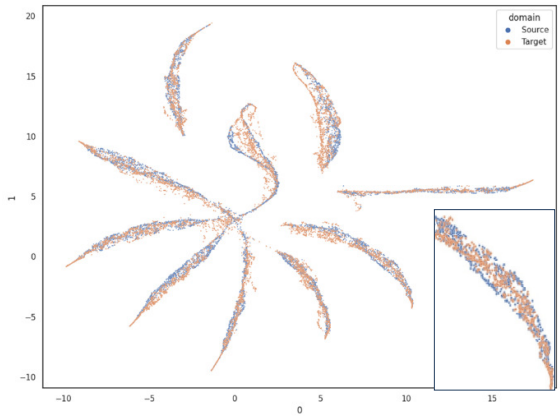


Fig. 8: Final Feature Visualization

TABLE 1
RESULT COMPARISON

	Method	Source → Target MNIST → MNIST-M
1	Source only	55.2
2	DANN [8]	76.7
3	DSN[82]	83.2
4	DeepJDOT[83]	92.4
5	DRANet[84]	98.7
6	Our Experimentation Results	80.9

V. DISCUSSION

The use of task-specific boundaries can have several advantages with regards to enhancing the transferability of neural models.

- Improved Generalization
- Better Feature Representation
- Better Model Architecture
- Improved Model Performance

By setting task-specific boundaries, the model can be better focused on the task at hand and the issue of overfitting to the training data can be avoided. This can result in improved generalization to unseen data, particularly when the target domain is different from the source domain. Task-specific boundaries can positively influence the architecture of the model. For instance, the number of classes or categories in a classification task can determine the number of output neurons and the length of the input sequence in a natural language processing task can determine the size of the recurrent neural network.The model can be optimized for the specific requirements of the task by taking task-specific boundaries into consideration. This can contribute to performance improvements over a general-purpose model.

With all these advantages, it is important to note the necessity to carefully balance the specificity of the task-specific boundaries with the need for the model to generalize to new data. The reasoning is that overly restrictive boundaries may limit the ability of the model to transfer to new tasks.

Another aspect to note is that the accuracy and transferability of neural models may be impacted by the disparity in complexity between the datasets from the source and destination domains. The varying degrees of complexity can be attributed to variations in the way features are distributed, the number of classes or categories present, the size of the dataset, and other aspects of the data itself. The model may struggle to adapt to the target domain and may not generalize well to the target data, for instance, if the source domain's feature distribution is simple and the target domain's feature distribution is more complex. Several measures, such as the KL divergence between the source and target domains, the Wasserstein distance, or other distance metrics, can be used to determine the complexity difference.The cost of accuracy that a neural model could suffer without having access to any unlabelled target domain data can be significant, especially if the source and target domains are significantly different. The domain shift refers to the difference between the distribution of features in the source domain and the target domain. This can result in poor performance on the target domain and a reduction in accuracy. Without access to labelled or unlabelled data in the target domain, UDA cannot be performed.In the absence of any unlabelled target domain data, the model would be limited to using only the information it has learned from the source domain. This can make it difficult for the model to generalize to the target domain and may result in overfitting or underfitting. The accuracy of the model on the target domain can typically be significantly increased by having access to unlabelled target domain data. This is so that the model can better generalize to the target

data by learning the features of the target domain. The model can learn a more accurate representation of the target domain and enhance its performance on the target data by using unsupervised learning techniques like clustering or generative models. Many performance metrics, including accuracy, F1-score, precision, recall, and others, can be used to compare the accuracy between having access to unlabelled target domain data and not having it. In cases where a limited amount of labelled data is available for the target domain where the rest of the available data are unlabelled, the accuracy of the model on the target domain can obtain significant improvements.

In the experiments conducted based on the model proposed in [8] where domain confusion is utilized to effectively achieve DA by confusing the model of its knowledge of the domain and reducing the loss in its label classifier, we obtained an accuracy of around 80.9% upon training it on the source dataset (MNIST) and applying on the target dataset (MNIST-M), an accuracy that improved proportionate to the number of epochs the training was performed. The maximization of loss in the domain classifier and simultaneous minimization of loss in the label predictor is achieved by the Gradient Reversal Layer component of the model that applies a -λ factor in the backward passes. With the introduction of gradually increasing its value from 0-1 during the course of training rather than setting it to 1 from the very beginning has allowed the model to not utilize initial bad convolutional features to make predictions which is an improvement over the conventional approach.

VI. CONCLUSION

The development of transferable deep neural networks has been a key focus of research in recent years, due to their potential to improve model generalization and reduce the need for labelled data. UDA has emerged as a promising approach to enhance the transferability of deep neural networks, enabling the models to generalize to new domains without the need for labelled data in the target domain. This thesis has explored various techniques and methodologies for UDA, including domain adversarial training, discrepancy-based methods, and self-supervised learning, and has shown their effectiveness in improving model performance on transfer tasks. Experiments have been conducted based on model proposed in [8] while adopting a new strategy in its Gradient Reversal Layer to gradually increase the lambda value from 0 to 1 throughout the course of training to ensure that the model maximizes the domain confusion loss and minimizes the class label loss without being affected by bad convolutional features. The testing conducted on the target dataset MNIST-M provided an accuracy of around 77% upon 30 epochs of training. It was subsequently improved to 80.9% with 50 epochs of training. It was observed that the accuracy is proportional to the number of training epochs. With higher number of training epochs, the model has the potential to reach a higher level of accuracy on the target data.

According to the experimental findings, UDA can significantly increase the ability of deep neural networks to transfer across domains, tasks, and modalities. By utilizing different domain-specific knowledge or prior information, the transfer learning performance can be further improved. UDA still faces a number of obstacles and restrictions, including the domain shift assumption, the choice of suitable domain discrepancy measures, and the difficulty of adjusting to drastic domain shifts. Thus, additional research is required to create unsupervised domain adaption methods that are more reliable and efficient.

Further improvements and future works include testing the model on different but related target datasets other than MNIST-M and comparing the accuracies with the other existing research works.

Additionally, the state-of-the-art UDA methods for improving the transferability of deep neural networks are reviewed and analysed in-depth in this paper, which can be helpful for both academics and industry professionals working on transfer learning and DA issue

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my supervisor, Prof. Qian Ting, for his unwavering support, guidance, and patience throughout the course of this research. His expertise and valuable

feedback have been instrumental in shaping this work, and I am truly fortunate to have had the opportunity to work with him.

REFERENCES

[1] M. Cordts *et al.*, 'The Cityscapes Dataset for Semantic Urban Scene Understanding'. arXiv, Apr. 07, 2016. doi: 10.48550/arXiv.1604.01685.

[2] W. M. Kouw and M. Loog, 'An introduction to domain adaptation and transfer learning', Dec. 2018. doi: 10.48550/arXiv.1812.11806.

[3] W. M. Kouw and M. Loog, 'A review of single-source unsupervised domain adaptation', *ArXiv*, Jan. 2019, Accessed: May 12, 2023. [Online]. Available: https://www.semanticscholar.org/paper/A-review-of-single-source-unsupervised-domain-Kouw-Loog/60680af7ed7eb3214e295863d106307a1d28bd30

[4] S. J. Pan and Q. Yang, 'A Survey on Transfer Learning', *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010, doi: 10.1109/TKDE.2009.191.

[5] S. J. Pan and Q. Yang, 'A Survey on Transfer Learning', *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010, doi: 10.1109/TKDE.2009.191.

[6] A. Argyriou, T. Evgeniou, and M. Pontil, 'Multi-Task Feature Learning', in *Advances in Neural Information Processing Systems*, MIT Press, 2006. Accessed: Mar. 26, 2023. [Online]. Available: https://proceedings.neurips.cc/paper/2006/hash/0afa92fc0f8a9cf051bf2961b06ac56b-Abstract.html

[7] S. Ruder, 'Neural transfer learning for natural language processing', Jun. 2019. Accessed: Feb. 21, 2023. [Online]. Available: https://www.semanticscholar.org/paper/Neural-transfer-learning-for-natural-language-Ruder/85e3010ff82c07961bc21f63b91a4981fa5123fe

[8] Y. Ganin *et al.*, 'Domain-Adversarial Training of Neural Networks'. arXiv, May 26, 2016. doi: 10.48550/arXiv.1505.07818.

[9] J. Liang, D. Hu, and J. Feng, 'Do We Really Need to Access the Source Data? Source Hypothesis Transfer for Unsupervised Domain Adaptation'. arXiv, Jun. 01, 2021. Accessed: Mar. 26, 2023. [Online]. Available: http://arxiv.org/abs/2002.08546

[10] 'Scaling Language Model Training to a Trillion Parameters Using Megatron | NVIDIA Technical Blog'. https://developer.nvidia.com/blog/scaling-language-model-training-to-a-trillion-parameters-using-megatron/ (accessed Mar. 26, 2023).

[11] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, 'Multimodal Unsupervised Image-to-Image Translation'. arXiv, Aug. 14, 2018. doi: 10.48550/arXiv.1804.04732.

[12] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, 'Perceptual Generative Adversarial Networks for Small Object Detection'. arXiv, Jun. 20, 2017. doi: 10.48550/arXiv.1706.05274.

[13] G. Litjens *et al.*, 'A survey on deep learning in medical image analysis', *Med. Image Anal.*, vol. 42, pp. 60–88, Dec. 2017, doi: 10.1016/j.media.2017.07.005.

[14] P. Hu *et al.*, 'Real-Time Semantic Segmentation With Fast Attention', *IEEE Robot. Autom. Lett.*, vol. 6, no. 1, pp. 263–270, Jan. 2021, doi: 10.1109/LRA.2020.3039744.

[15] Z. Huang *et al.*, 'CCNet: Criss-Cross Attention for Semantic Segmentation'. arXiv, Jul. 09, 2020. doi: 10.48550/arXiv.1811.11721.

[16] J. Dong, J. Guo, H. Yue, and H. Gao, 'EANET: Efficient Attention-Augmented Network for Real-Time Semantic Segmentation', in *2022 IEEE International Conference on Image Processing (ICIP)*, Oct. 2022, pp. 3968–3972. doi: 10.1109/ICIP46576.2022.9897589.

[17] J. Long, E. Shelhamer, and T. Darrell, 'Fully convolutional networks for semantic segmentation', in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 3431–3440. doi: 10.1109/CVPR.2015.7298965.

[18] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman,

'The PASCAL Visual Object Classes Challenge: A Retrospective', *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, Jan. 2015, doi: 10.1007/s11263-014-0733-5.

[19] 'Measures of the Amount of Ecologic Association Between Species on JSTOR'. https://www.jstor.org/stable/1932409 (accessed Mar. 26, 2023).

[20] Y. Ganin and V. Lempitsky, 'Unsupervised domain adaptation by backpropagation', in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, in ICML'15. Lille, France: JMLR.org, Jul. 2015, pp. 1180–1189.

[21] M. Long and J. Wang, 'Learning Transferable Features with Deep Adaptation Networks', Feb. 2015.

[22] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, 'Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks', in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 95–104. doi: 10.1109/CVPR.2017.18.

[23] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, 'StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation'. arXiv, Sep. 21, 2018. doi: 10.48550/arXiv.1711.09020.

[24] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, 'Adversarial Discriminative Domain Adaptation', in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 2962–2971. doi: 10.1109/CVPR.2017.316.

[25] M. Long, Z. Cao, J. Wang, and M. I. Jordan, 'Conditional Adversarial Domain Adaptation'. arXiv, Dec. 29, 2018. doi: 10.48550/arXiv.1705.10667.

[26] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, 'A theory of learning from different domains', *Mach. Learn.*, vol. 79, no. 1, pp. 151–175, May 2010, doi: 10.1007/s10994-009-5152-4.

[27] B. Sun, J. Feng, and K. Saenko, 'Return of Frustratingly Easy Domain Adaptation'. arXiv, Dec. 09, 2015. doi: 10.48550/arXiv.1511.05547.

[28] J. Blitzer, R. McDonald, and F. Pereira, 'Domain Adaptation with Structural Correspondence Learning', in *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia: Association for Computational Linguistics, Jul. 2006, pp. 120–128. Accessed: Mar. 17, 2023. [Online]. Available: https://aclanthology.org/W06-1615

[29] X. Glorot, A. Bordes, and Y. Bengio, 'Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach', presented at the International Conference on Machine Learning, Jun. 2011. Accessed: May 12, 2023. [Online]. Available: https://www.semanticscholar.org/paper/Domain-Adaptation-for-Large-Scale-Sentiment-A-Deep-Glorot-Bordes/6f4065f0cc99a0839b0248ffb4457e5f0277b30d

[30] H. Daumé III, 'Frustratingly Easy Domain Adaptation'. arXiv, Jul. 10, 2009. doi: 10.48550/arXiv.0907.1815.

[31] Daumé, H. Iii, A. Kumar, and A. Saha, 'Frustratingly Easy Semi-Supervised Domain Adaptation', Jul. 2010. Accessed: May 12, 2023. [Online]. Available: https://www.semanticscholar.org/paper/Frustratingly-Easy-Semi-Supervised-Domain-Daum%C3%A9-H./07e739d9d019c7c7974c264567b35ab7849e5cc4

[32] Y. Man, 'Feature Extension for Short Text Categorization Using Frequent Term Sets', in *Procedia Computer Science*, 2014, pp. 663–670. doi: 10.1016/j.procs.2014.05.314.

[33] M. Chen, K. Q. Weinberger, Z. Xu, and F. Sha, 'Marginalizing stacked linear denoising autoencoders', *J Mach Learn Res*, 2015, Accessed: May 12, 2023. [Online]. Available: https://www.semanticscholar.org/paper/Marginalizing-stacked-linear-denoising-autoencoders-Chen-Weinberger/c88bd42892bf3735bd482626b56f2b93ca7e00d8

[34] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen, 'Cross-domain sentiment classification via spectral feature alignment', in *Proceedings of the 19th international conference on World wide web*, in WWW '10. New York, NY, USA: Association for Computing Machinery, Apr. 2010, pp. 751–760. doi: 10.1145/1772690.1772767.

[35] D. Bollegala, T. Mu, and J. Y. Goulermas, 'Cross-Domain Sentiment Classification Using Sentiment Sensitive Embeddings', *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 2, pp. 398–410, Feb. 2016, doi: 10.1109/TKDE.2015.2475761.

[36] K. W. Church and P. Hanks, 'Word association norms, mutual information, and lexicography', *Comput. Linguist.*, vol. 16, no. 1, pp. 22–29, Mar. 1990.

[37] G. Weikum, 'Foundations of statistical natural language processing', in *ACM SIGMOD Record*, Sep. 2002, pp. 37–38. doi: 10.1145/601858.601867.

[38] P. D. Turney, 'Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL', in *Proceedings of the 12th European Conference on Machine Learning*, in EMCL '01. Berlin, Heidelberg: Springer-Verlag, Sep. 2001, pp. 491–502.

[39] P. D. Turney, 'Similarity of Semantic Relations', *Comput. Linguist.*, vol. 32, no. 3, pp. 379–416, Sep. 2006, doi: 10.1162/coli.2006.32.3.379.

[40] D. Lin, 'Automatic retrieval and clustering of similar words', in *Proceedings of the 36th annual meeting on Association for Computational Linguistics* -, Montreal, Quebec, Canada: Association for Computational Linguistics, 1998, pp. 768–774. doi: 10.3115/980691.980696.

[41] D. Bollegala, D. Weir, and J. Carroll, 'Learning to predict distributions of words across domains', *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Jun. 11, 2014. http://dx.doi.org/10.3115/v1/P14-1058 (accessed May 12, 2023).

[42] F. Wu and Y. Huang, 'Sentiment Domain Adaptation with Multiple Sources', in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 301–310. doi: 10.18653/v1/P16-1029.

[43] H. Ma, L. Di, X. Zeng, L. Yan, and Y. Ma, 'Short Text Feature Extension Based on Improved Frequent Term Sets', in *Intelligent Information Processing VIII*, Z. Shi, S. Vadera, and G. Li, Eds., in IFIP Advances in Information and Communication Technology. Cham: Springer International Publishing, 2016, pp. 169–178. doi: 10.1007/978-3-319-48390-0_18.

[44] B. Liu, M. Huang, J. Sun, and X. Zhu, 'Incorporating Domain and Sentiment Supervision in Representation Learning for Domain Adaptation', presented at the International Joint Conference on Artificial Intelligence, Jul. 2015. Accessed: May 12, 2023. [Online]. Available: https://www.semanticscholar.org/paper/Incorporating-Domain-and-Sentiment-Supervision-in-Liu-Huang/aab7c5c61c8df2417a2dbd231f82f87442be98b6

[45] M. Chen, Z. Xu, K. Weinberger, and F. Sha, 'Marginalized Denoising Autoencoders for Domain Adaptation'. arXiv, Jun. 18, 2012. doi: 10.48550/arXiv.1206.4683.

[46] U. Kamath, J. Liu, J. Whitaker, U. Kamath, J. Liu, and J. Whitaker, 'Transfer Learning: Domain Adaptation', pp. 495–535, 2019, doi: 10.1007/978-3-030-14596-5_11.

[47] R. Liu, Y. Shi, C. Ji, and M. Jia, 'A Survey of Sentiment Analysis Based on Transfer Learning', *IEEE Access*, vol. 7, pp. 85401–85412, 2019, doi: 10.1109/ACCESS.2019.2925059.

[48] D. V. Guebel and N. V. Torres, 'Partial Least-Squares Regression (PLSR)', in *Encyclopedia of Systems Biology*, W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, Eds.,

New York, NY: Springer, 2013, pp. 1646–1648. doi: 10.1007/978-1-4419-9863-7_1274.

[49] Y. Yang and J. Eisenstein, 'Unsupervised Multi-Domain Adaptation with Feature Embeddings', in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Denver, Colorado: Association for Computational Linguistics, 2015, pp. 672–682. doi: 10.3115/v1/N15-1069.

[50] Y. Yang and J. Eisenstein, 'Fast Easy Unsupervised Domain Adaptation with Marginalized Structured Dropout', in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Baltimore, Maryland: Association for Computational Linguistics, 2014, pp. 538–544. doi: 10.3115/v1/P14-2088.

[51] Y. Ziser and R. Reichart, 'Neural Structural Correspondence Learning for Domain Adaptation', *Proc. 21st Conf. Comput. Nat. Lang. Learn. CoNLL 2017*, pp. 400–410, 2017, doi: 10.18653/v1/K17-1040.

[52] Z. Li, Y. Zhang, Y. Wei, Y. Wu, and Q. Yang, 'End-to-end adversarial memory network for cross-domain sentiment classification', in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, in IJCAI'17. Melbourne, Australia: AAAI Press, Aug. 2017, pp. 2237–2243.

[53] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, 'End-To-End Memory Networks'. arXiv, Nov. 24, 2015. doi: 10.48550/arXiv.1503.08895.

[54] R. Xia, X. Hu, J. Lu, J. Yang, and C. Zong, 'Instance selection and instance weighting for cross-domain sentiment classification via PU learning', in *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, in IJCAI '13. Beijing, China: AAAI Press, Aug. 2013, pp. 2176–2182.

[55] H. Shimodaira, 'Improving predictive inference under covariate shift by weighting the log-likelihood function', *J. Stat. Plan. Inference*, vol. 90, no. 2, pp. 227–244, Oct. 2000, doi: 10.1016/S0378-3758(00)00115-4.

[56] B. Schölkopf, J. Platt, and T. Hofmann, Eds., 'Correcting Sample Selection Bias by Unlabeled Data', The MIT Press, 2007. doi: 10.7551/mitpress/7503.003.0080.

[57] X. (Jerry) Zhu, 'Semi-Supervised Learning Literature Survey', University of Wisconsin-Madison Department of Computer Sciences, Technical Report, 2005. Accessed: May 13, 2023. [Online]. Available: https://minds.wisconsin.edu/handle/1793/60444

[58] B. Schölkopf, J. Platt, and T. Hofmann, Eds., 'A Kernel Method for the Two-Sample-Problem', The MIT Press, 2007. doi: 10.7551/mitpress/7503.003.0069.

[59] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, 'Domain Adaptation via Transfer Component Analysis', *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011, doi: 10.1109/TNN.2010.2091281.

[60] T. Grubinger, A. Birlutiu, H. Schöner, T. Natschläger, and T. Heskes, 'Domain Generalization Based on Transfer Component Analysis', I. Rojas, G. Joya, and A. Catala, Eds., in Lecture Notes in Computer Science, vol. 9094. Cham: Springer International Publishing, 2015, pp. 325–334. doi: 10.1007/978-3-319-19258-1_28.

[61] B. Plank and G. van Noord, 'Effective measures of domain similarity for parsing', in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, in HLT '11. USA: Association for Computational Linguistics, Jun. 2011, pp. 1566–1576.

[62] V. Van Asch and W. Daelemans, 'Using domain similarity for performance estimation', in *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, in DANLP 2010. USA: Association for Computational Linguistics, Jul. 2010, pp. 31–36.

[63] E. Brochu, V. M. Cora, and N. D. Freitas, 'A Tutorial on Bayesian Optimization of

Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning', *ArXiv*, Dec. 2010, Accessed: May 13, 2023. [Online]. Available: https://www.semanticscholar.org/paper/A-Tutorial-on-Bayesian-Optimization-of-Expensive-to-Brochu-Cora/cd5a26b89f0799db1cbc1dff5607cb6815739fe7

[64] A. Axelrod, X. He, and J. Gao, 'Domain Adaptation via Pseudo In-Domain Data Selection', presented at the EMNLP 2011 - Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, Jan. 2011, pp. 355–362.

[65] H. Yan, Y. Guo, and C. Yang, 'Augmented Self-Labeling for Source-Free Unsupervised Domain Adaptation', presented at the NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications, Dec. 2021. Accessed: May 13, 2023. [Online]. Available: https://openreview.net/forum?id=c_XaCsX3gtA

[66] X. Liu, B. Hu, X. Liu, J. Lu, J. You, and L. Kong, 'Energy-constrained Self-training for Unsupervised Domain Adaptation', *2020 25th Int. Conf. Pattern Recognit. ICPR*, pp. 7515–7520, Jan. 2021, doi: 10.1109/ICPR48806.2021.9413284.

[67] H. Liu, J. Wang, and M. Long, 'Cycle Self-Training for Domain Adaptation', presented at the Neural Information Processing Systems, Mar. 2021. Accessed: Feb. 21, 2023. [Online]. Available: https://www.semanticscholar.org/paper/Cycle-Self-Training-for-Domain-Adaptation-Liu-Wang/e602ce17a993d33d114381be4dc54e7c19d01bce

[68] X. Liu *et al.*, 'Generative Self-training for Cross-domain Unsupervised Tagged-to-Cine MRI Synthesis', *Med. Image Comput. Comput.-Assist. Interv. MICCAI Int. Conf. Med. Image Comput. Comput.-Assist. Interv.*, vol. 12903, pp. 138–148, 2021, doi: 10.1007/978-3-030-87199-4_13.

[69] A. Blum and T. Mitchell, 'Combining labeled and unlabeled data with co-training', in *Proceedings of the eleventh annual conference on Computational learning theory*, in COLT' 98. New York, NY, USA: Association for Computing Machinery, Jul. 1998, pp. 92–100. doi: 10.1145/279943.279962.

[70] M. Chen, K. Q. Weinberger, and J. C. Blitzer, 'Co-training for domain adaptation', in *Proceedings of the 24th International Conference on Neural Information Processing Systems*, in NIPS'11. Red Hook, NY, USA: Curran Associates Inc., Dec. 2011, pp. 2456–2464.

[71] Z.-H. Zhou and M. Li, 'Tri-training: exploiting unlabeled data using three classifiers', *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1529–1541, Nov. 2005, doi: 10.1109/TKDE.2005.186.

[72] A. Søgaard, 'Simple semi-supervised training of part-of-speech taggers', in *Proceedings of the ACL 2010 Conference Short Papers*, in ACLShort '10. USA: Association for Computational Linguistics, Jul. 2010, pp. 205–208.

[73] K. Saito, Y. Ushiku, and T. Harada, 'Asymmetric tri-training for unsupervised domain adaptation', in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, in ICML'17. Sydney, NSW, Australia: JMLR.org, Aug. 2017, pp. 2988–2997.

[74] S. Ruder and B. Plank, 'Strong Baselines for Neural Semi-Supervised Learning under Domain Shift', in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia: Association for Computational Linguistics, 2018, pp. 1044–1054. doi: 10.18653/v1/P18-1096.

[75] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, 'Deep Domain Confusion: Maximizing for Domain Invariance', *ArXiv*, Dec. 2014, Accessed: Feb. 21, 2023. [Online]. Available: https://www.semanticscholar.org/paper/Deep-Domain-Confusion%3A-Maximizing-for-

Domain-Tzeng-Hoffman/1c734a14c2325cb76783ca0431862c7f04a69268

[76] A. Krizhevsky, I. Sutskever, and G. E. Hinton, 'ImageNet classification with deep convolutional neural networks', *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[77] H. Zhao, S. Zhang, G. Wu, J. P. Costeira, J. M. F. Moura, and G. J. Gordon, 'Adversarial multiple source domain adaptation: 32nd Conference on Neural Information Processing Systems, NeurIPS 2018', *Adv. Neural Inf. Process. Syst.*, vol. 2018-December, pp. 8559–8570, 2018.

[78] I. J. Goodfellow *et al.*, 'Generative Adversarial Networks'. arXiv, Jun. 10, 2014. doi: 10.48550/arXiv.1406.2661.

[79] J. Guo, D. J. Shah, and R. Barzilay, 'Multi-Source Domain Adaptation with Mixture of Experts'. arXiv, Oct. 16, 2018. doi: 10.48550/arXiv.1809.02256.

[80] S. Thrun, 'Lifelong learning algorithms', in *Learning to learn*, USA: Kluwer Academic Publishers, 1998, pp. 181–209.

[81] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, 'Gradient-based learning applied to document recognition', *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.

[82] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, 'Domain Separation Networks'. arXiv, Aug. 21, 2016. doi: 10.48550/arXiv.1608.06019.

[83] B. B. Damodaran, B. Kellenberger, R. Flamary, D. Tuia, and N. Courty, 'DeepJDOT: Deep Joint Distribution Optimal Transport for Unsupervised Domain Adaptation', in *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part IV*, Berlin, Heidelberg: Springer-Verlag, Sep. 2018, pp. 467–483. doi: 10.1007/978-3-030-01225-0_28.

[84] S. Lee, S. Cho, and S. Im, 'DRANet: Disentangling Representation and Adaptation Networks for Unsupervised Cross-Domain Adaptation'. arXiv, Mar. 28, 2021. doi: 10.48550/arXiv.2103.13447.

