applied sciences

MDPI

Article

DA-FER: Domain Adaptive Facial Expression Recognition

Mei Bie [1,2], Huan Xu [1], Quanle Liu [1], Yan Gao [1], Kai Song [2] and Xiangjiu Che [1,*]

[1] College of Computer Science and Technology, Jilin University, Changchun 130012, China; biemei@ccsfu.edu.cn (M.B.); xuhuan20@mails.jlu.edu.cn (H.X.); qlliu18@mails.jlu.edu.cn (Q.L.); yan19860628@163.com (Y.G.)
[2] Institute of Education, Changchun Normal University, Changchun 130032, China
* Correspondence: chexj@jlu.edu.cn

Abstract: Facial expression recognition (FER) is an important field in computer vision with many practical applications. However, one of the challenges in FER is dealing with small sample data, where the number of samples available for training machine learning algorithms is limited. To address this issue, a domain adaptive learning strategy is proposed in this paper. The approach uses a public dataset with sufficient samples as the source domain and a small sample dataset as the target domain. Furthermore, the maximum mean discrepancy with kernel mean embedding is utilized to reduce the disparity between the source and target domain data samples, thereby enhancing expression recognition accuracy. The proposed Domain Adaptive Facial Expression Recognition (DA-FER) method integrates the SSPP module and Slice module to fuse expression features of different dimensions. Moreover, this method retains the regions of interest of the five senses to accomplish more discriminative feature extraction and improve the transfer learning capability of the network. Experimental results indicate that the proposed method can effectively enhance the performance of expression recognition. Specifically, when the self-collected Selfie-Expression dataset is used as the target domain, and the public datasets RAF-DB and Fer2013 are used as the source domain, the performance of expression recognition is improved to varying degrees, which demonstrates the effectiveness of this domain adaptive method.

Keywords: domain adaptive; facial expression recognition; transfer learning; classification

Citation: Bie, M.; Xu, H.; Liu, Q.; Gao, Y.; Song, K.; Che, X. DA-FER: Domain Adaptive Facial Expression Recognition. *Appl. Sci.* **2023**, *13*, 6314. https://doi.org/10.3390/app13106314

Academic Editors: Pedro J. S. Cardoso, João M. F. Rodrigues and Cristina Portalés Ricart

Received: 13 April 2023
Revised: 11 May 2023
Accepted: 20 May 2023
Published: 22 May 2023

1. Introduction

Facial expression recognition (FER) is an important field in computer vision that has gained significant attention recently. One of the main challenges in FER is dealing with small sample data, which refers to situations where the number of samples available for training machine learning algorithms is limited. Training deep learning models in such situations becomes challenging as they are susceptible to overfitting [1]. To address this issue, we propose a transfer learning method to tackle the problem of small sample sizes in expression recognition tasks by utilizing public datasets with sufficient samples as the upstream task for network training. Furthermore, apply domain adaptive strategy to promote the training of target domain datasets. As a result, the network model is more tailored to the downstream task to improve the accuracy of facial expression recognition. The main contributions of this paper can be summarized as follows.

1. Aiming at the problem of small sample size problem in the field of expression recognition, we use RAF-DB as the source domain datasets and the self-collected datasets as the target domain. A domain adaptive facial expression recognition (DA-FER) method is simultaneously implemented in both the source and target domains. The features are extracted to minimize differentiation and improve network accuracy.

2. Integrate the SSPP module and Slice module to fuse expression features of different dimensions. As a result, the regions of interest of the five senses are retained selectively. In

addition, the discriminative features are extracted, which enhances the domain transfer capability of the network.

3. The proposed DA-FER method shows the improvement in expression recognition when the self-collected datasets are used as the target domain, and the public datasets RAF-DB and Fer2013 are used as the source domain, demonstrating the effectiveness of this domain adaptive method.

2. Related Works

2.1. Small Sample Size Problem in the Field of Expression Recognition

The small sample size problem means that only a limited number of samples are available for learning high-dimensional features to complete the classification task. In facial expression recognition, it is very time-consuming and laborious to collect, select and annotate many expression images. Wang et al. [2] applied a data augmentation technique to get fine-grained expression recognition. Compositional Generative Adversarial Network (Comp-GAN) generates and edits realistic images while preserving identity information. Shome and Kar [3] proposed a joint learning approach to facial expression recognition, using a small number of labeled samples in each training round to train a local model. The weights of the local model are then passed and integrated into the global optimization model, achieving good experimental results. Zhu et al. [4] designed the Convolutional Relation Network (CRN) to learn a metric space for the problem of small sample size expression samples in the wild. The method adopted Jesnsen-Shannon divergence to constrain and extract the discriminative expression features.

2.2. Domain Adaptive Method for Facial Expression Recognition

Transfer learning describes how humans can apply knowledge and skills they have already mastered to other related problem-solving processes. For example, learning boxing first can make it easier to learn sparring, and learning badminton first can make it easier to play tennis. This process of promoting new knowledge learning is also applicable to machine learning. When transfer learning applies existing experience to a new task, there are four transfer methods: instance-based, feature representation, model-based and relation knowledge-based [5]. The domain adaptive method is one of the feature-based transfer learning methods. The hybrid improved unsupervised cross-domain adaptation method proposed by Jin improved the separability of source domain samples [6]. It minimized the intra-class sample distance and maximized the inter-class sample distance. Alvarez-Pato et al. [7] fused multimodal information for data fusion and analysis. Peng et al. [8] proposed an AU-guided domain adaptative method to alleviate the problem of annotation bias in different expression datasets. Kong et al. [9] experimented with domain adaptive networks under various settings of source domain datasets and target domain datasets, including with-setting, cross-setting, and mixed-setting. Their research found that the variability between different datasets is very high, which led to different experimental results when they were used as the source datasets for transfer learning. Regardless of the above settings, training the network with a larger number of datasets could improve the performance of transfer learning. However, the practical challenges of computational power and time cost must be considered. To address the domain shift problem, Yuhao Xie et al. [10] proposed a consistent global-local representation learning approach and a semantic learning framework. The network generated pseudo-labels with high confidence to facilitate cross-domain facial expression recognition.

3. Proposed Method

For the small sample problem in the field of expression recognition, we use RAF-DB as the source domain dataset and the self-collected Selfie-Expression datasets as the target domain and apply the domain adaptive method. We simultaneously extracted features from both the source and target domains to minimize feature discretization and improve the accuracy of the expression recognition model.

3.1. Learning Process of Domain Adaptive Method

Figure 1 illustrates the learning process of domain adaptation methods, focusing on the design of the loss function during training. It comprises three main components: input, training, and output [11]. Specifically, all the labels of source domain datasets are used in the input part, whereas target domain datasets are not used [12]. Only during verification the labels of the target domain are employed, and the accuracy is computed by comparing the predicted label generated by the network with the actual label of the target domain [13]. Consequently, domain transfer learning is fundamentally an unsupervised learning technique [14].

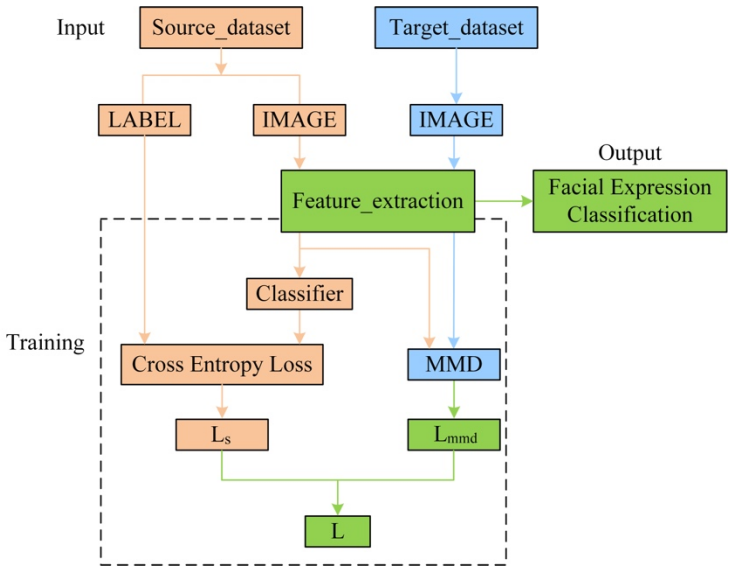


Figure 1. Learning process of domain adaptive method.

The network loads images of the source domain and target domain, and they share a Feature_ extraction network [15]. In the orange part of the flow chart, the source domain image is fed into the feature_extraction network, followed by a classifier that produces a predicted label. The predicted label is then compared with the ground-truth label and the loss function L_s is calculated using the cross-entropy function. In contrast, the blue branch of the flow chart involves processing the target domain image through the feature extraction network without the classifier. Instead, the difference between the source and target domain features is calculated using the maximum mean difference (MMD) method and the loss function L_{mmd} is obtained. The overall objective of network learning is to minimize the values of both L_s and L_{mmd} to achieve effective transfer learning. That can be described as:

$$L = L_s + \lambda L_{mmd} \quad (1)$$

The weight parameter, denoted by λ, calculates the weight of the discrepancy. While L_s captures the classification loss of the source domain, L_{mmd} measures the difference between the source domain and the target domain. However, the two loss functions cannot be assigned the same weight [16]. This is because the true labels of the source domain are involved in the network training, and the accuracy of the source domain classification should be prioritized. Therefore, λ is considered a hyperparameter to control the balance between L_s and L_{mmd}. It is typically set to a value between 0 and 1. In the experiments conducted in this study, λ is set to 0.5. This setting is suitable for most datasets, and satisfactory results have been obtained in previous studies.

MMD is utilized for domain adaptation, which aims to map the source and target domain variables into a high-dimensional feature space, and to measure the distribution differences between the two domains in this feature space [11] (see Figure 2).

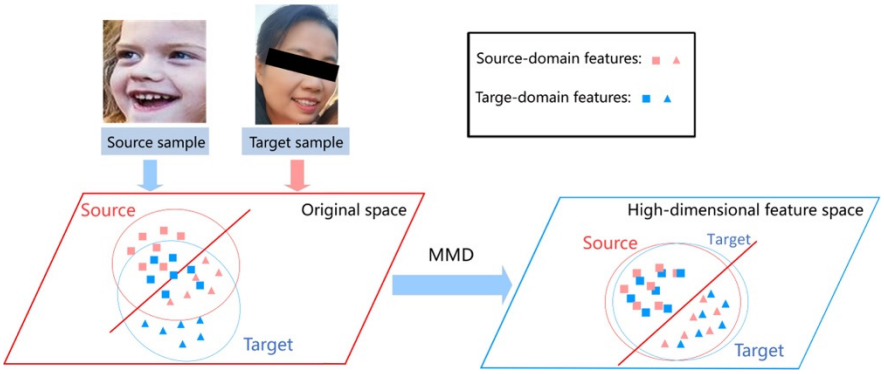


Figure 2. High-dimensional space for expression feature mapping.

It can be defined as follows:

$$\begin{aligned} \mathrm{MMD}[F, p, q] &:= \sup_{f \in F}\left(E_p[f(s)] - E_q[f(t)]\right) \\ &= \sup_{f \in F}\left(E_p[< f, \phi(s) >] - E_q[< f, \phi(t) >_H]\right) \\ &= \sup_{f \in F}\left(< f,\ \mu_p - \mu_q >_H\right) \\ &= \|\mu_p - \mu_q\|_H \end{aligned} \tag{2}$$

The source domain data s distribution, denoted as p, is mapped to the Reproducing Kernel Hilbert Space (RKHS) by f, and the expectation μ_p of $f(s)$ is computed. Similarly, the target domain data t distribution, denoted as q, is mapped to the RKHS by f, and the expectation μ_q of $f(t)$ is computed. The upper exact bound of the difference between μ_p and μ_q is calculated. However, there is a problem in calculating MMD directly. For multiple kernel functions, the MMD value cannot be estimated accurately by a limited number of samples [17]. To address this issue, the maximum mean discrepancy with the kernel Mean Embedding (MK-MMD) method is utilized, which expresses MMD as an inner product in the feature space generated by the kernel function [18]. This approach is often superior to MMD in practical applications [19]. The Gaussian kernel function is used with different parameters σ, which can be expressed as:

$$K(s, t) = \exp\left(\frac{-\|s - t\|^2}{2\sigma^2}\right) \tag{3}$$

To further elaborate on the method, we represent the mapping of the Gaussian kernel function by $f(\cdot)$ as $\Phi(\cdot)$. The corresponding MK-MMD of the regenerative Hilbert space [20] is defined as:

$$\mathrm{MMD}_{mk}[H, p, q] := \|E_p[\Phi(s)] - E_q[\Phi(t)]\|_H \tag{4}$$

The kernel matrix K can be expressed as:

$$K = \begin{bmatrix} K_{s,s} & K_{s,t} \\ K_{t,s} & K_{t,t} \end{bmatrix} \tag{5}$$

Assuming that the number of samples in the source domain is m and the number of samples in the target domain is n, the MK-MMD can be calculated by the following equation.

$$\begin{aligned} \mathrm{MMD}_{mk} &= \mathrm{tr}(KL) \\ L &= \begin{cases} \frac{1}{m^2}, s_i \in S \\ \frac{1}{n^2}, t_j \in T \\ -\frac{2}{mn}, \text{otherwise} \end{cases} \end{aligned} \tag{6}$$

Updating of the loss function is influenced by two factors: the learning rate and the training epoch [21]. The learning rate of the optimizer parameters is not fixed and needs to be adjusted at each step and epoch to prevent overfitting [22]. The initial value of the learning rate is set to 0.003. Unlike other deep learning methods, where the loss function depends only on feature extraction and final classification, domain adaptive learning involves calculating the difference between the features of the source domain and target domain [23]. Therefore, the learning rate of the domain adaptive network needs to be non-constant [24]. As the learning process continues, the loss function decreases, getting smaller and smaller. This gradual reduction helps to avoid overfitting and ensures that the model learns relevant information from the source and target domains [25].

3.2. Pre-Processing

Data preprocessing and enhancement techniques are usually applied before feeding the data into the network [26]. In the case of many pre-trained models, the input image is often resized to 224∗224, which is the same as the images in the ImageNet datasets [27]. During the intermediate operation steps, the image size does not change, and the size of the input image determines the size of the final fully connected layer. When scaled to 224∗224, the pre-trained model can be used directly without changing the input of the fully connected layer [28].

However, facial expression datasets require special treatment due to their unique characteristics. For example, adding random cropping operations to the images may inadvertently crop out important expression areas, which will have a negative impact on the accuracy of expression recognition. The horizontal flip operation, with the face, turned left or right, has little effect on the result but can increase the generalization of the model [29]. In addition, edge cropping processes can be performed to remove the spare areas of the image and highlight the expression region. The test set is used as a fixed dataset, in which the samples are not horizontally flipped but normalized to the same size.

During the experimental process of utilizing the pre-trained models, it is imperative to incorporate the weight parameters of the pre-existing model. Furthermore, to ensure that the data format used during training is consistent with the pre-training data, it is necessary to carry out data normalization. The purpose of this normalization is to promote the data to be "independent and identically distributed" and prevent the gradient from disappearance and explosion. As a result, the network training process can be more stable and rapid. Generally speaking, the data normalization process consists of centering and scaling. Centralization refers to moving the data to the center of the coordinate system to zero the average value of the data [30]. Scaling involves scaling data to ensure that they have similar scales. These can be expressed as:

$$\text{Centering}: \ x_i' = x_i - \mu \tag{7}$$

$$\text{Scaling}: \ x_i'' = \frac{x_i'}{\sigma} \tag{8}$$

where x_i is the original data, x_i' is the centralized data, x_i'' is the scaled data, μ is the mean of the data, and σ is the standard deviation.

Applying the normalization process to the entire dataset is crucial in practical applications, including training and testing sets [31]. This ensures that the data used in the training and testing process have similar scales, thus avoiding deviation caused by inconsistent data formats [32].

3.3. The Architecture of DA-FER

As depicted in Figure 3, the feature extraction part is the core of the transfer learning network, and any convolutional neural network can be utilized for this purpose. In this paper, a comparative analysis is conducted on seven different networks, and Densenet121 performs better results. Hence, it is selected as the backbone network of feature extraction. The

input image size is B∗3∗224∗224 (B∗C∗W∗H), where B is the batchsize, C is the number of channels, W is the width, and H is the height. To extract spatial features of the data through weight sharing and local connectivity, a Convolution + BatchNormal + ReLU + Maxpool module is initially applied to each local region of the input data. The convolution operation is followed by batch normalization, which normalizes the mean and variance of the output to prevent the gradient from disappearing and exploding and accelerate the training process simultaneously. The ReLU function, as a nonlinear activation function, promotes the nonlinearity of the convolution layer output and thereby enhancing the expressive force of the network. In addition, maxpool is employed to reduce the spatial dimension of the output and the number of network parameters. It also strengthens the translation invariance of the network.

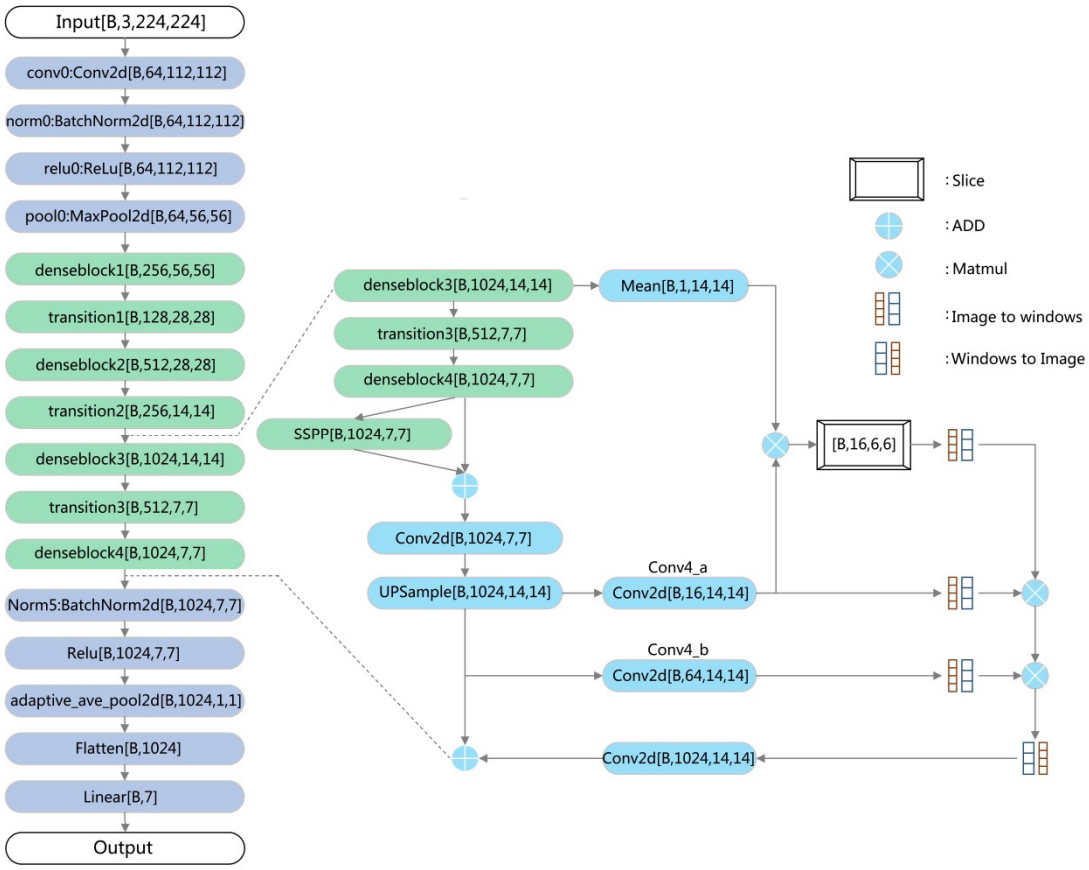


Figure 3. The architecture of the DA-FER network.

We mainly focus on the part of feature extraction (see Figure 2). First, the output of denseblock4 is fed through the Small Space Pyramid Pooling (SSPP) module [33] to obtain multi-dimensional feature fusion. Next, Conv4_a is multiplied with the result of denseblock3 after the mean operation. The result of the output is applied to the slice module to enhance the extraction of the five sensory features. Next, Conv4 is convolved once again to get Conv5_b. This step may seem optional, but it can improve the generalization ability of the model and strengthen the feature extraction. Next, the fusion result converted from Windows to image operation must be convolved. Generally, the convolution layer immediately follows the data operation. Finally, the input data is convolved by a sliding convolution kernel, which can better use the spatial information in the data for feature extraction and optimize the network performance [30].

Following the cyclic structure of dense blocks and transitions, batch normal for the feature extraction, ReLU activation function, adaptive average pool, tensors flatten, and fully connected layer are executed. This series of operations flattens the feature maps output

from the convolutional layer into a one-dimensional vector, maps the high-dimensional features into the target variables space, and generates the final prediction results.

3.3.1. Mean

We employ the mean module for denseblock3, which reduces the parameters of the whole network. The number of entered channels is 1024, averaged, and the rest is 1. We have also experimented with other methods here. For instance, we attempted to use the maximum value instead of the mean value but found that this approach was unstable in the training process, resulting in the training loss value of none. Additionally, we tried to adopt a convolution operation to convolve 1024 into a single dimension; however, this approach introduced extra parameters associated with another convolution group. The mean method is the most effective approach for reducing the parameter quantity while maintaining its overall performance [34].

3.3.2. Small Space Pyramid Pooling

The proposed Small Space Pyramid Pooling (SSPP) module on denseblock4, as illustrated in Figure 3, is a method to obtain multi-dimensional feature fusion on the same feature map by performing pooling of different scales. The denseblock4 layer is a deep layer with rich semantic information, so it is essential to perform feature aggregation from this layer in different dimensions to enhance the representation ability of the final feature map [35]. Since the output of denseblock4 is already very small, 7∗7, it is not meaningful to use too large pooling parameters at this time. Therefore, various methods, including convolutional kernels of different sizes, such as 3∗3, 5∗5, 13∗13, 15∗15, and 17∗17, have been tested individually or in combination. But most of them have produced unsatisfactory results and even caused gradient disappearance. Finally, the only retained layer for max-pooling was the layer with a 5∗5 convolution kernel. The result of max-pooling and its original result is concatenated and then processed by convolution and upsampling to obtain the new Conv4 output. It is worth noting that the SSPP module is relatively small and has several parameters, which is why it is called a small space pyramid pooling module.

3.3.3. Slice Module

The task of expression recognition involves identifying the region of interest (ROI) that contains the five senses in the face. The Slice module is a computational mechanism used to extract and preserve the most significant region of the face related to the task at hand. The specific operation is described by the following formula:

$$L = \frac{(L_o - \alpha)}{2} + L_o \tag{9}$$

The original size of the input image is denoted as L_o, and a scalar parameter α is used to control the size of the facial region of interest. Since the input to the Slice module is a 6∗6 feature map, the value of α is limited between 2 and 6 to ensure that the final retained feature map size L is appropriate for the task. Experimental results show that setting α to 4 is effective in retaining the most important facial features for expression recognition. This selection is performed both in width and height directions. To facilitate transfer learning, the preserved features are converted from 4-D to 3-D, which includes transforming the image into windows. This technique is usually used to convert the output of the convolutional layer in pre-training models into a 3-D shape tensor (batch_size, feature_size), which can be applied to subsequent convolutional layers for further processing and fine-tuning [36]. Specifically, the input tensor is segmented in a fixed size, and the values within each segmented region are concatenated along the spatial dimension to obtain a new tensor. This process is similar to the Transformer method that divides the input image into regional windows and stitches the features of each window [37].

This operation involves dividing the original image data into regional windows according to their spatial locations and merging the pixel values in each window into

a feature vector. The "windows to image" method is the inverse of the "windows to image" method, which extends the 3-D tensor into a 4-D tensor, which is equivalent to reopening the spatial features [38].

3.3.4. Adaptive Average Pooling

Transfer learning requires more parameters than the benchmark network [39]. Therefore, we perform adaptive averaging pooling at the last layer of feature extraction to reduce the size to 1∗1. This pooling operation reduces the number of parameters in the articulation layer by about half, decreasing the overall number of parameters in the network [40]. As a result, the total number of parameters of the proposed model is only 9.37 M, significantly reducing the computational pressure and improving the model training speed.

For a given input feature map X and pooling kernel size k × k, the averaging pooling operation can be expressed as:

$$Y_{i,j} = \frac{1}{k \times k} \sum_{x=0}^{k-1} \sum_{y=0}^{k-1} X_{i \times k+x, j \times k+y} \tag{10}$$

where $Y_{i,j}$ denotes the value of the (i, j) position of the pooled feature map. This operation divides the input feature map into k × k blocks and then calculates the average value within each block. The adaptive average pooling can be defined as:

$$Y_{i,j} = \frac{1}{\left|\frac{h}{k}\right| \times \left|\frac{w}{k}\right|} \sum_{x=0}^{|\frac{h}{k}|-1} \sum_{y=0}^{|\frac{w}{k}|-1} X_{i \times k+x, j \times k+y} \tag{11}$$

where h and w represent the height and width of the input feature map. The adaptive averaging pooling operation divides the input feature map into k × k blocks. However, the size and number of blocks are adaptive and depend on the size of the input feature map and the size of the pooling kernel. Adaptive averaging pooling is introduced to facilitate domain adaptive learning by reducing the offset between the source and target samples [41]. This technique has been proven to improve the robustness and generalization performance of the model because it dynamically adjusts the pooling size during training to effectively reduce the dimension of the feature map and the total number of parameters.

4. Experiment

4.1. Dataset

4.1.1. Selfie-Expression as Target Domain

To collect a dataset of expression recognition, we created a questionnaire link to invite willing participants to upload their selfie photos. A total of 97 participants contributed to the dataset. Almost all images were selfies, while a few participants submitted photos of multiple people. The original plan for the experiment was to collect datasets under laboratory conditions. Therefore, in describing each category of expressions, it was suggested that "we need to collect seven categories of facial expressions. You can upload 3–5 images per category—the more, the merrier. Please try to take only close-up, frontal shots of the head. Try to avoid tilting or covering your face. Without your permission, the images you provide will not be published anywhere". However, as many people do not like to read product manuals, most participants chose to take pictures from various angles, poses, and lighting conditions based on their preferences. The method of collecting samples by allowing users to provide their selfies has the advantage of being relatively convenient and fast. However, the disadvantage is that the participants may not take and upload pictures according to our intentions.

The final Selfie-Expression dataset consisted mainly of half-body images, and most participants uploaded images in only a few categories, with some uploading images in all seven categories. The dataset was further processed by removing 21 images with too much

occlusion (e.g., wearing a mask or too much hand occlusion), blurred faces, and images with facial deformation effects, which accounted for approximately 2% of the total sample size. Finally, the cropping operation was applied to only retain the head area, resulting in a final dataset of 866 images, with each category having a range of 102 to 162 images, as illustrated in Table 1. The link for collecting images was sent to interested participants by four Chinese college teachers. It took about a week to collect this small sample-size dataset. The collected samples were mostly provided by undergraduate students, resulting in a relatively small age range. In addition, our dataset only contains Chinese people, while Fer2013 and RAF-DB come from people worldwide with a wider age range. The differences in race and age range between the two source domain datasets and the target domain datasets also pose some challenges and difficulties for facial expression recognition.

Table 1. Description of the Selfie-Expression dataset.

Expression	Neutral	Happy	Surprise	Disgust	Fear	Sad	Anger	Total
Train	114	105	83	81	81	71	71	606
Test	48	45	36	35	34	31	31	260
Total	162	150	119	116	115	102	102	866

4.1.2. RAF-DB as Source Domain

For the RAF-DB [42] dataset, we use a total of 15,339 images of its 7-class base expressions, of which 12,271 images are applied for the training set, and 3068 images for the validation set, and the specific sample distribution is shown in Table 2. The sample expressions from each classification can be seen in Figure 4.

Table 2. Description of the RAF-DB dataset.

Expression	Happy	Neutral	Sad	Surprise	Disgust	Anger	Fear	Total
Train	4772	2524	1982	1290	717	705	281	12,271
Test	1185	680	478	329	160	162	74	3068
Total	5957	3204	2460	1619	877	867	355	15,339

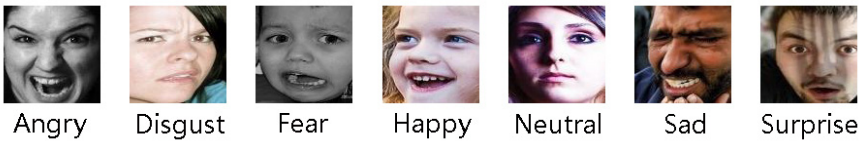


Figure 4. Sample images of each category from RAF-DB datasets.

4.1.3. Fer2013 as Source Domain

After preprocessing, the samples in the FER-2013 dataset were scaled to 48∗48 pixels. Datasets are usually divided into 28,709 for training, 3589 for validation, and 3589 for testing. See Table 3 for the distribution of specific samples. The sample expressions from each classification can be seen in Figure 5.

Table 3. Description of the Fer2013 dataset.

Expression	Happy	Neutral	Sad	Fear	Anger	Surprise	Disgust	Total
Train	7215	4965	4830	4097	3995	3171	436	28,709
Test	1774	1233	1247	1024	958	831	111	7178
Total	8989	6198	6077	5121	4953	4002	547	35,877

Figure 5. Sample images of each category from Fer2013 datasets.

4.2. Experimental Environment

The experiments were performed with Intel(R) Xeon(R) processor, 62 GB RAM, and NVIDIA GeForce GTX3080Ti GPU. The experimental parameters are shown in Table 4. The Lambda function adjusts the learning rate to help the network model converge better. The batchsize is fixed to 64.

Table 4. Parameters of the proposed method.

Parameters Name	Parameters
Learning rate	0.003
Learning gamma	0.0003
Learning decay	0.75
Weight for mmd	0.5
Seed	1024
Total epoch	100
Batchsize	64
Mini_batchsize	100/200
Optimizer	SGD
Learning scheduler	Lambda

Mini_batchsize is a unique parameter in transfer learning, which specifies the number of times to extract data from datasets in each round of network training. It is widely acknowledged that a larger mini_batchsize allows for more samples to be compared, resulting in richer feature extraction. However, this approach is very time-consuming. On the contrary, a smaller mini_batchsize takes up less memory and allows the network to converge faster. However, the accuracy of the experiment is sacrificed by iterating the source domain samples fewer times. It is worth noting that once the network has been learned to a certain extent, further learning may not be updated and optimized, resulting in no new knowledge. Therefore, selecting an appropriate mini_batchsize is critical to ensure the balance of efficiency and accuracy [43].

4.3. Results

The first and second columns of Table 5 show the results obtained by applying conventional methods and using seven baseline networks on the target domain for training and predicting. The "Baseline-none" indicates no pre-training strategy was used, while "Baseline-true" means using pre-trained models. Although some networks, such as AlexNet and VGG13 did not show obvious improvement, while others, such as VGG11 declined in accuracy, most networks exhibited an increase in recognition accuracy after pre-training.

A comparison of experimental results is conducted by utilizing various benchmark networks as the feature extraction component of the domain adaptive network. The third and fourth columns of Table 5 compare experimental results when using RAF-DB as the source domain and our self-collected dataset as the target domain, with mini_batchsize set to 100 and 200, respectively. On the other hand, the fifth and sixth columns of Table 5 display the experimental results obtained when using Fer2013 as the source domain and our self-collected dataset as the target domain. First, all domain adaptive networks were loaded with pre-trained models, which reduced the time cost of network training and effectively avoided overfitting. Our experimental results indicate that the domain adaptive method significantly improved the feature extraction ability of the network models due to the additional training of the source domain. Even models with poor performance, such as

AlexNet, VGG13, and VGG19, have achieved relatively good experimental results when utilized as the feature extraction component of transfer learning.

Table 5. Experimental results of source domain RAF-DB and source domain Fer2013.

Network	Baseline-None	Baseline-True	Source-RAF-DB Mini_100	Source-Fer2013 Mini_100	Source-RAF-DB Mini_200	Source-Fer2013 Mini_200
AlexNet	18.75	18.75	41.15	43.07	41.15	45.00
DenseNet121	**48.82**	**53.51**	52.69	48.46	52.69	49.23
Resnet-34	47.65	50.00	53.85	**51.92**	51.92	50.00
Resnet-50	35.54	39.45	49.62	49.62	49.23	50.38
VGG11	42.57	40.62	40	48.85	40.38	46.92
VGG13	18.75	18.75	40.77	46.54	40.77	46.54
VGG16	18.75	19.14	43.46	48.07	42.31	46.92
DA-FER	-	-	**58.46**	51.54	**56.15**	**50.77**

After comparing the performance of different benchmark networks, it can be seen that the performance of DenseNet121 and Resnet-34 is excellent, with an accuracy of about 50% (see Table 5, rows 2 and 3). To further investigate the comparative performances of these two models in the domain of adaptive learning, we plotted their receiver operating characteristic curve (ROC). We recorded the corresponding area under the curve (AUC), as shown in Table 6. The results show that selecting DenseNet121 as the domain adaptive network, RAF-DB as the source domain and Mini_batchsize of 100, is the best trade-off between network classification performance and training speed. Based on this, we designed the DA-FER network, achieving the highest accuracy of 58.46%, which is 5.77% higher than Transfer_Dense_100 (the number 100 indicates that the mini_batchsize of the transfer network is set to 100) and 4.95% higher than original DenseNet121 (see Table 5).

Table 6. Comparison of the area under the curve (AUC).

Network	Baseline-None	Baseline-True	Source-RAF-DB Mini_100	Source-Fer2013 Mini_100	Source-RAF-DB Mini_200	Source-Fer2013 Mini_200
DenseNet121	0.63	**0.66**	**0.77**	0.69	0.71	0.73
Resnet-34	**0.66**	0.64	0.76	0.73	0.77	0.70
DA-FER			0.76	**0.76**	**0.80**	**0.76**

Compared with RAF-DB, DA-FER doses do not perform with higher accuracy when FER2013 was used as the source domain (see Table 5). Nonetheless, it improved the area under the curve (AUC) compared to Transfer_Dense_100 and Transfer_Dense_200 (the number 200 indicates that the mini_ batchsize is set to 200). Furthermore, it attained closer accuracy than these models (see Table 6). This finding suggests that DA-FER produces fewer false positive and false negative cases and performs better classification. Therefore, it validates the effectiveness of the proposed domain adaptive method [44].

The experimental results of our proposed DA-FER with RAF-DB as the source domain dataset are shown in Figures 6 and 7. It can be seen that the network tends to converge at about 40 epochs, 60–80 epochs are stable, and finally, it reaches convergence at 80–100 epochs. As shown in Figures 8 and 9, from the perspective of AUC (area under the curve), the model classification ability of the DA-FER_100 is the same as that of the original Transfer_Dense network. However, compared with mini_100, the classification result of DA-FER_200 is higher, and the AUC of the model reaches the highest value of 0.8. Since the area under the curve is calculated based on all possible truncation values while considering the classification ability of the classifier for each sample, it is more robust, indicating the effectiveness of the DAF-FER. The accuracy of DA-FER_200 is 56.15%, which

is still the highest value compared to other networks in the same setting, which proves the effectiveness of this network.

As shown in Figure 10, confusion matrices of DenseNet121, ResNet34 and DA-FER networks with mini_batchsize set to 100 and 200, respectively, using RAF-DB as the source domain. From the given confusion matrix, happiness, surprise, and neutral categories are easy-to-recognize categories in our dataset. Furthermore, it can be observed from Table 1 that happy, neutral, and surprise are the categories with the largest number of samples. As a small sample size dataset, categories with relatively more samples have a better chance of achieving relatively higher accuracy. This is because more samples mean more training, which is more conducive to feature extraction and expression classification.

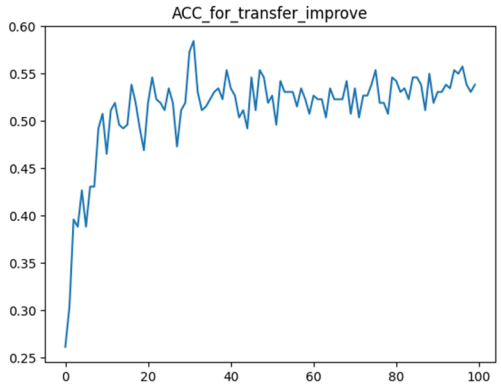


Figure 6. Accuracy of DA-FER_100.

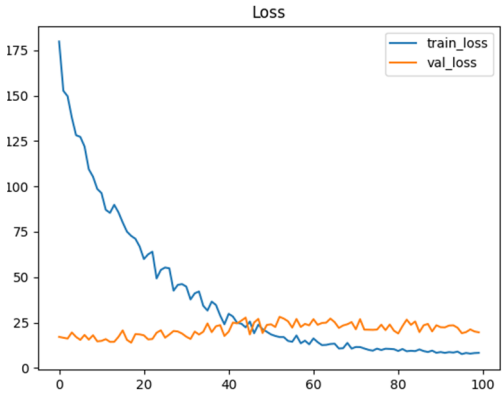


Figure 7. Loss of DA-FER_100.

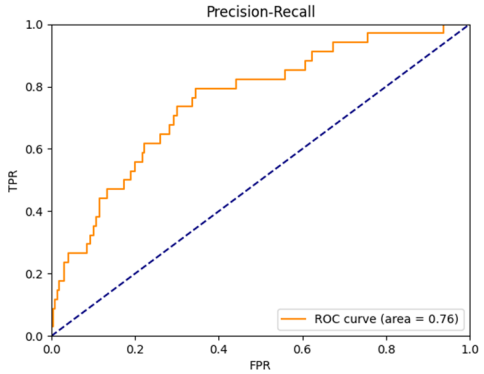


Figure 8. ROC curve of DA-FER_100.

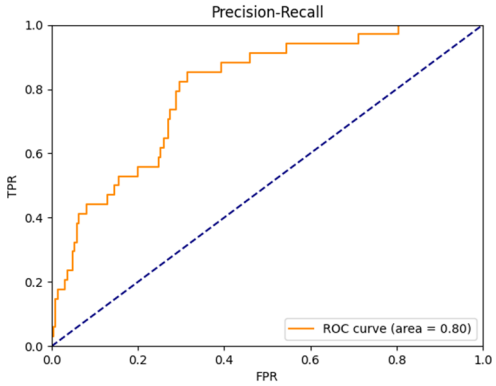
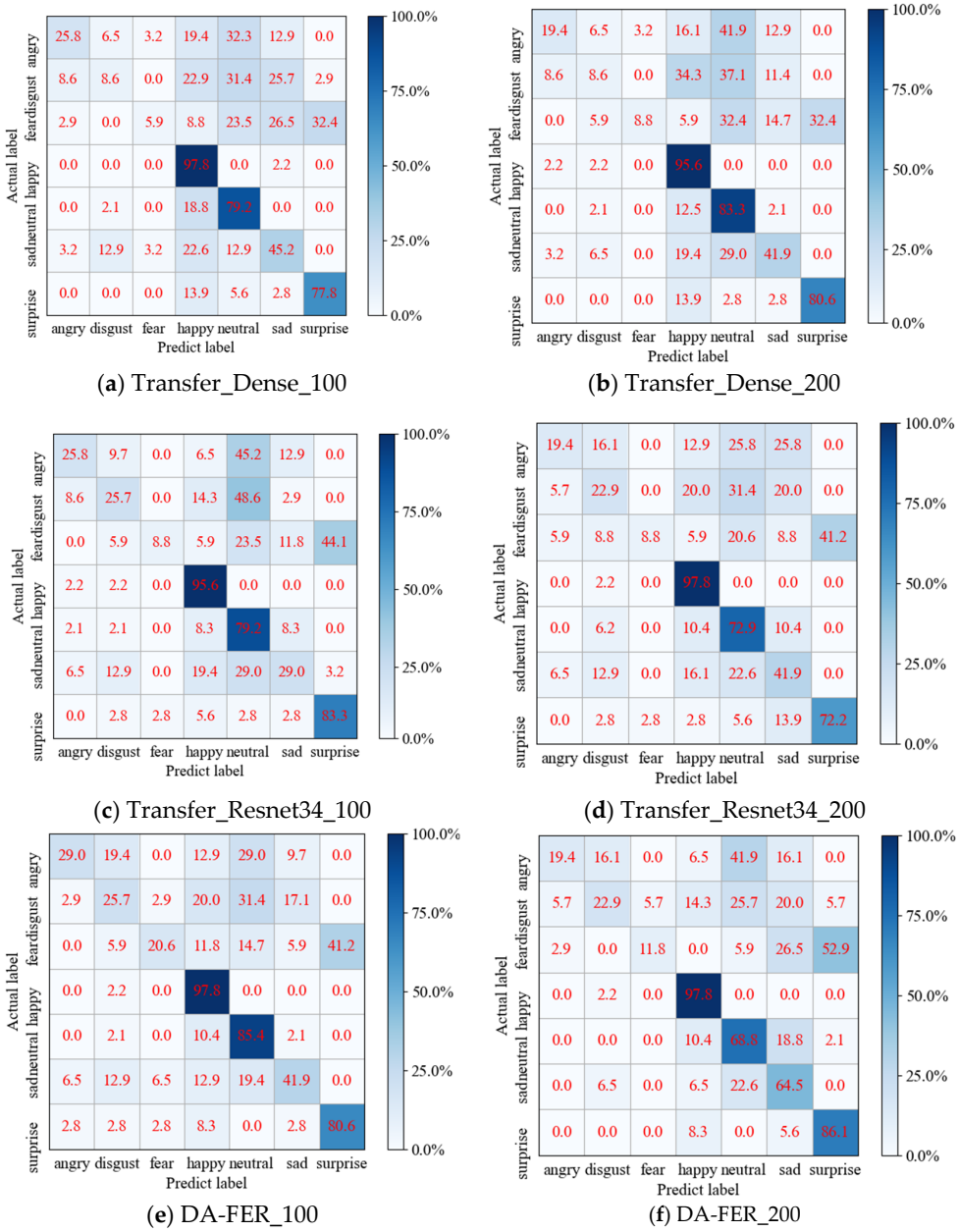


Figure 9. ROC curve of DA-FER_200.



(**a**) Transfer_Dense_100 (**b**) Transfer_Dense_200

(**c**) Transfer_Resnet34_100 (**d**) Transfer_Resnet34_200

(**e**) DA-FER_100 (**f**) DA-FER_200

Figure 10. The confusion matrix on DA-FER.

As can be seen from the experimental results shown in Figure 10a–d, the recognition results of anger, fear and disgust are generally low. In addition, from some misclassified samples, we observed that some faces exhibited excessive head rotation angles, leading to difficulties in recognition by the network. Another issue is the intra-class variability of expressions [45]. Some expressions have large amplitudes, making them easy to distinguish, while others have less prominent features, leading to confusion with other categories. Li et al. [46] also discussed the issue of confidence in expression samples, which explains why anger and sadness classification are considered easy in some studies [47].

In contrast, in our research, they fall into the more challenging category. Additionally, the complexity of these emotions and the limited availability of high-quality training data for these specific emotions may also lead to these challenges. The accuracy of DA-FER_100 in these three difficult categories has been significantly improved, compared with the original Transfer_DenseNet121, angry (+3.2%), disgust (+17.1%), fear (+14.7%) (see Figure 10a,d).

From Figure 10e,f, it can be observed that there is no change in the accuracy of the happy classification when comparing DA-FER_200 and DA-FER_100; the accuracy of the sad (+22.6%) and surprise (+5.5%) classifications has improved, while the accuracy of the angry (−9.6%), disgust (−2.8%), fear (−8.8%), and neutral (−16.6%) classifications have decreased. The overall average accuracy decreased by 2.31%, possibly due to over-fitting caused by the extra 100 rounds of learning.

As shown in Table 7, the total number of parameters in the original densenet121 is 19.8 M. The number of parameters in the transfer network tends to be much larger. Through the application of the Mean module and adaptive averaging pooling, we can effectively reduce the number of parameters by about 50%, resulting in a total number of parameters of 9.37 M. It is worth noting that although we have significantly reduced the number of parameters, the average inference time of each round is still basically the same as that of the original network. It demonstrates that the DA-FER network can balance accuracy and speed.

Table 7. Comparison of the number of parameters of each network.

Feature Extraction Network	Number of Trainable Parameters	Number of Parameters	Inference Time Mini-100
AlexNet	4,831,047	4.83 M	6.69 s
DenseNet121	19,800,967	19.8 M	5 s
Resnet-34	21,417,799	21.42 M	6.82 s
Resnet-50	24,034,375	24.03 M	5 s
VGG11	15,645,063	15.65 M	5 s
VGG13	15,829,575	15.83 M	7.47 s
VGG16	21,139,271	21.14 M	7.36 s
DA-FER	9,365,415	9.37 M	5s

5. Conclusions

Facial expression recognition (FER) is an essential field with many practical applications, but dealing with small sample data is a significant challenge. To improve the classification accuracy of FER under small sample data, we proposed a domain adaptive transfer learning method in this paper. Our approach involved using the public datasets RAF-DB and Fer2013 as the source domain, and a self-collected dataset as the target domain. We also applied the SSPP module for multi-dimensional feature fusion with variable perceptual fields and the slice module to retain regions of interest. Our optimized paradigm of transfer learning, mean module, and adaptive averaging pooling operation effectively reduced the number of model parameters, thereby achieving the best classification performance in terms of complexity and accuracy of the target domain. Experimental results showed that our proposed method outperformed the other seven popular networks. The domain adaptive method DA-FER, compared with the original DenseNet121 network,

improved the accuracy by 2.54% when mini_batchsize was set to 200 and reached the highest value of 58.46% when mini_batchsize was set to 100, which was 4.95% higher than the original network.

At present, we have not given enough consideration to the problem of network generalization, such as training on the source domain and predicting the target domain. In the future, we also intend to explore the effects of larger mini_batchsize settings (e.g., 300–600) on experimental outcomes and test more networks for feature extraction and additional datasets as source domain datasets. Through such efforts, we can enhance the generalizability of our experimental results and further expand our understanding of transfer learning and domain adaptive techniques.

Author Contributions: Conceptualization, M.B. and X.C.; methodology, H.X. and Q.L.; investigation, H.X. and Y.G.; writing—original draft preparation, M.B., K.S. and H.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was financially supported by the National Natural Science Foundation of China under Grant 62172184, Science and Technology Development Plan of Jilin Province of China under Grant 20200401077GX, 20200201292JC, Social Science Research of the Education Department of Jilin Province (JJKH20210901SK), Jilin Educational Scientific Research Leading Group (ZD21003) and Humanities and Social Science Foundation of Changchun Normal University (2020[011]).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, Y.; Zhang, H.; Shen, Q. Spectral–spatial classification of hyperspectral imagery with 3D convolutional neural network. *Remote Sens.* **2017**, *9*, 67. [CrossRef]
2. Wang, W.; Fu, Y.; Sun, Q.; Chen, T.; Cao, C.; Zheng, Z.; Xu, G.; Qiu, H.; Jiang, Y.G.; Xue, X. Learning to augment expressions for few-shot fine-grained facial expression recognition. *arXiv* **2020**, arXiv:2001.06144.
3. Shome, D.; Kar, T. FedAffect: Few-shot federated learning for facial expression recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Electr Network, Montreal, QC, Canada, 11–17 October 2021; pp. 4168–4175. [CrossRef]
4. Zhu, Q.; Mao, Q.; Jia, H.; Noi, O.E.N.; Tu, J. Convolutional relation network for facial expression recognition in the wild with few-shot learning. *Expert Syst. Appl.* **2022**, *189*, 116046. [CrossRef]
5. Niu, S.; Liu, Y.; Wang, J.; Song, H. A decade survey of transfer learning (2010–2020). *IEEE Trans. Artif. Intell.* **2020**, *1*, 151–166. [CrossRef]
6. Jin, C. Cross-database facial expression recognition based on hybrid improved unsupervised domain adaptation. *Multimed. Tools Appl.* **2023**, *82*, 1105–1129. [CrossRef]
7. Álvarez-Pato, V.M.; Sánchez, C.N.; Domínguez-Soberanes, J.; Méndoza-Pérez, D.E.; Velázquez, R. A multisensor data fusion approach for predicting consumer acceptance of food products. *Foods* **2020**, *9*, 774. [CrossRef] [PubMed]
8. Peng, X.; Gu, Y.; Zhang, P. Au-guided unsupervised domain-adaptive facial expression recognition. *Appl. Sci.* **2022**, *12*, 4366. [CrossRef]
9. Kong, Y.S.; Suresh, V.; Soh, J.; Ong, D.C. A systematic evaluation of domain adaptation in facial expression recognition. *arXiv* **2021**, arXiv:2106.15453.
10. Xie, Y.; Gao, Y.; Lin, J.; Chen, T. Learning Consistent Global-Local Representation for Cross-Domain Facial Expression Recognition. In Proceedings of the 2022 26th International Conference on Pattern Recognition (ICPR), Montreal, QC, Canada, 21–25 August 2022; pp. 2489–2495. [CrossRef]
11. Wu, Y.; Zhao, R.; Ma, H.; He, Q.; Du, S.; Wu, J. Adversarial domain adaptation convolutional neural network for intelligent recognition of bearing faults. *Measurement* **2022**, *195*, 111150. [CrossRef]
12. Wang, T.; Ding, Z.; Shao, W.; Tang, H.; Huang, K. Towards fair cross-domain adaptation via generative learning. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Electr Network, 5–9 January 2021; pp. 454–463. [CrossRef]

13. Kang, G.; Jiang, L.; Yang, Y.; Hauptmann, A.G. Contrastive adaptation network for unsupervised domain adaptation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4893–4902. [CrossRef]
14. Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A comprehensive survey on transfer learning. *Proc. IEEE* **2020**, *109*, 43–76. [CrossRef]
15. Shen, J.; Qu, Y.; Zhang, W.; Yu, Y. Wasserstein distance guided representation learning for domain adaptation. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32. [CrossRef]
16. Zhu, Y.; Zhuang, F.; Wang, J.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. Deep subdomain adaptation network for image classification. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 1713–1722. [CrossRef] [PubMed]
17. Yang, B.; Lei, Y.; Jia, F.; Li, N.; Du, Z. A polynomial kernel induced distance metric to improve deep transfer learning for fault diagnosis of machines. *IEEE Trans. Ind. Electron.* **2019**, *67*, 9747–9757. [CrossRef]
18. Zhuang, J.; Jia, M.; Ding, Y.; Ding, P. Temporal convolution-based transferable cross-domain adaptation approach for remaining useful life estimation under variable failure behaviors. *Reliab. Eng. Syst. Saf.* **2021**, *216*, 107946. [CrossRef]
19. Ding, R.; Li, X.; Nie, L.; Li, J.; Si, X.; Chu, D.; Liu, G.; Zhan, D. Empirical study and improvement on deep transfer learning for human activity recognition. *Sensors* **2018**, *19*, 57. [CrossRef] [PubMed]
20. Chen, Z.; Wang, C.; Wu, J.; Deng, C.; Wang, Y. Deep convolutional transfer learning-based structural damage detection with domain adaptation. *Appl. Intell.* **2022**, *53*, 5085–5099. [CrossRef]
21. Wu, X.; Ward, R.; Bottou, L. Wngrad: Learn the learning rate in gradient descent. *arXiv* **2018**, arXiv:1803.02865.
22. Takase, T.; Oyama, S.; Kurihara, M. Effective neural network training with adaptive learning rate based on training loss. *Neural Netw.* **2018**, *101*, 68–78. [CrossRef]
23. Venkateswara, H.; Eusebio, J.; Chakraborty, S.; Panchanathan, S. Deep hashing network for unsupervised domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5018–5027. [CrossRef]
24. Al-Khamees, H.A.A.A.; Al-A'araji, N.; Al-Shamery, E.S. Enhancing the stability of the deep neural network using a non-constant learning rate for data stream. *Int. J. Electr. Comput. Eng.* **2023**, *13*, 2123–2130. [CrossRef]
25. Long, M.; Cao, Z.; Wang, J.; Jordan, M.I. Conditional adversarial domain adaptation. *Adv. Neural Inf. Process. Syst.* **2018**, 31. [CrossRef]
26. Pérez-García, F.; Sparks, R.; Ourselin, S. TorchIO: A Python library for efficient loading, preprocessing, augmentation and patch-based sampling of medical images in deep learning. *Computer Methods Programs Biomed.* **2021**, *208*, 106236. [CrossRef]
27. Azizi, S.; Mustafa, B.; Ryan, F.; Beaver, Z.; Freyberg, J.; Deaton, J.; Loh, A.; Karthikesalingam, A.; Kornblith, S.; Chen, T.; et al. Big self-supervised models advance medical image classification. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Electr Network, Montreal, QC, Canada, 11–17 October 2021; pp. 3478–3488. [CrossRef]
28. Hu, F.; Xia, G.S.; Hu, J.; Zhang, L. Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery. *Remote Sens.* **2015**, *7*, 14680–14707. [CrossRef]
29. Aghamaleki, J.A.; Ashkani Chenarlogh, V. Multi-stream CNN for facial expression recognition in limited training data. *Multimed. Tools Appl.* **2019**, *78*, 22861–22882. [CrossRef]
30. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, UK, 2016. [CrossRef]
31. Tan, X.; Triggs, B. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Trans. Image Process.* **2010**, *19*, 1635–1650. [CrossRef] [PubMed]
32. James, G.; Witten, D.; Hastie, T.; Tibshirani, R. *An Introduction to Statistical Learning*; Springer: New York, NY, USA, 2013; Volume 112.
33. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [CrossRef]
34. Zheng, K.; Lan, C.; Zeng, W.; Zhang, Z.; Zha, Z.J. Exploiting sample uncertainty for domain adaptive person re-identification. In Proceedings of the AAAI Conference on Artificial Intelligence, Electr Network, 2–9 February 2021; Volume 35, pp. 3538–3546. [CrossRef]
35. Alahmadi, A.; Hussain, M.; Aboalsamh, H.A.; Zuair, M. PCAPooL: Unsupervised feature learning for face recognition using PCA, LBP, and pyramid pooling. *Pattern Anal. Appl.* **2020**, *23*, 673–682. [CrossRef]
36. Gu, Y.; Yan, H.; Zhang, X.; Liu, Z.; Ren, F.J. 3-d facial expression recognition via attention-based multichannel data fusion network. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 3125972. [CrossRef]
37. Bi, C.; Hu, N.; Zou, Y.; Zhang, S.; Xu, S.; Yu, H. Development of deep learning methodology for maize seed variety recognition based on improved swin transformer. *Agronomy* **2022**, *12*, 1843. [CrossRef]
38. Donahue, J.; Jia, Y.; Vinyals, O.; Hoffman, J.; Zhang, N.; Tzeng, E.; Darrell, T. DeCAF: A deep convolutional activation feature for generic visual recognition. In Proceedings of the International Conference on Machine Learning, (CYCLE1), Beijing, China, 22–24 June 2014; Volume 32, pp. 647–655. [CrossRef]
39. Shin, H.C.; Roth, H.R.; Gao, M.; Lu, L.; Xu, Z.; Nogues, I.; Yao, J.; Mollura, D.; Summers, R.M. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Trans. Med. Imaging* **2016**, *35*, 1285–1298. [CrossRef]

40. Ozcan, T.; Basturk, A. Static facial expression recognition using convolutional neural networks based on transfer learning and hyperparameter optimization. *Multimed. Tools Appl.* **2020**, *79*, 26587–26604. [CrossRef]
41. Tzeng, E.; Hoffman, J.; Darrell, T.; Saenko, K. Simultaneous deep transfer across domains and tasks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (ICCV), Santiago, Chile, 11–18 December 2015; pp. 4068–4076. [CrossRef]
42. Li, S.; Deng, W.; Du, J.P. Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2852–2861. [CrossRef]
43. Nguyen, D.; Sridharan, S.; Nguyen, D.T.; Denman, S.; Tran, S.N.; Zeng, R.; Fookes, C. Joint Deep Cross-Domain Transfer Learning for Emotion Recognition. *arXiv* **2020**, arXiv:2003.11136.
44. Chen, Y.; Joo, J. Understanding and mitigating annotation bias in facial expression recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Electr Network, Montreal, QC, Canada, 11–17 October 2021; pp. 14980–14991. [CrossRef]
45. Xue, F.; Wang, Q.; Guo, G. Transfer: Learning relation-aware facial expression representations with transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Electr Network, Montreal, QC, Canada, 11–17 October 2021; pp. 3601–3610. [CrossRef]
46. Li, H.; Wang, N.; Yang, X.; Wang, X.; Gao, X. Towards semi-supervised deep facial expression recognition with an adaptive confidence margin. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 4166–4175. [CrossRef]
47. Farzaneh, A.H.; Qi, X. Facial expression recognition in the wild via deep attentive center loss. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Electr Network, Waikoloa, HI, USA, 5–9 January 2021; pp. 2402–2411. [CrossRef]