Applied Intelligence (2023) 53:15909–15922
https://doi.org/10.1007/s10489-022-04288-4

Semi-supervised adversarial discriminative domain adaptation

Thai-Vu Nguyen[1,2] · **Anh Nguyen**[3] · **Nghia Le**[2,4] · **Bac Le**[1,2]

Accepted: 20 October 2022 / Published online: 29 November 2022

Abstract
Domain adaptation is a potential method to train a powerful deep neural network across various datasets. More precisely, domain adaptation methods train the model on training data and test that model on a completely separate dataset. The adversarial-based adaptation method became popular among other domain adaptation methods. Relying on the idea of GAN, the adversarial-based domain adaptation tries to minimize the distribution between the training and testing dataset based on the adversarial learning process. We observe that the semi-supervised learning approach can combine with the adversarial-based method to solve the domain adaptation problem. In this paper, we propose an improved adversarial domain adaptation method called Semi-Supervised Adversarial Discriminative Domain Adaptation (SADDA), which can outperform other prior domain adaptation methods. We also show that SADDA has a wide range of applications and illustrate the promise of our method for image classification and sentiment classification problems.

Keywords Domain adaptation · Semi-supervised domain adaptation · Semi-supervised adversarial discriminative domain adaptation

1 Introduction

Over the past few years, deep neural networks have achieved significant achievements in many applications. One of the major limitations of deep neural networks is the dataset bias or domain shift problems [1]. These phenomena occur when the model obtains good results on the training dataset; however, showing poor performance on a testing dataset or a real-world sample.

As shown in Fig. 1, because of numerous reasons (illumination, image quality, background), there is always a different distribution between two datasets, which is the main factor reducing the performance of deep neural networks. Even though various research has proved that deep neural networks can learn transferable feature representation over different datasets [2, 3], Donahue et al. [4] showed that domain shift still influences the accuracy of the deep neural network when testing these networks in a different dataset.

The solution for the aforementioned problems is domain adaptation techniques [13, 14]. The main idea of domain adaptation techniques is to learn how a deep neural network can map the source domain and target domain into a common feature space, which minimize the negative influence of domain shift or dataset bias.

The adversarial-based adaptation method [15, 16] has become a well-known technique among other domain adaptation methods. Adversarial adaptation includes two networks - an encoder and a discriminator, trained simultaneously with conflicting objectives. The encoder is trained to encode images from the original domain (source domain) and new domain (target domain) such that it puzzles the discriminator. In contrast, the discriminator

✉ Bac Le
lhbac@fit.hcmus.edu.vn

Thai-Vu Nguyen
vunguyenthai73@gmail.com

Anh Nguyen
anh.nguyen@liverpool.ac.uk

Nghia Le
nghialh@uit.edu.vn

1 Faculty of Information Technology, University of Science, Ho Chi Minh City, Vietnam

2 Vietnam National University, Ho Chi Minh City, Vietnam

3 Department of Computer Science, University of Liverpool, London, UK

4 University of Information Technology, Ho Chi Minh City, Vietnam

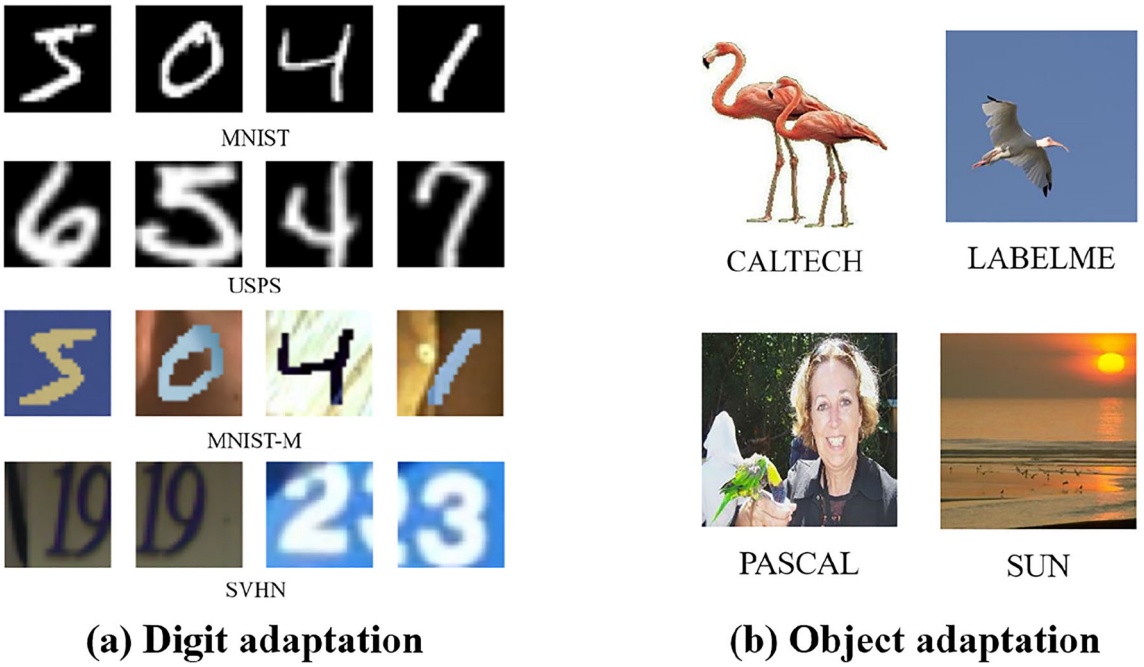

Fig. 1 Examples of images from different datasets. (a) Some digit images from MNIST [5], USPS [6], MNIST-M [7], and SVHN [8] datasets. (b) Some object images from the "bird" category in CALTECH [9], LABELME [10], PASCAL [11], and SUN [12] datasets

tries to distinguish between the source and target domain. Recently, Adversarial Discriminative Domain Adaptation (ADDA) by Tzeng et al. [16] has shown that adversarial adaptation can handle dataset bias and domain shift problems. From there, we extend the ADDA method to the semi-supervised learning context by obliging the discriminator network to predict class labels.

Semi-supervised learning [17] is an approach that builds a predictive model with a small labeled dataset and a large unlabeled dataset. The model must learn from the small labeled dataset and somehow exploit the larger unlabeled dataset to classify new samples. In the context of unsupervised domain adaptation tasks, the semi-supervised learning approach needs to take advantage of the labeled source dataset to map to the unlabeled target dataset, thereby correctly classifying the labels of the target dataset. The Semi-Supervised GAN [18] is designed to handle the semi-supervised learning tasks and inspired us to develop our model.

In this paper, we present a novel method called Semi-supervised Adversarial Discriminative Domain Adaptation (SADDA), where the discriminator is a multi-class classifier. Instead of only distinguishing between source images and target images (method like ADDA [16]), the discriminator learns to distinguish N + 1 classes, where N is the number of classes in the classification task, and the last one uses to distinguish between the source dataset or the target dataset. The discriminator focuses not only on the domain label between two datasets but also on the labeled images from the source dataset, which improves the generalization ability of the discriminator and the encoder as well as the classification accuracy.

To validate the effectiveness of our methodology, we experiment with domain adaptation tasks on digit datasets, including MNIST [5], USPS [6], MNIST-M [7], and SVHN [8]. In addition, we also prove the robustness ability of the SADDA method by using t-SNE visualization of the digit datasets, the SADDA method keeps the t-SNE clusters as tight as possible and maximizes the separation between two clusters. We also test its potential with a more sophisticated dataset, by object recognition task with CALTECH [9], LABELME [10], PASCAL [11], and SUN [12] datasets. In addition, we evaluate our method for the natural language processing task, with three text datasets including Women's E-Commerce Clothing Reviews [19], Coronavirus tweets NLP - Text Classification [20], and Trip Advisor Hotel Reviews [21]. The Python code of the SADDA method for object recognition tasks can be downloaded at https://github.com/NguyenThaiVu/SADDA.

Our contributions can be summarized as follows:

- We propose a new Semi-supervised Adversarial Discriminative Domain Adaptation method (SADDA) for addressing the unsupervised domain adaptation task.
- We illustrate that SADDA improves digit classification tasks and achieves competitive performance with other adversarial adaptation methods.
- We also demonstrate that the SADDA method can apply to multiple applications, including object recognition and natural language processing tasks.

2 Related work

Domain adaptation is an active research field, which can handle numerous problems such as imbalanced data [22], dataset bias [23], and domain shift [24]. Recent research has focused on domain adaptation from a labeled source dataset to an unlabeled target dataset, also known as unsupervised domain adaptation [25, 26]. The principle technique is minimizing the distinction between the source and target distribution [27]. Some popular approaches are Maximum Mean Discrepancy (MMD) [1], deep reconstruction classification network (DRCN) [28] or Autoencoder-based domain adaptation [29].

Adversarial-based domain adaptation With the rise of generative adversarial networks [15], the adversarial-based made huge advancements in the domain adaptation task [30–32]. Adversarial-based techniques try to achieve domain adaptation by using domain discriminators, which increases domain confusion through an adversarial process. A popular adversarial-based domain adaptation method is the Adversarial Discriminative Domain Adaptation (ADDA) by Tzeng et al. [16]. ADDA approach aims to diminish the distance between the source encoder and target encoder distributions through the domain-adversarial process. However, this method only distinguishes between the source and target domain. Instead, our SADDA method not only predicts whether the source domain or the target domain, but also classifies the label of the source dataset. More concretely, we force the adversarial-based method to the semi-supervised context. We will show that this creation can produce a more efficient classification model.

Combining adversarial-based domain adaptation with other auxiliary tasks Recently, some works have focused on combining auxiliary tasks for adversarial-based adaptation to exploit more information [33, 34]. Xavier and Bengio introduce Stacked Denoising Autoencoders [35, 36], reconstructing the merging data from numerous domains with the same network, such that the representations can be symbolized by both the source and target domain. Deep reconstruction classification network (DRCN) [28] attempts to solve two sub-problem at the same time: classification of the source data, and reconstruction of the unlabeled target data. However, these auxiliary tasks are not towards the same goal. We observe that during the adversarial process, we can classify the source or target dataset and predict the label of the source dataset simultaneously. That allows us to re-use the same output layers in the discriminator model as well as forces two discriminator models towards the same goal (Section 3.2 for more details). In addition, we also demonstrate that our SADDA method not only applies to computer vision tasks but also the natural language processing task.

3 Proposed method

3.1 Semi-supervised adversarial discriminative domain adaptation

In this section, we describe in detail our Semi-supervised Adversarial Discriminative Domain Adaptation (SADDA) method. An overview of our method can be found in Fig. 2.

In the unsupervised domain adaptation task, we already have source images $\mathbf{X}_s$ and source labels $\mathbf{Y}_s$ come from the source domain distribution $\mathbf{p}_s(x, y)$. Besides that, a target dataset $\mathbf{X}_t$ comes from a target distribution $\mathbf{p}_t(x, y)$, where the label of the target dataset is non-exist. We desire to learn a target encoder M_t and classifier C_t, which can accurately predict the target image's label. In an adversarial-based adaptation approach, we aim to diminish the distance between the source mapping distribution $(M_s(X_s))$ and target mapping distributions $(M_t(X_t))$. As a result, we can

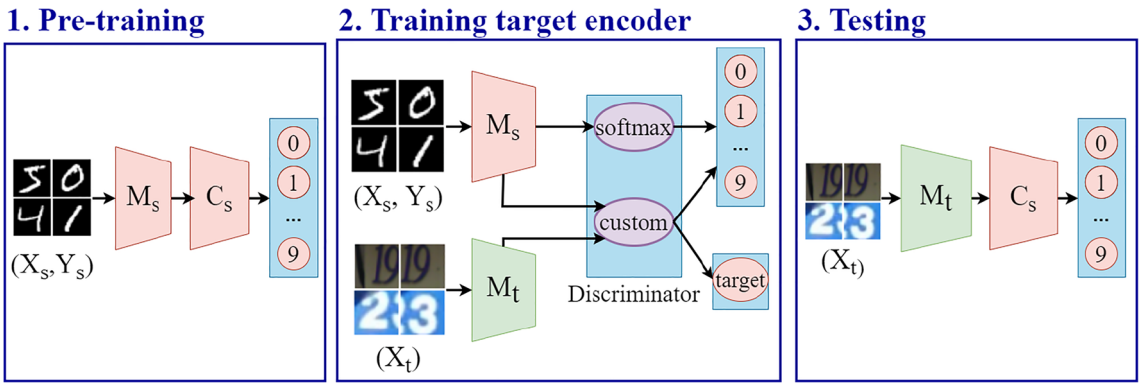

Fig. 2 An overview of the SADDA. Firstly, training the source encoder (M_s) and the classification (C_s) using the source labeled images (X_s, Y_s). Secondly, training a target encoder (M_t) through the domain adversarial process. Finally, in the testing phase, concatenate the target encoder (M_t) and the classification (C_s) to create the complete model, which will predict the label of the target dataset precisely

straightly apply the source classifier C_s to classify the target images, in other words, C = C_t = C_s. The summary process of SADDA includes three steps: pre-training, training target encoder, and testing.

Pre-training In the pre-training phase, training source encoder (M_s) and source classifier (C_s), by using the source labeled images ($\mathbf{X}_s, \mathbf{Y}_s$). This step is a standard supervised classification task, a common form can be denoted as:

$$\arg\min_{M_s, C_s} \mathcal{L}_{cls}(\mathbf{X}_s, \mathbf{Y}_s) = -\mathbb{E}_{(x,y)\sim(\mathbf{X}_s,\mathbf{Y}_s)} \sum_{n=1}^{N} y_n \log C_s(M_s(x_n)) \quad (1)$$

where $\mathcal{L}_{cls}$ is a supervised classification loss (categorical crossentropy loss), and N is the number of classes.

Training target encoder In the training target encoder phase, we first present a training discriminator process and then present a procedure for training the target encoder.

Firstly, training the discriminator (D) in two modes, each giving a corresponding output. (1) Supervised mode, where the supervised discriminator (D_{sup}) predicts N labels from the original classification task. (2) Unsupervised mode, where the unsupervised discriminator (D_{unsup}) classifies between $\mathbf{X}_s$ and $\mathbf{X}_t$. Discriminator correlates with unconstrained optimization:

$$\arg\min_{D_{sup}} \mathcal{L}_{cls}(\mathbf{X}_s, \mathbf{Y}_s) = -\mathbb{E}_{(x,y)\sim(\mathbf{X}_s,\mathbf{Y}_s)} \sum_{n=1}^{N} y_n \log D_{sup}(M_s(x_n)) \quad (2)$$

$$\arg\min_{D_{unsup}} \mathcal{L}_{adv_D}(\mathbf{X}_s, \mathbf{X}_t, M_s, M_t) = -\mathbb{E}_{x_s\sim\mathbf{X}_s} \log D_{unsup}(M_s(x_s)) - \mathbb{E}_{x_t\sim\mathbf{X}_t} \log(1 - D_{unsup}(M_t(x_t))) \quad (3)$$

In (2), $\mathcal{L}_{cls}$ is a supervised classification loss corresponding to predicting N labels from the original classification task in the source dataset ($\mathbf{X}_s$), which will update the parameter in D_{sup}. In (3), $\mathcal{L}_{adv_D}$ is an adversarial loss for unsupervised discriminator D_{unsup}, which trains (D_{unsup}) to maximize the probability of predicting the correct label from the source dataset or target dataset. One thing to notice is that the unsupervised discriminator uses a custom activation function (5), which returns a probability to determine whether a source image or target image (Section 3.2 for more details).

Secondly, training the target encoder M_t with the standard loss function and inverted labels [15]. This implies that the unsupervised discriminator D_{unsup} is fooled by the target encoder M_t, in other words, D_{unsup} is unable to determine between $\mathbf{X}_s$ and $\mathbf{X}_t$. The feedback from the unsupervised discriminator D_{unsup} allows the M_t to learn how to produce a more authentic encoder. The loss for $\mathcal{L}_{adv_M}$ can be denoted:

$$\arg\min_{M_t} \mathcal{L}_{adv_M}(\mathbf{X}_s, \mathbf{X}_t, D) = -\mathbb{E}_{x_t\sim X_t} \log D_{unsup}(M_t(x_t)) \quad (4)$$

Testing In the testing phase, we concatenate the target encoder M_t and the source classifier C_s to predict the label of target images $\mathbf{X}_t$.

3.2 The discriminator model

In this section, we describe the detail of the discriminator model and provide some arguments to prove the effectiveness of the discriminator model in the SADDA method.

In the training target encoder step, the discriminator model is trained to predict N+1 classes, where N is the number of classes in the original classification task (supervised mode) and the final class label predicts whether the sample comes from the source dataset or target dataset (unsupervised mode). The supervised discriminator and the unsupervised discriminator have different output layers but have the same feature extraction layers - via backpropagation when we train the network, updating the weights in one model will impact the other model as well.

The supervised discriminator model produces N output classes (with a softmax activation function). The unsupervised discriminator is defined such that it grabs the output layer of the supervised mode *prior softmax activation* and computes a normalized sum of the exponential outputs (custom activation). When training the unsupervised discriminator, the source sample will have a class label of 1.0, while the target sample will be labeled as 0.0. The explicit formula of custom activation [37] is:

$$D(x) = \frac{Z(x)}{Z(x)+1} \quad (5)$$

where

$$Z(x) = \sum_{n=1}^{N} \exp[l_n(x)] \quad (6)$$

The experiment of (5) is described in Table 1, and the outputs are between 0.0 and 1.0. If the probability of output value *prior softmax activation* is a large number (meaning: low entropy) then the custom activation output value is close to 1.0. In contrast, if the output probability is a small value (meaning: high entropy), then the custom activation output value is close to 0.0. Implied that the discriminator is encouraged to output a confidence class prediction for the source sample, while it predicts a small probability for the target sample. That is an elegant method allowing re-use of the same feature extraction layers for both the supervised discriminator and the unsupervised discriminator.

It is reasonable that learning the well supervised discriminator will improve the unsupervised discriminator. Moreover, training the discriminator in unsupervised mode allows the model to learn useful feature extraction capabilities from huge unlabeled datasets. As a sequence, improving the supervised discriminator will improve the

Table 1 Experimental compute on custom activation - the output of unsupervised discriminator model

Output probabilities (prior softmax)	Custom activation	Entropy
[9.0, 1.0, 1.0]	0.9999	Low
[5.0, 1.0, 1.0]	0.9935	Low
[-5.0, -5.0, -5.0]	0.0198	High

unsupervised discriminator and vice versa. Improving the discriminator will enhance the target encoder [18]. In total, this is one kind of advantage circle, in which three elements (unsupervised discriminator, supervised discriminator, and target encoder) iteratively make each other better.

3.3 Guideline for stable SADDA

In general, training SADDA is an extremely hard process, there are two losses we need to optimize: the loss for the discriminator and the loss for the target encoder. For that reason, the loss landscape of SADDA is fluctuating and dynamic (detail in Section 4.4). When implementing and training the SADDA, we find that is a tough process. To overcome the limitation of the adversarial process, we present a full architecture of SADDA. This designed architecture increases training stability and prevents non-convergence. In this section, we present the key ideas in designing the model for the image classification and the sentiment classification task. Readers can see section Appendix for more details about our designed architecture.

Image classification The design of SADDA is inspired by Deep Convolutional GAN (DCGAN) architecture [38]. The summary architecture of the SADDA method for digit recognition is shown in Fig. 5. On the one hand, the encoder is used to capture the content in the image, increasing the number of filters while decreasing the spatial dimension by the convolutional layer. On the other hand, the discriminator is symmetric expansion with the encoder by using fractionally-strided convolutions (transpose convolution).

Moreover, our recommendation for efficient training SADDA in the image classification task:

- Replace any pooling layers with convolution layers (or transposed convolution) with strides larger than 1.
- Remove fully connected layers in both encoder and discriminator (except the last fully connected layers, which are used for prediction).
- Use ReLU activation [39] in the encoder and LeakyReLU activation [39] (with alpha=0.2) in the discriminator.

Sentiment classification The design of the SADDA method for sentiment classification is inspired by the architecture called Autoencoders LSTM [40, 41]. The summary architecture of the SADDA method for sentiment classification is demonstrated in Fig. 7. In general, the architecture of the model used in the sentiment classification task has many similarities with the architecture used in image classification. Firstly, we remove fully connected layers in both the encoder and the discriminator. Instead, we use the Long Short Term Memory [42] (LSTM) to handle sequences of text data. Secondly, the network is organized into an architecture called the Encoder-Decoder LSTM, with the Encoder LSTM being the encoder block and the Decoder LSTM being the discriminator block respectively. The Encoder-Decoder LSTM was built for the NLP task where it illustrated state-of-the-art performance, such as machine translation [43]. From the empirical, we find that the Encoder-Decoder is suitable for the unsupervised domain adaptation task.

4 Experiments

In this section, we evaluate our SADDA method for unsupervised domain adaptation tasks in three scenarios: digit recognition, object recognition, and sentiment classification.

In the experiments, we focus on probing how the SADDA method improves the unsupervised domain adaptation task. For this purpose, we only choose shallow architecture rather than a deep network. We leave the sophisticated design for a future job.

4.1 Digit recognition

4.1.1 Datasets and domain adaptation scenarios

We evaluate SADDA on various unsupervised domain adaptation experiments, examining the following popular used digits datasets and settings (the visualization is in Fig. 1):

MNIST $\longleftrightarrow$ **USPS**: MNIST [5] includes 28x28 pixels, which are grayscale images of digit numbers. USPS [6] is a digit dataset, which contains 9298 grayscale images. The image is 16x16 pixels. In this experiment, we follow the evaluation protocol of [44].
MNIST $\rightarrow$ **MNIST-M**: MNIST-M [7] is made by merging MNIST digits with the patches arbitrarily extracted from color images of BSDS500 [45]. In this

Table 2 Experimental results on unsupervised domain adaptation on digit datasets

Method	mnist→usps	usps→mnist	mnist→mnist-m	svhn→mnist
Source only	78.9	57.1 ± 1.7	63.6	60.1 ± 1.1
DANN [7]	85.1	73.0 ± 2.0	77.4	73.9
DRCN [28]	91.8 ± 0.1	73.7 ± 0.1	-	82.0 ± 0.2
ADDA [16]	89.4 ± 0.2	90.1 ± 0.8	-	76.0 ± 1.8
SBADA-GAN [47]	97.6	95.0	99.4	76.1
SHOT [48]	98.0	98.4	-	98.9
DFA-MCD [49]	98.6	96.6	-	98.9
SADDA (our)	98.1	97.8	78.2	86.5

The results are not re-implement, instead, we select based on the available result in the previous publication (some experimental results have the standard deviation because that publication has the standard deviation while others do not.)

experiment, we set the input size is 28x28x3 pixels, and we follow the evaluation protocol of [44]

SVHN → MNIST: The Street View House Number (SVHN) [8] is a digit dataset, which contains 600000 32×32 RGB images. In this experiment, we convert the SVHN dataset to grayscale images and resize the MNIST images into 32x32 grayscale images. We use the evaluation protocol of [28].

4.1.2 Implementation details

The SADDA model is trained with different learning rates in different phases. In the pre-training phases, this is a standard classification task, we use a learning rate is 0.001 in our experiment. In the training target encoder phases, we suggested a learning rate of 0.0002 as well as an Adam optimizer [46] and setting the β_1 equal to 0.5 to help stabilize training. In the LeakyReLU activation, we set α = 0.2 in the whole model.

In this experiment, the encoder consists of four convolutional layers with 4 x 4 kernel size, 2 x 2 strides, same padding, and ReLU activation. The number of filters for four convolution layers are 32, 64, 128, and 256, respectively. The target encoder has the same architecture as the source encoder, and the source encoder is used as an initialization for the target encoder.

The classifier takes the outputs of the encoder as input. Next, a fully connected layer with 100 feature channels,

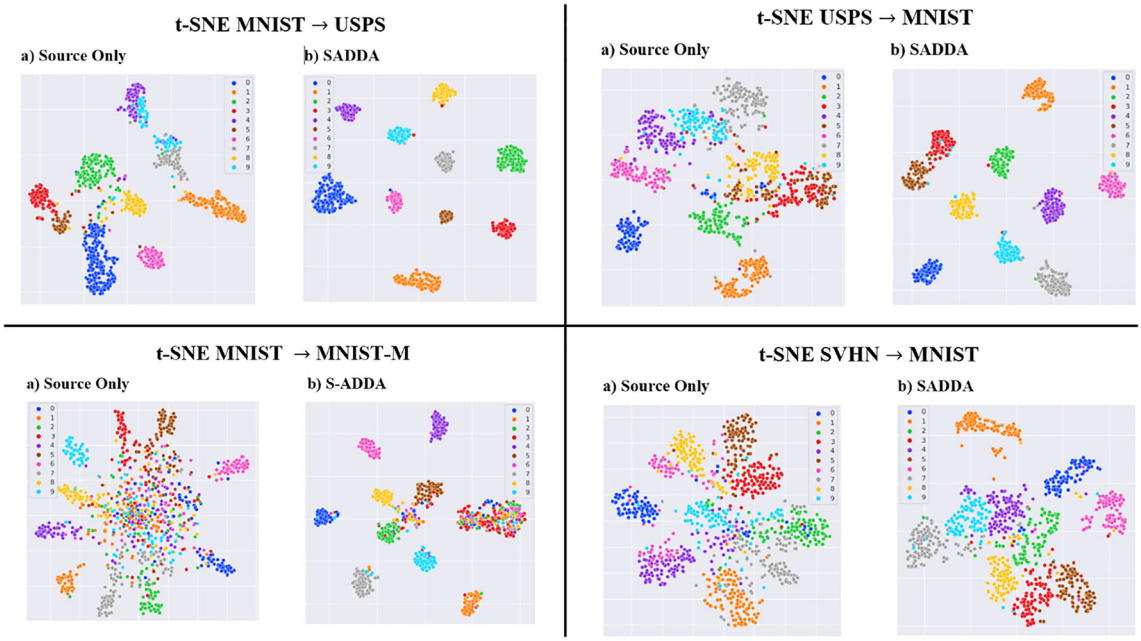

Fig. 3 t-SNE embedding of digit classification, using (2 x 2 x 256) dimensional representation, with Source only (on the left) and SADDA (on the right) on the target dataset. Note that SADDA minimizes intra-class distance and maximizes inter-class distance

followed by ReLU activation. Finally, the fully connected layer with ten feature channels and the softmax activation.

In the training target encoder phases, the outputs of the encoder serve as the input of the discriminator. The discriminator consists of four Transpose Convolutional layers with 4 x 4 kernels size, 2 x 2 strides, the same padding, and the Leaky ReLU activation (alpha = 0.2). The number of kernels for four Transpose Convolutional is 256, 128, 64, and 32, respectively. We illustrate the overall architecture in Fig. 5.

4.1.3 Results on digit datasets

In this experiment, we compare our SADDA method against multiple state-of-the-art unsupervised domain adaptation methods.

Experimental results are shown in Table 2. In the real-world dataset SVHN → MNIST, the SADDA model showed approximately 26% improvement over the Source only model, 10% more than the ADDA method [16]. In addition, in the first two experiments (MNIST → USPS and USPS → MNIST), the SADDA method achieves extremely high accuracy, which results in 98.1% and 97.8% respectively. However, SADDA has a little lower accuracy than other methods like SBADA-GAN [47], SHOT [48], and DFA-MCD [49] in some experiments.

Although, our method did not achieve the highest accuracy in any of the experiments. Our method still has competitive accuracy when compared with the state-of-the-art methods in the last two years (SHOT [48], DFA-MCD [49]). For example, our SADDA method compared with the DFA-MCD method, in the MNIST → USPS experiment, we have lower accuracy (98.1% versus 98.6%); however, in the USPS → MNIST experiment, we achieved a higher accuracy (SADDA achieved 97.8% compared to 96.6%).

For further insight into the SADDA model effect on the digit classification tasks, we use t-SNE [50] to visualize the 2D point of the last encoder layer of SADDA (as described in Fig. 3). Ten labels are from 0 to 9 corresponding, and 100 samples per label. The domain invariance is determined by the degree of overlap between features. Regarding the Source only model, the distribution and the density are messy. In contrast, the SADDA method splits different labels into different regions, and the overlap is more prominent.

4.2 Object recognition

4.2.1 Datasets and preprocessing

In this subsection, we present the experiments for evaluating the SADDA method. The experiment is performed on the VLCS [23] dataset, including PASCAL VOC2007 (V) [11], LABELME (L) [10], CALTECH (C) [9], and SUN (S) [12] datasets. Each dataset contains five categories: bird, car, chair, dog, and person. Since the number of images per class is not equal, we use data augmentation techniques to balance the number of images. In this experiment, we use the Albumentations library [51] to increase to 5000 images per class. We divide the dataset into a training set (60%), validation set (20%), and test set (20%).

Table 3 The accuracy (%) on the VLCS dataset

	LABELME	CALTECH	SUN
Source only	33.26	45.10	33.78
SADDA	37.73	55.30	36.21
Train on target	86.52	99.27	88.26
(a)Source domain: PASCAL			
	PASCAL	CALTECH	SUN
Source only	28.73	32.75	27.71
SADDA	32.71	39.36	31.01
Train on target	75.28	99.27	88.26
(b)Source domain: LABELME			
	PASCAL	LABELME	SUN
Source only	26.71	28.27	31.62
SADDA	31.82	31.39	37.67
Train on target	75.28	86.52	88.26
(c)Source domain: CALTECH			
	PASCAL	LABELME	CALTECH
Source only	29.72	25.87	39.66
SADDA	33.04	27.35	41.52
Train on target	75.28	86.52	99.27
(d)Source domain: SUN			

The detailed architecture is shown in Fig. 6. The rest of the other installation (optimization algorithm, learning rate) is the same as Section 4.1.2.

In the experiments, one dataset is used as the source domain and the rest is used as the target domain, resulting in four different cases (Table 3). In addition, we do not compare our SADDA model with other domain adaptation methods due to different setups.

4.2.2 Results

The results on the VLCS are shown in Table 3. Source only is a model that only trains on the source dataset without using any domain adaptation methods. Overall, the accuracy when applying the SADDA method overcomes the Source only model in all cases. In some specific cases like PASCAL → CALTECH, the classification accuracy goes from 45.10 to 55.30 (improving approximately 10%). In case LABELME → CALTECH, the accuracy grows from 32.75% to 39.36%.

Examining the results in Table 3, the Source only model has low accuracy, which reveals that the domain shift is quite large. In other words, the Source only model does not learn any knowledge about the source dataset to predict the target dataset. In contrast, the SADDA model learns a more useful feature representation, leading to higher accuracy when performing a prediction on the target dataset.

Although the SADDA method has certain improvements compared to the Source only method, the accuracy of the SADDA method is still low and not ideal. For further comparison, we also test the hypothesis situation where the target labels are present (the train on target model). There is still a big gap between the accuracy of the SADDA method and the train on target method.

4.3 Sentiment classification

4.3.1 Datasets and preprocessing

In this subsection, we evaluate the SADDA method for the sentiment classification task. We use three sentimental datasets, including Women's E-Commerce Clothing Reviews [19], Coronavirus tweets NLP - Text Classification [20], and Trip Advisor Hotel Reviews [21]:

Women's E-Commerce Clothing Reviews [19] This is real commercial data, where the reviews are written by customers. In this task, we only use two features called *Review Text* (the raw text review) and *Rating* (the positive integer for the product, provided by the customer from 1 Worst to 5 Best). Regarding the *Rating*, we relabel into the sets {positive, neutral, negative} with the following rule: if a review is greater than 3, it is considered a positive comment; if a review is equal to 3, it is considered a neutral comment; if a review is less than 3, it is considered a negative comment.

Coronavirus tweets NLP - Text Classification [20] The tweets were downloaded from Twitter and tagged manually. Although there are four columns in total, we only use the *Original Tweet* feature and *Label* in our experiment. In the case of *Label*, the original label includes Extremely Negative, Negative, Neutral, Positive, and Extremely Positive. However, we convert to the sets {positive, neutral, negative} respectively.

Trip Advisor Hotel Reviews [21] This contains reviews crawled from the travel company called Tripadvisor. The dataset contains two features, including *Review Text* and *Rating* (the positive integer from 1 Worst to 5 Best). Regarding *Rating*, we process the same as the case Women's E-Commerce Clothing Reviews above.

In all three datasets, we perform the following text preprocessing steps: removing the punctuation, URL, hashtags, mentions, and stop words (with the support of the NLTK [52] library). We limit the input sentence to a max length equal to 50. The GloVe [53] word embedding is applied to map the word in the text review to the vector space.

Because the number of samples per class is not balanced, we use the data augmentation techniques to balance the number of samples - with the support of the TextAugment [54] library. Particularly, we perform data augmentation such that each class has up to 20 000 samples. The dataset is divided into a training set (60%), validation set (20%), and test set (20%).

4.3.2 Experiments and results

For this experiment, our architecture is illustrated in Fig. 5. Elements in that architecture such as the LSTM layer and the Repeat Vector layer are implemented by the TensorFlow library with default settings. The optimization algorithms and learning rates are set up as in Section 4.1.2. In addition, we do not attempt to fine-tune the architecture and leave it for future work.

The results of our experiment are provided in Table 4. Compared with the Source only model, the SADDA method shows a little improvement in the accuracy of this sentiment analysis task. For a certain experiment, like T → W, the classification accuracy goes from 41.07% to 49.28%. However, not all experiments improve, such as experiment T → C, the accuracy even dropped a bit (from 38.46% down to 38.05%). Additionally, a comparison with the "Train on target" model exposes that the SADDA model is far from

Table 4 The accuracy (%) of unsupervised domain adaptation on the sentiment classification task

	W → C	W → T	C → W	C → T	T → W	T → C
Source only	37.97	50.11	45.91	49.93	41.07	38.46
SADDA	43.88	55.54	48.02	56.10	49.28	38.05
Train on target	79.01	96.10	93.02	96.10	93.02	79.01

In the table, there are three datasets: Women's E-Commerce Clothing Reviews (W) [19], Coronavirus tweets (C) [20], Trip Advisor Hotel Reviews (T) [21]

the ideal model. We hope that is the motivation for future development.

4.4 Challenge and convergence analysis

In this subsection, we will discuss the challenge of training a stable SADDA model and how to trigger the early stopping of the training progress to achieve the convergent state.

In the SADDA model, we have three stages: Pre-training, Training target encoder, and Testing. The difficulty comes from the Training target encoder process. The reason is that both the discriminator and target encoder are trained simultaneously in that procedure, which can lead to updating the parameter of one model will reduce the performance of the other model. More concretely, there are two loss functions we need to optimize: the discriminator loss and the adversarial loss. The discriminator loss (3) is the loss when the unsupervised discriminator predicts the source or target samples. The adversarial loss (4) is the loss of the target encoder when training with the inverted labels.

When training the target encoder, we do not try to find a minimum value for either discriminator loss or adversarial loss. Instead, we are looking for a Nash equilibrium state for both losses [37]. In practice, we observe the discriminator loss and adversarial loss after each epoch, when both loss values no longer change, we trigger an early stopping. Early stopping is the technique to stop the training process at a certain point before the model overfits the training dataset and has poor performance on the test set. In that case, we consider that our model has converged. Keep in mind that the loss of 0.0 in the discriminator loss or the adversarial loss during the training process is a failure mode.

Regarding Fig. 4, the convergence point is in the epoch 20. At that point, we will stop the training process because the discriminator loss and the adversarial loss are saturated at around 0.33 and 2.30 respectively. In that experiment (and

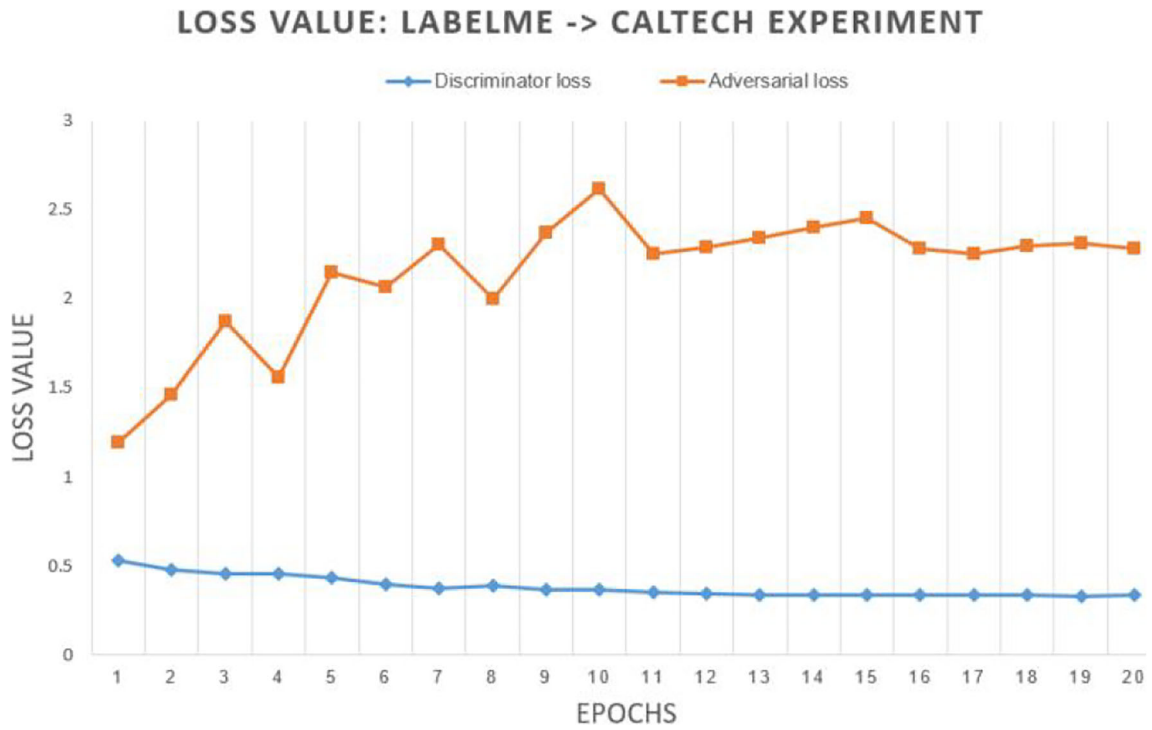

Fig. 4 The discriminator loss and adversarial loss in the LABEL → CALTECH experiment

other object recognition tasks), it took around 1 hour on a single Tesla T4 GPU to complete the training procedure.

5 Conclusion

We proposed a more stable and high-accuracy architecture for training adversarial-based domain adaptation methods. The key idea of this approach is to train discriminators in two modes: supervised mode and unsupervised mode. Moreover, utilize this to create a more efficient target encoder, which will help improve the classification accuracy.

While the SADDA method has demonstrated an improvement in many tasks like image classification or sentiment classification, there are still open challenges. Particularly, the SADDA model in object recognition and sentiment classification is far from the desired accuracy model. We hope that the intuition of this research will facilitate further advances in domain adaptation tasks.

Appendix

In this appendix section, we present in detail the design architecture of our SADDA method in three experiments,

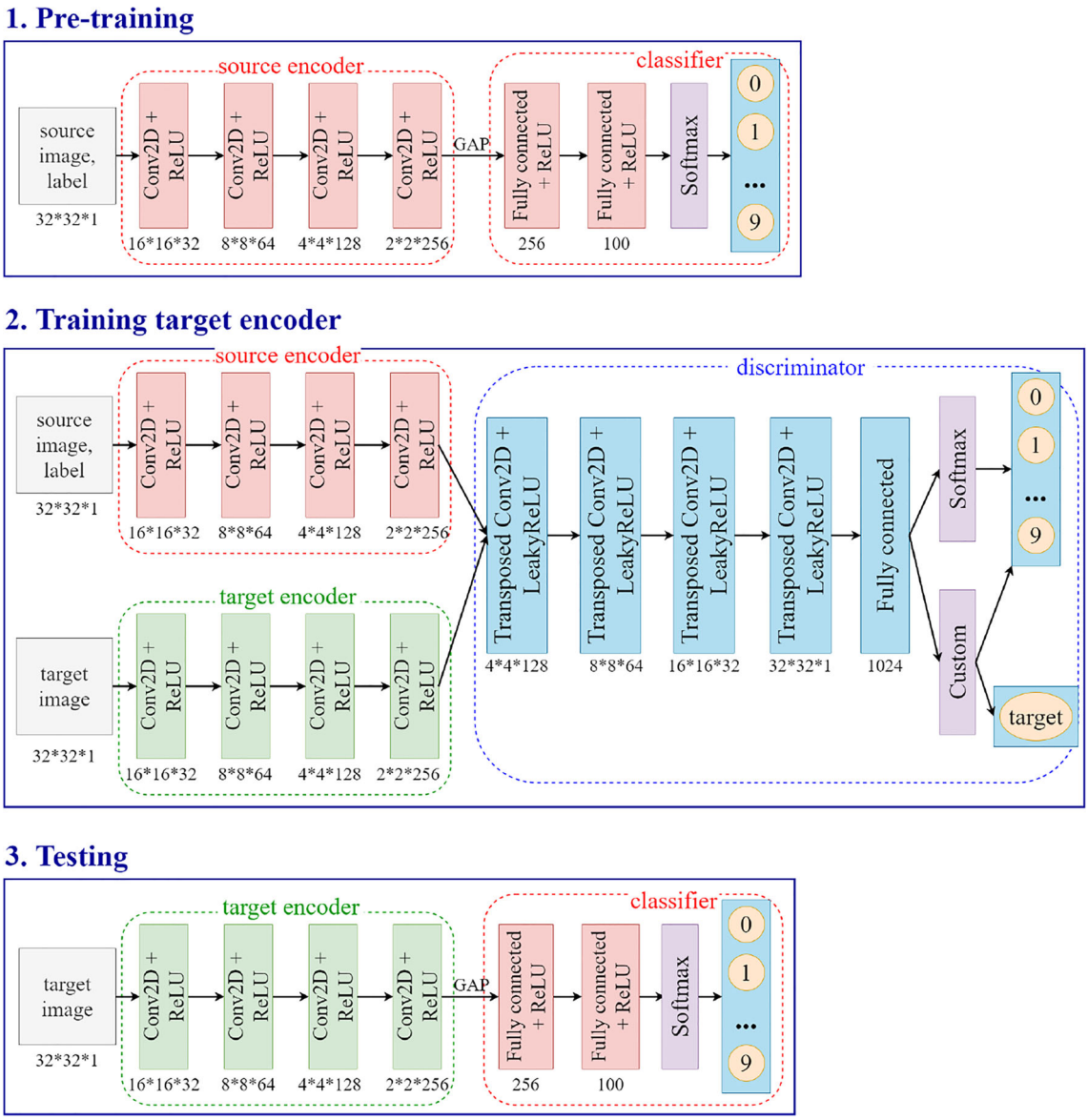

Fig. 5 The overview of the SADDA method for the digit recognition task. We found that Global Average Pooling (GAP) [55] increased model stability and reduce the number of parameter

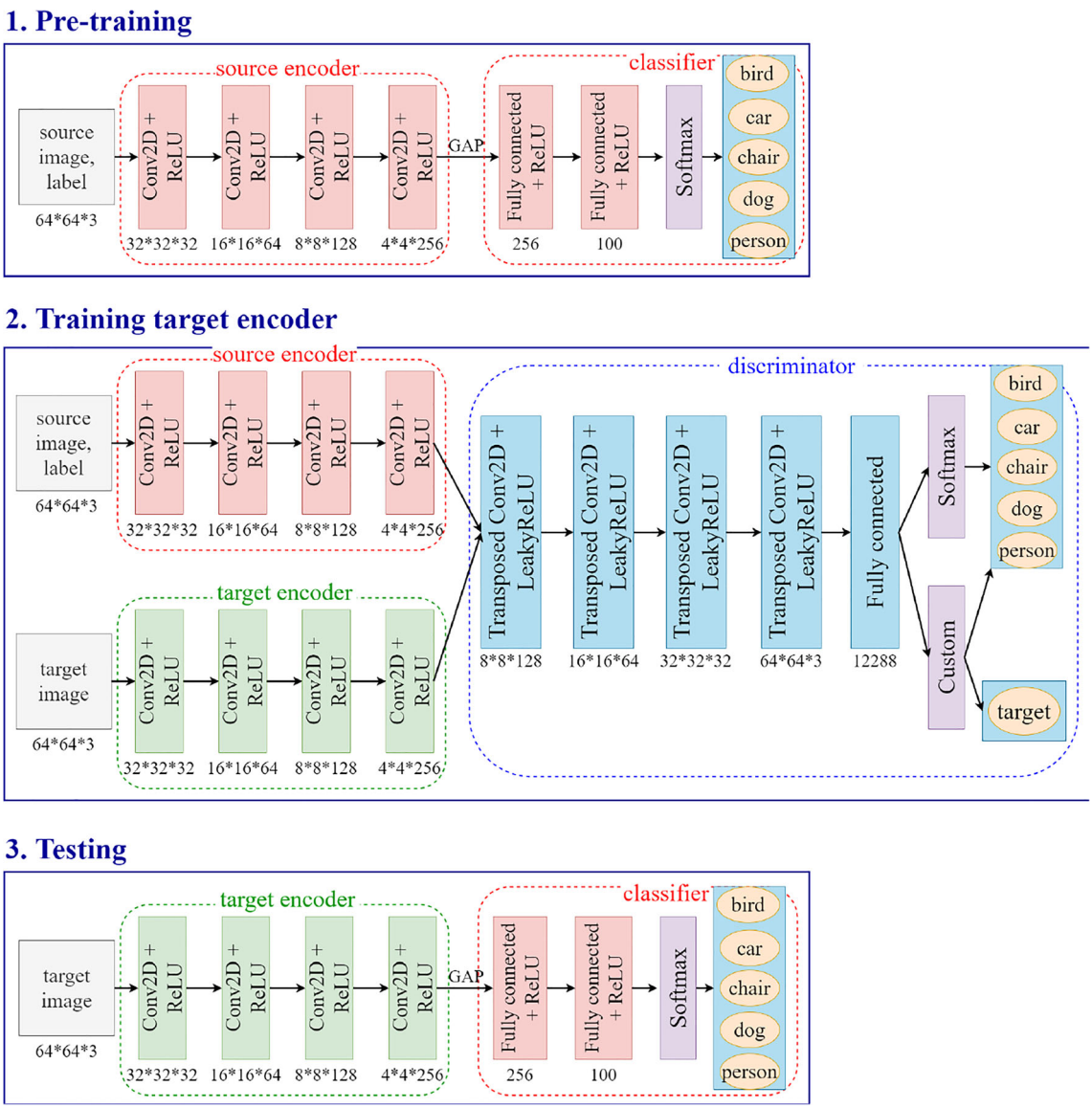

Fig. 6 The overview of the SADDA method for the object recognition task on the VLCS [23] dataset. With the input image's shape is 64 x 64 x 3

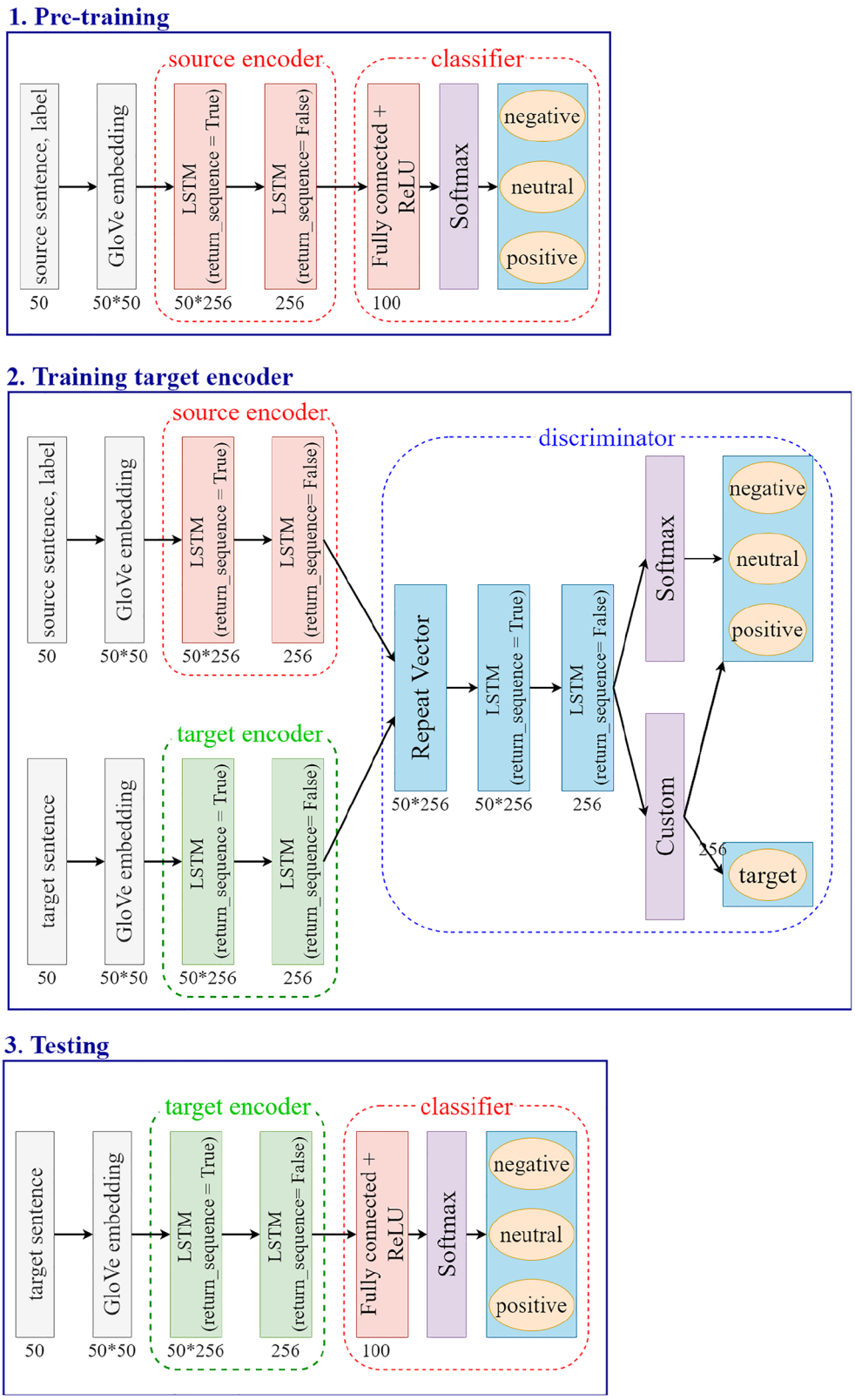

Fig. 7 The overview of the SADDA method for the sentiment classification task. The input sentence has a max length equal to 50. In the design above, to prevent overfitting, the LSTM layer is always followed by the dropout layer with 0.2 rates. The numbers under the particular layer are the output shape of that layer

including digit recognition, object recognition, and sentiment classification.

References

1. Gretton A, Smola A, Huang J, Schmittfull M, Borgwardt K, Schölkopf B (2009) Covariate shift and local learning by distribution matching, pp 131–160. Cambridge, MA USA: MIT Press
2. Long M, Cao Y, Wang J, Jordan MI (2015) Learning transferable features with deep adaptation networks. arXiv:1502.02791
3. Nguyen A, Nguyen N, Tran K, Tjiputra E, Tran QD (2020) Autonomous navigation in complex environments with deep multimodal fusion network. In: 2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)
4. Donahue J, Jia Y, Vinyals O, Hoffman J, Zhang N, Tzeng E, Darrell T (2013) Decaf: A deep convolutional activation feature for generic visual recognition
5. Lecun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
6. Hull JJ (1994) A database for handwritten text recognition research. IEEE Trans Pattern Anal Mach Intell 16(5):550–554
7. Ganin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, Marchand M, Lempitsky V (2015) Domain-adversarial training of neural networks
8. Netzer Y, Wang T, Coates A, Bissacco A, Wu B, Ng AY (2011) Reading digits in natural images with unsupervised feature learning. In: NIPS Workshop on deep learning and unsupervised feature learning 2011
9. Fei-Fei L, Fergus R, Perona P (2006) One-shot learning of object categories. IEEE Trans Pattern Anal Mach Intell 28(4):594–611
10. Russell BC, Torralba A, Murphy KP, Freeman WT (2007) Labelme: a database and web-based tool for image annotation. Int J Comput Vis 77:157–173
11. Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A (2022) The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html
12. Choi MJ, Lim JJ, Torralba A, Willsky AS (2010) Exploiting hierarchical context on a large database of object categories. In: 2010 IEEE Computer society conference on computer vision and pattern recognition, pp 129–136
13. Gheisari M, Baghshah MS (2015) Unsupervised domain adaptation via representation learning and adaptive classifier learning. Neurocomput 165:300–311
14. Pan SJ, Tsang IW, Kwok JT, Yang Q (2011) Domain adaptation via transfer component analysis. IEEE Trans Neural Netw 22(2):199–210
15. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, eds.), vol 27, Curran Associates., Inc.
16. Tzeng E, Hoffman J, Saenko K, Darrell T (2017) Adversarial discriminative domain adaptation. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR), pp 2962–2971
17. Reddy Y, Pulabaigari V (2018) E. B Semi-supervised learning: a brief review. Int J Eng Technol 7:81, 02
18. Odena A (2016) Semi-supervised learning with generative adversarial networks
19. Nicapotato (2018) Women's e-commerce clothing reviews
20. Miglani A (2020) Coronavirus tweets nlp - text classification
21. Alam MH, Ryu W-J, Lee S (2016) Joint multi-grain topic sentiment: modeling semantic aspects for online reviews. Inf Sci 339:206–223
22. Krawczyk B (2016) Learning from imbalanced data: open challenges and future directions. Prog Artif Intell 5:221–232
23. Torralba A, Efros AA (2011) Unbiased look at dataset bias, CVPR 2011, pp 1521–1528
24. Chi W, Dagnino G, Kwok TM, Nguyen A, Kundrat D, Abdelaziz ME, Riga C, Bicknell C, Yang G-Z (2020) Collaborative robot-assisted endovascular catheterization with generative adversarial imitation learning. In: 2020 IEEE International conference on robotics and automation (ICRA), pp 2414–2420 IEEE
25. Kouw WM, Loog M (2021) A review of domain adaptation without target labels. IEEE Trans Pattern Anal Mach Intell 43:766–785
26. Margolis A (2011) A literature review of domain adaptation with unlabeled data, Rapport Technique, University of Washington, p 01
27. Wang M, Deng W (2018) Deep visual domain adaptation: a survey. Neurocomputing 312:135–153
28. Ghifary M, Kleijn WB, Zhang M, Balduzzi D, Li W (2016) Deep reconstruction-classification networks for unsupervised domain adaptation
29. Deng J, Zhang Z, Eyben F, Schuller B (2014) Autoencoder-based unsupervised domain adaptation for speech emotion recognition. IEEE Signal Process Lett 21(9):1068–1072
30. Long M, Zhu H, Wang J, Jordan MI (2016) Deep transfer learning with joint adaptation networks
31. Long M, Cao Z, Wang J, Jordan MI (2017) Conditional adversarial domain adaptation
32. Saito K, Watanabe K, Ushiku Y, Harada T (2018) Maximum classifier discrepancy for unsupervised domain adaptation
33. Ghifary M, Kleijn WB, Zhang M, Balduzzi D (2015) Domain generalization for object recognition with multi-task autoencoders
34. Bousmalis K, Trigeorgis G, Silberman N, Krishnan D, Erhan D (2016) Domain separation networks
35. Glorot X, Bordes A, Bengio Y (2011) Domain adaptation for large-scale sentiment classification:, A deep learning approach
36. Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol P-A (2010) Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res 11:3371–3408
37. Salimans T, Goodfellow I, Zaremba W, Cheung V, Radford A, Chen X (2016) Improved techniques for training gans
38. Radford A, Metz L, Chintala S (2015) Unsupervised representation learning with deep convolutional generative adversarial networks
39. Maas AL, Hannun AY, Ng AY (2013) Rectifier nonlinearities improve neural network acoustic models. In: ICML Workshop on deep learning for audio, speech and language processing
40. Srivastava N, Mansimov E, Salakhutdinov R (2015) Unsupervised learning of video representations using lstms
41. Brownlee J (2020) A gentle introduction to lstm autoencoders
42. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 12:1735–80
43. Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), (Doha, Qatar), pp 1724–1734, Association for Computational Linguistics
44. Bousmalis K, Silberman N, Dohan D, Erhan D, Krishnan D (2016) Unsupervised pixel-level domain adaptation with generative adversarial networks
45. Arbeláez P, Maire M, Fowlkes C, Malik J (2011) Contour detection and hierarchical image segmentation. IEEE Trans Pattern Anal Mach Intell 33(5):898–916

46. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization
47. Russo P, Carlucci FM, Tommasi T, Caputo B (2017) From source to target and back: symmetric bi-directional adaptive gan
48. Liang J, Hu D, Feng J (2020) Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In: international conference on machine learning (ICML), pp 6028–6039
49. Wang J, Chen J, Lin J, Sigal L, de Silva CW (2021) Discriminative feature alignment: Improving transferability of unsupervised domain adaptation by gaussian-guided latent alignment. Pattern Recognition, p 107943
50. van der Maaten L, Hinton G (2008) Visualizing data using t-sne. J Mach Learn Res 9(86):2579–2605
51. Buslaev A, Iglovikov VI, Khvedchenya E, Parinov A, Druzhinin M, Kalinin AA (2020) Albumentations: fast and flexible image augmentations, information, vol 11, no. 2
52. Loper E, Bird S (2002) Nltk: the natural language toolkit, CoRR, vol. cs.CL/0205028, 07
53. Pennington J, Socher R, Manning C (2014) GLoVe: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), (Doha, Qatar), pp 1532–1543, Association for Computational Linguistics
54. Marivate V, Sefara T (2020) Improving short text classification through global augmentation methods
55. Lin M, Chen Q, Yan S (2013) Network in network