Doctoral Dissertation

Estimation of Human Complex Emotions by Analysis of Facial Expression Images —Analysis-by-Synthesis of Mixed Facial Expressions Image by Emotion GANs (EmoGANs)—

WIN Shwe Sin Khine

Supervisor: Kazunori Kotani

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

September, 2023

Abstract

Facial expressions are the most interpretable visual signals used to perceive emotions in human social communication. Each emotion corresponds to distinct facial expressions. For instance, happiness is often conveyed through a smile, while surprise is expressed by the wide opening of the eyes and mouth. With advancements in technology, emotion recognition extends not only to humans but also to computers through the development of systems that can recognize emotions from signals like facial expressions.

Consequently, in literature, facial expressions are often accompanied by their corresponding emotion labels for analysis. However, the current state of the art in emotion research primarily focuses on basic emotions, namely anger, disgust, fear, happiness, sadness, and surprise. It is important to acknowledge that human emotions are complex and not solely limited to these basic emotions. Human emotions can be a mixture of multiple basic emotions. For example, graduating from a school can evoke a combination of happiness and sadness, often referred to as 'bittersweet' emotion. This illustrates that human emotions can be diverse and heterogeneous. However, the existing literature on computer vision has limited knowledge of such mixed emotions that are composed of multiple basic emotions. Besides, deep learning models are data demanding to generalize well and work efficiently.

Therefore, in the current study, we aim to estimate the emotions of mixed facial expression images using the Analysis-by-Synthesis (AbS) approach. By following the research objectives, we expect two research milestones: synthesized mixed facial expression images and mixed emotion labels. To accomplish the first milestone, we propose a generative model called Emotion Generative Adversarial Networks (EmoGANs) to synthesize mixed facial expression images that represent mixed emotions. The EmoGANs models operate by modeling the data from the existing prior distribution, also known as the latent space, and perform mapping of samples from the latent space onto the image space for image generation. Every sample from the latent space can be mapped onto the image space by the trained generator and create a new image. As a result, it provides a rich environment for image formation. Besides, mixed facial expressions and subject identity information can be controlled by the model parameters, which are not available in the existing research in the mixed emotion literature.

In the current study, we propose four generation models named EmoGANs, EmoGANs1, EmoGANs2, and EmoGANs3, based on the stability of their adversarial training, generated image quality, and spatial resolution. The generated images by all EmoGANs models are evaluated from various perspectives, including image quality, data diversity in image generation, involvement of mixed facial expressions, ability to control the image generation process, and disentanglement property.

For the second milestone, we approach the estimation of mixed emotions as a multi-label formulation. Each generated image contains two emotion labels out of the six basic classes, such as happiness and sadness in the example case. The estimation model is compared with the other state of art models. The result reported that our model obtained higher recognition performance than the compared models. Besides, we apply the model to estimate the emotions from the basic facial expression images as well. According to the result, the model also obtained high recognition performance on basic emotions. Moreover, we tested the model on unseen real images and it can select good features on unseen data.

Keywords: Mixed Emotions, Analysis-by-Synthesis, EmoGANs, Facial Expressions Analysis, Mixed Emotions Estimation.

Acknowledgements

I am deeply grateful to my advisors, Professor KOTANI Kazunori and Assistant Professor SIRITANAWAN Prarinya, for their invaluable guidance, encouragement, and expertise throughout the entire dissertation process. Their wisdom, encouragement, and insightful feedback have played a significant role in shaping and refining my research.

I would like to express my gratitude to Professor OGATA Kazuhiro, who is my second supervisor, for recruiting me to JAIST and offering me the opportunity to pursue a master's degree which has now set me a path to my academic journey towards a doctoral degree.

I would like to express my gratitude to Associate Professor OKADA Shogo for his guidance and expertise during the minor research project.

I would also like to express my appreciation for the time, comments, and guidance provided by my committee members from JAIST, namely Professor OGATA Kazuhiro, Professor IKEDA Kokolo and Professor HASEGAWA Shinobu. Their valuable comments and feedback have greatly enhanced the quality of this work.

Furthermore, I extend my gratitude to the external committee member from Tohoku University, Associate Professor ABE Toru, for his valuable insights and feedback, which has significantly enriched my research.

I am thankful to the Japan Advanced Institute of Science and Technology for providing research funding, enabling my participation in international conferences and academic journals. I would like to express my gratitude to JAIST and the IUCHI Asia Students Memorial Foundation for their support in granting me scholarships to pursue my degree.

I would like to express my heartfelt appreciation to my family in Myanmar for their emotional support, listening ear and positive mindset, which have been invaluable throughout my research journey. Their encouragement has played a crucial role in my progress and achievements.

I am also grateful for the support of our lab members, both from Myanmar and other international friends, as well as the members of the badminton club at JAIST. Their encouragement and understanding have been significant sources of motivation and strength for me.

Once again, I want to express my heartfelt thanks to all those who have supported me throughout this dissertation journey. Their contributions have been instrumental in the successful completion of this work.

WIN Shwe Sin Khine

Contents

Acknowledgements **1**

1 Introduction **1**
1.1 Introduction . . . 1
1.2 Overview of Emotions, Facial Expressions and Their Relationship . . . 2
1.3 Overview of Emotion Representations . . . 4
1.4 Research Problems and Motivation . . . 6
1.5 Research Purpose and Its Significance . . . 6
1.6 Dissertation Organization . . . 8

2 Literature Review **10**
2.1 Image Formation for Facial Expressions . . . 10
2.1.1 Conventional Methods . . . 10
2.1.2 Generative Adversarial Networks and Its Variants . . . 11
2.1.2.1 Variants of Generative Adversarial Networks . . . 12
2.2 Facial Expressions Recognition . . . 16
2.2.1 Conventional Methods . . . 16
2.2.2 Deep Learning Methods . . . 17

3 Data Preparation **19**
3.1 Dataset . . . 19
3.2 Pre-processing . . . 20
3.3 Peak Frame Selection . . . 22
3.4 Image Pair Selection . . . 29
3.5 Morphing . . . 30

4 Generating Complex Facial Expressions Images for Mixed Emotions **33**
4.1 Unsupervised Training . . . 34
4.1.1 Emotion Generative Adversarial Networks (EmoGANs) . . . 34
4.1.1.1 Exploring the advantages of convolution layers over densely connected layers . . . 35
4.1.1.2 Network configurations of Emotion Generative Adversarial Networks . . . 36
4.1.1.3 Network Training . . . 37
4.1.1.4 Discussion . . . 39
4.1.2 Wasserstein Distance Based Emotion Generative Adversarial Networks (EmoGANs1) . . . 41
4.1.2.1 Problems in GANs Training . . . 41
4.1.2.2 Wasserstein Distance or Earth Mover Distance . . . 43

4.1.2.3 Network Configurations of EmoGANs1 . . . 44
4.1.2.4 Network Training . . . 45
4.1.2.5 Discussion . . . 46
4.1.3 Methodology for Generating Mixture of Facial Expressions Images for Complex Emotions (EmoGANs2) . . . 47
4.1.3.1 Feature Extraction . . . 48
4.1.3.2 Image Generation . . . 51
4.1.3.3 Feature Conversion . . . 55
4.1.3.4 Discussion . . . 56
4.2 Supervised Training . . . 59
4.2.1 Conditional Emotion Generative Adversarial Networks with Identity Preservation (EmoGANs3) . . . 59
4.2.1.1 Encoding the label information for a mixture of emotions . . . 59
4.2.1.2 Encoding the identity information into latent space . . . 61
4.2.1.3 Conditioned Generative Adversarial Networks . . . 62
4.2.1.4 Evaluation Methods . . . 63
4.2.1.5 Discussion . . . 65

5 Evaluation on Synthesis Images **69**
5.1 Metrics for Generated Images Quality and Data Diversity . . . 69
5.1.1 Inception score . . . 69
5.1.2 Frechet Inception Distance (FID) . . . 71
5.1.3 Blind/Referenceless Image Spatial Quality Evaluator (BRISQE) . . . 72
5.2 Metric to evaluate mixed facial expressions in the generated images . . . 75
5.2.1 Cosine Distance Metric for Facial Expressions Features between generated images and morphed images . . . 75
5.3 Metric to evaluate feature disentanglement property . . . 76
5.3.1 Methodology to manipulate latent space . . . 76
5.3.1.1 Image Generation . . . 77
5.3.1.2 Emotion Classification . . . 79
5.3.1.3 Latent Sample Collection With Respect To Emotion Labels . . . 80
5.3.1.4 Facial Expression Manipulation . . . 81
5.3.1.5 Discussion . . . 82
5.3.2 Cosine similarity metric for subject identity information between an input image and transformed image . . . 83
5.3.3 Facial expressions recognition on images transformed in the latent space 87
5.4 Summary on synthesized images by all EmoGANs models . . . 89

6 Emotion Estimation **91**
6.1 Estimation of Basic Emotions through Basic Facial Expressions Features . . . 91
6.1.1 Extraction of Facial Expression Features . . . 92
6.1.1.1 Networks Training With Different Strategies . . . 93
6.1.2 Analysis of Facial Expression Features . . . 97
6.1.3 Discussion . . . 98
6.2 Estimation of Mixed Emotions through Synthesized Images by EmoGANs3 . . . 102
6.2.1 Network configurations of multi-labels facial expressions recognition model 103
6.2.2 Recognition Performance . . . 104

7 Conclusion and Future Work **109**
7.1 Chapters Summary 109
7.2 Conclusion 110
7.3 Future Work 111

Publications **113**

A Notation **123**

B Additional Information **125**
B.1 Image Generation by EmoGANs, EmoGANs1 and EmoGANs2 From Given Inputs For Each Mixture of Emotions 126
B.2 Full Evaluation Results of Conditional Emotion Generative Adversarial Networks (EmoGANs3) For Each Mixture of Emotions Class 140
B.3 Complete Experimental Results for Manipulation of Facial Expressions Attributes in Latent Space 155

List of Figures

1.1 Example of mixed emotions. Photo credit:(a)CW News, (b)Mainichi News . . . 2
1.2 The four components of emotions by Hamilton David R. [3] 3
1.3 Emotion representation by Prarinya Siritanawan [6] 4
1.4 Examples of dimensional emotion models 5
1.5 Example of complex emotions in 2D facial expressions feature space. Two dimensions (d_1 and d_2) are used for illustration purpose. 5
1.6 Difference between spontaneous and non-spontaneous facial expression for mixed emotion ("Happily Surprise"). (a) is the facial expressions presented in Compound Facial Expressions of Emotions Database [13]. (c) show the spontaneous facial expressions of actor Ben Affleck during his Oscar award, as captured by CW News. (e) shows the spontaneous facial expressions of Gold Medalist Risako Kawai during Rio Olympic, as captured by Mainichi News. (b), (d) and (f) show the confidence probability of emotions recognized in the respective figures using PyFeat: Python Facial Expressions Analysis Toolbox [16]. 7
1.7 An example diagram to show how a generative model G map from a sample from the latent Space Z onto image space X 8

2.1 Overview of facial expressions recognition models in literature 16

3.1 Number of peak images in each basic emotion in CK+, JAFFE and MUG dataset 19
3.2 Examples of six basic emotions from the CK+ (upper row), JAFFE (middle), and MUG (lower row) 20
3.3 Overview of face processing stages 21
3.4 Three types of Haar's kernel functions defined in Viola and Jones' algorithm[41], (a) and (b) are 2 rectangles Haar's features for vertical and horizontal edges, (c) is 3 rectangles feature for lines, and (d) is 4 rectangles features for complex structure. 21
3.5 Face processing results, (a) is an input image, (b) is a result of face localization which bounded the detected face region returned by Viola and Jones' algorithm[41], (c) is the result of face extraction for the bounded area. 22
3.6 Overview of peak frame selection framework. A represents recognition model parameterized by learned weights $\boldsymbol{\phi}$ to extract the facial expression features. $\mathbf{x}$ represents the image from the training set $\mathbb{X}$. $p(y)$ represents the confidence probability of being label y. 23
3.7 Visualization of first 9 feature maps from final convolution layers of each block in VGG face model 25
3.8 Visualization the clustering results, $\mathbf{c}_1$ refers to first cluster and $\mathbf{c}_1$ refers to second cluster. The rectangular box represents the key frame of the corresponding cluster. 27

3.9 Visualization the total feature response values with two key frames from each cluster. ★ refers to peak key frame. ● refers to neutral frame. 28
3.10 Overview of facial feature extraction by pre-trained model B parameterized by trained weights $\boldsymbol{\sigma}$[89]. $\mathbb{X}$ is the set of training images. j is the feature attribute where $j \leq 128$. 29
3.11 Overview of measuring the similarity among facial features. Pre-trained model B parameterized by $\boldsymbol{\sigma}$ takes training sample $\mathbf{x}$ and returns 128-dimensional facial features where $j \leq 128$. c_i represents the emotion class where $c_1 \neq c_2$. 30
3.12 Detected landmarks by Davis [90] with additional points 31
3.13 Output of morphing 32

4.1 Overview of the proposed model to estimate mixed emotion labels through analysis-by-synthesis approach 33
4.2 Overview of proposed image generation models 34
4.3 A simple example of a fully connected network with input and output layers, $\mathbf{x}^{(j)}$ is the input neuron and $\mathbf{y}^{j'}$ is the output neuron. $\mathbf{W}$ is the weighted matrix. j and j' are the feature indices for $\mathbf{x}$ and $\mathbf{y}$ respectively. 35
4.4 Matrix form of convolution operation in Convolution Neural Network with input and output layers 36
4.5 Overview of EmoGANs' generator 37
4.6 Overview of EmoGANs' discriminator 38
4.7 Example images generated by EmoGANs' generator using EmoGANs' objective function. Each row represents the generated samples from CK+, JAFFE, and MUG in sequential order. 41
4.8 An example case where Jensen Shannon (JS) divergence is not continuous. P_0 is the distribution such that $(0, \mathcal{N}(0,1))$ and P_θ is the prior distribution such that $(\theta, \mathcal{N}(0,1))$. θ is the parameter of the distribution. 43
4.9 Overview of EmoGANs1' generator 45
4.10 Overview of EmoGANs1' discriminator 45
4.11 Example images generated by EmoGANs1' generator using Wasserstein objective function. Each row represents the generated samples from CK+, JAFFE, and MUG in sequential order. 47
4.12 Overview of EmoGANs2 framework 48
4.13 An example of sparse interactions or connectivity in convolution neural network. The input image is resized into 8x8x1 and the edge filter is used as a convolution kernel for illustration purposes. 49
4.14 An example of translation equivariance in convolution neural network. 50
4.15 Network configurations of Convolution Neural Network (CNN) for basic emotion recognition 50
4.16 Plots for training a convolutional neural network (CNN) with Adam stochastic gradient descent optimization for every 16 batches. The learning rate is set to 1e-3. 52
4.17 Performance of convolutional neural network (confusion matrix over a test set) 53
4.18 An example of adding new layers onto the initial models G and D in Progressive Growing GANs [36]. (a) is the initial model for 4x4 resolution. (b) is the transition phase for adding the new layers to the initial models. (c) is the finalized model for 32x32 resolution. 54
4.19 Evaluation framework to validate the converted features 56

4.20 Distribution of cosine similarity scores measured among features from images synthesized by G on the initial latent sampled from $\mathcal{N}(0,1)$ and the latent estimated by the regression model. The scores range from +1 (the most similarity) to -1 (the most dissimilarity). 57
4.21 Samples of generated images by G on initial random latent and estimated latent by the regression model. 1^{st}, 3^{rd} and 5^{th} rows represent the images synthesized from the initial latent vectors sampled from the Gaussian distribution $\mathcal{N}(0,1)$. 2^{nd}, 4^{th}, and 6^{th} rows indicate the images synthesized from the latent estimated by the regression model. The first two rows are the result of CK+, the middle two rows are the result of JAFFE, and the rest are for MUG. 58
4.22 Example images generated by Progressie GANs' generator using the latent transformed from the average facial expressions features to represent the mixture of emotions. Each row represents the generated samples from CK+ and JAFFE in sequential order. 59
4.23 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'happiness'*** emotion, and c_j refers to the ***'surprise'*** class. Column (c) is images synthesized by the generator of the EmoGANs2 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). Results from other classes can be found in the Appendix section B.1. 60
4.24 Overview of EmoGANs3 framework. $\mathbf{y}$ is the label vector for the class of mixture of emotions, which is encoded in six-dimensional space. Each dimension chronologically refers to the class of anger, disgust, fear, happiness, sadness, and surprise. $\hat{\mathbf{z}}$ is the encoded feature vector for subject identity information. 60
4.25 Network configuration of encoder model to encode the facial representation into latent space 62
4.26 Network configuration of Conditioned Generative Adversarial Networks (cGANs) 63
4.27 Evaluation framework to assess the property of subject identity preservation . . 64
4.28 Evaluation framework to assess the property of disentanglement between facial expressions and facial representations 65
4.29 Prototypical action units (AUs) for basic emotions ('***happiness***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. Full results for the other mixed emotions can be found in the Appendix section B.2. 66
4.30 Example output of evaluation framework in Figure.4.27 for identity preservation. s is the cosine similarity score between the facial representation of the image from the initial latent (left) and reconstructed image from estimated latent $\hat{\mathbf{z}}$ (right) for each pair. The score ranges from +1 (most similarity) to -1 (most dissimilarity). 67
4.31 Example output of evaluation framework in Figure.4.28 for feature disentanglement. G is the pre-trained generator of DCGANs. $\mathbf{z}$ is the latent sample randomly drawn from prior Gaussian distribution. $\mathbf{y}$ is the random label that is two-hot encoded. $G(\mathbf{z}, \mathbf{y})$ indicates that the latent vector is mapped onto image space by the pre-trained generator. The upper row is from JAFFE. The lower row is from MUG. 68

5.1 Network configuration for multi classes mixed facial expressions recognition model 70
5.2 An random sample from each generator of GANs, DCGANs, FastGANs, and EmoGANs3 used in inception score metric for JAFFE dataset 71
5.3 An random sample from each generator of WGANs, and EmoGANs3 used in Frechet Inception Distance metric for JAFFE dataset 73
5.4 An random sample from each generator of (a)GANs, (b)DCGANs, (c)Morphing, (d)EmoGANs2 and (e)EmoGANs3 used in BRISQE metric for JAFFE dataset 75
5.5 Methodology to measure the distance between generated and training images in the feature space. I_g and $I_{\mathbf{x}}$ are the generated and training images. $\mathbf{v}$ refers to feature vectors extracted by the facial expressions recognition model from Section.5.1.1. d refers to cosine distance. c refers to the number of samples which has a close distance than the threshold value 0.5. 75
5.6 An random sample from each generator of EmoGANs, EmoGANs1, and EmoGANs3 used in measuring cosine distance in feature space. The upper row is for CK+, the middle is for JAFFE and the lower is for MUG. 77
5.7 Overview of facial expression manipulation in the latent space 77
5.8 Network configurations of image generation model (Deep Convolution Generative Adversarial Networks) 78
5.9 Network configurations of emotion recognition model (Convolution Neural Network(CNN)) 79
5.10 Confusion matrix of the emotion classification model for each emotion 80
5.11 System flow to collect latent samples with respect to emotion labels. $\mathbf{z}$ is the latent sampled from the prior distribution $\mathcal{N}(0,1)$. $\mathbf{y}$ is the desired emotion label. $\hat{\mathbf{y}}$ is the label predicted by the emotion classifier. $\mathbb{S}_{\mathbf{y}}$ is a set that includes latent samples with emotion label $\mathbf{y}$ and initially an empty set ϕ. 80
5.12 Diagram to manipulate the facial expressions attribute in the latent space. $\mathbb{S}_1$ belongs to the set containing the latent samples $\mathbf{z}$ with emotion label 1, for example, '*happiness*' emotion. $\mathbb{S}_2$ refers to the set for emotion label 2, for example, '*sadness*' emotion. $\hat{\mathbf{a}}$ refers to the decision boundary that separates the latent samples from two different classes. λ is the control parameter for direction. n is the number of latent samples. In this experiment, two thousand samples with respect to emotion labels are collected. d is the dimensions of the latent samples. Only 2 dimensions are used for illustration purpose. 81
5.13 An example output of facial expressions manipulation from *happiness* class to *sadness* class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. More experimental results for different classes can be found in Appendix section B.3. 83
5.14 Output of manipulating facial expressions attribute in the latent space using arithmetic property proposed in DCGANs by Radford et.al [31]. Columns (a) and (b) represent the generated images synthesized from the mean latent of anger and neutral emotions. Column (c) refers to generated images synthesized from the mean latent of the corresponding emotions such as (i) disgust, (ii) fear, (iii) happiness, (iv) sadness and (v) surprise. Column (d) is the output of arithmetic operation performed on those mean latent vectors. 84
5.15 Output of editing facial attributes that results to change facial expressions in the image such as (a) '*smiling*', (b) '*frowning*', (c) '*mouth slightly open*', (d) '*mouth wide open*' using deep feature interpolation proposed by Upchurch et.al [108]. . 85

5.16 Confusion matrix of the emotion classification model for each emotion after adding the samples created using the proposed methodology 85
5.17 Methodology to measure the similarity between an input image and output image transformed in the latent space. **I** is an input image and $\hat{I}$ is the manipulated or transformed image. **v** represents the feature vectors containing the subject identity information extracted by the VGG Face model [87]. s is a score given by the cosine similarity metric, which ranges from -1 for the most dissimilarity to +1 for the most similarity. 86
5.18 Output of evaluation methodology to measure facial similarity between a given input image (left) and transformed output image (right) in each pair. The output image at the upper row is transformed by the proposed methodology [105] and the ones at the lower row are generated from the predicted latent by the facial encoder of EmoGANs3. s is the cosine similarity score calculated from facial embedding vectors. The value ranges from +1 for the most similarity and -1 for the most dissimilarity. 86
5.19 Output of evaluation methodology to measure facial similarity between input images and transformed output image. Images from 1^{st} and 2^{nd} columns are random generated images from StyleGANs[37]. Images at 3^{rd} are the results of mixing style from input images. s is the cosine similarity score calculated from facial embedding vectors. The value ranges from +1 for the most similarity and -1 for the most dissimilarity. 87
5.20 Confidence probability score of synthesized images transformed in the latent space recognized by PyFeat Library[16]. 88
5.21 Confidence probability score of synthesized images transformed in the latent space recognized by PyFeat Library[16]. 88
5.22 Confidence probability score of synthesized images transformed in the latent space recognized by PyFeat Library[16]. 89
5.23 Example of generated images by each EmoGANs for CK+ (upper row), JAFFE (middle row), MUG(lower row). 90

6.1 Overview of emotion recognition model with transfer learning using various Training Strategies for performance comparison 92
6.2 An example output of data pre-processing phase 93
6.3 Visualization of projected feature points in the principal component space with the first two components (pca1, pca2). ● (A), ● (F), ● (D), ● (H), ● (S), ● (U) represent 'anger', 'fear', 'happiness', 'sadness', and 'surprise', respectively. . . . 99
6.4 Performance of emotion recognition models with transfer learning technique (confusion matrix over a test set) 100
6.5 Visualization of activated facial components while recognition the emotion by EfficientNet B0 model from setting 1-4. Upper row in each class represents average activation over all samples of the respective class. Lower row represents the output of a particular sample with activated faical components. 101
6.6 (a) An example image synthesized by the proposed EmoGANs3 model for the MUG dataset. The given mixed labels for example are 'happiness' and 'sadness' such as [0,0,0,1,1,0] in vector form. (b) and (c) are confidence probability scores predicted by Efficientnet B0 [112] and Py-feat[16]. 102
6.7 Network configurations to estimate multi-emotions labels of generated images (multi-labels CNN model) 103

6.8 Estimation of mixed emotions of an example generated image by multi-labels CNN model in MUG. 105
6.9 Estimation of mixed emotions of a real image by multi-labels CNN model. . . . 105
6.10 Confusion matrices evaluated over **basic facial expressions images** by multi-label CNN model and Efficient Net B0 [112]. 106
6.11 3D plot of Linear Discirminant Analysis (LDA) for features related to mixed facial expressions extracted by multi-labels CNN model. Each dimension represents each LDA component. Each color represents each mixed emotions class, which can be referred in Table.6.6. 108

B.1 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'anger'*** emotion, and c_j refers to the ***'disgust'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). Results from other classes can be found in the Appendix section. 126
B.2 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'anger'*** emotion, and c_j refers to the ***'fear'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 127
B.3 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'anger'*** emotion, and c_j refers to the ***'happiness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 128
B.4 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'anger'*** emotion, and c_j refers to the ***'sadness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 129
B.5 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'anger'*** emotion, and c_j refers to the ***'surprise'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 130
B.6 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'disgust'*** emotion, and c_j refers to the ***'fear'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 131

B.7 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'disgust'*** emotion, and c_j refers to the ***'happiness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 132
B.8 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'disgust'*** emotion, and c_j refers to the ***'sadness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 133
B.9 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'disgust'*** emotion, and c_j refers to the ***'surprise'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 134
B.10 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'fear'*** emotion, and c_j refers to the ***'happiness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 135
B.11 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'fear'*** emotion, and c_j refers to the ***'sadness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 136
B.12 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'fear'*** emotion, and c_j refers to the ***'surprise'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 137
B.13 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'happiness'*** emotion, and c_j refers to the ***'sadness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 138
B.14 Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'sadness'*** emotion, and c_j refers to the ***'surprise'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). 139

B.15 Prototypical action units (AUs) for basic emotions ('***anger***' and '***disgust***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 140
B.16 Prototypical action units (AUs) for basic emotions ('***anger***' and '***fear***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 141
B.17 Prototypical action units (AUs) for basic emotions ('***anger***' and '***happiness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 142
B.18 Prototypical action units (AUs) for basic emotions ('***anger***' and '***sadness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 143
B.19 Prototypical action units (AUs) for basic emotions ('***anger***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 144
B.20 Prototypical action units (AUs) for basic emotions ('***disgust***' and '***fear***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 145
B.21 Prototypical action units (AUs) for basic emotions ('***disgust***' and '***happiness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 146
B.22 Prototypical action units (AUs) for basic emotions ('***disgust***' and '***sadness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 147

B.23 Prototypical action units (AUs) for basic emotions ('***disgust***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 148
B.24 Prototypical action units (AUs) for basic emotions ('***fear***' and '***happiness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 149
B.25 Prototypical action units (AUs) for basic emotions ('***fear***' and '***sadness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 150
B.26 Prototypical action units (AUs) for basic emotions ('***fear***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 151
B.27 Prototypical action units (AUs) for basic emotions ('***happiness***' and '***sadness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 152
B.28 Prototypical action units (AUs) for basic emotions ('***happiness***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 153
B.29 Prototypical action units (AUs) for basic emotions ('***sadness***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. 154
B.30 An example output of facial expressions manipulation from '***anger***' class to '***disgust***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 155
B.31 An example output of facial expressions manipulation from '***anger***' class to '***fear***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 156

B.32 An example output of facial expressions manipulation from '***anger***' class to '***happiness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 157
B.33 An example output of facial expressions manipulation from '***anger***' class to '***sadness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 158
B.34 An example output of facial expressions manipulation from '***anger***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 159
B.35 An example output of facial expressions manipulation from '***disgust***' class to '***fear***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 160
B.36 An example output of facial expressions manipulation from '***disgust***' class to '***happiness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 161
B.37 An example output of facial expressions manipulation from '***disgust***' class to '***sadness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 162
B.38 An example output of facial expressions manipulation from '***disgust***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 163
B.39 An example output of facial expressions manipulation from '***fear***' class to '***happiness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 164
B.40 An example output of facial expressions manipulation from '***fear***' class to '***sadness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 165
B.41 An example output of facial expressions manipulation from '***fear***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 166
B.42 An example output of facial expressions manipulation from '***happiness***' class to '***sadness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 167
B.43 An example output of facial expressions manipulation from '***happiness***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 168

B.44 An example output of facial expressions manipulation from '***sadness***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. 169

List of Tables

1.1 Translation from facial expressions with groups of Action Units(AU) to emotions in Facial Action Coding Systems (FACS) [5] . . . 3

3.1 Accuracy of tuned VGG face model after last three dense layers (FC6, FC7, FC8) for facial expressions recognition task . . . 26

4.1 Accuracy of Convolution Neural Network (CNN) in 10 fold cross validation set for basic emotion recognition . . . 51

5.1 Accuracy of multi classes mixed facial expressions recognition model in 10 fold cross validation set . . . 70
5.2 Results for Inception Score metric . . . 71
5.3 Results for Frechet Inception Distance metric . . . 72
5.4 Results for Blind/Refereenceless Image Spatial Quality Evaluator metric . . . 74
5.5 Results for measuring the cosine distance between facial expressions from generated and training images. . . . 76
5.6 Summary of EmoGANs models proposed in this dissertation . . . 90

6.1 Training configurations for transfer learning and new training for emotion recognition . . . 94
6.2 Average accuracy of each model evaluated over 5-fold cross-validation sets for transfer learning. Accuracy is shown in percentage (±standard deviation). Maximum score is highlighted in bold font. . . . 95
6.3 Average accuracy of each model evaluated over 5-fold cross-validation sets for new training. Accuracy is shown in percentage (± standard deviation). Maximum score is highlighted in bold font. . . . 95
6.4 Average accuracy of each model tested across different data segmentation for 5 trials in each settings. Accuracy is shown in percentage (± standard deviation). Maximum score is highlighted in bold font. . . . 96
6.5 Evaluation for multiple labels estimation model over test set. f_β is the f-beta metric and EMR is the exact match ratio that calculates the ratio of the predicted labels as identical as true labels . . . 104
6.6 Name code for mixed emotions classes . . . 107

A.1 Notation of symbols, variables and functions with their descriptions used in the dissertation. . . . 123

B.1 Relationship between Action Units (AUs) and Facial Muscle in Facial Action Coding Systems (FACS) [5] . . . 125

Chapter 1

Introduction

1.1 Introduction

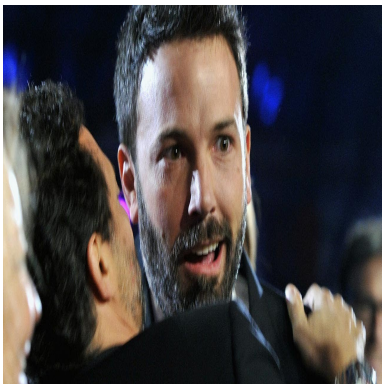
Emotions are intangible psychological states and play an important role in social interaction. Although emotions are intangible, humans are built upon practicing reading social cues from their surroundings. As a result, they can perceive emotions from each other by reading social cues or non-verbal signals. With advanced technology in computer vision and machine learning, emotions can now be perceived not only by human beings but also by machines or computers. Therefore, it is possible to build an automatic system that can observe and interpret emotions based on social cues or signals, thus delivering emotion perception in machines.

The importance of social signals in emotion recognition has been researched in the literature. These signals can come in different forms or data modalities such as voice tone, choice of spoken words, facial expressions, body language, and others. Mehrabian's study [1] analyzed the contributions of three communication signals used in social interaction for recognizing emotions. It was reported that 55% of the signals come from facial expressions, 38% from speech, and 7% from the choice of spoken words. This report indicates that facial expressions majorly contribute to human emotion perception and suggests that faces are the first engaged area in human-to-human communication. Therefore, human beings are aware of and used to recognizing signals from faces, specifically facial expressions. Facial expressions are thus useful social signals for building emotion recognition systems.

Facial expressions can have multiple meanings and can be used in different contexts. For example, although a smiling face typically exhibits happiness in emotion recognition, it can also be a sign of courtesy or politeness. It is also important to note that facial expressions can be formed without the presence of emotion or can be presented as the opposite expressions against the emotions. For example, a smiling face can be used to cope with or cover up the emotion of sadness, which is often employed in social communication. To clarify the standpoint of this dissertation, the meaning of facial expressions used in the study refers to visible signals evolved from emotions, which are used to study and recognize emotions.

State-of-the-art research on facial expression recognition for emotion estimation focuses on Ekman's six basic emotions: anger, disgust, fear, happiness, sadness, and surprise. However, there are situations where the definition of basic emotions cannot cover certain experiences. For example, when we encounter unexpected joyful news, events, or outcomes, we may feel not only happiness but also surprise (see Figure 1.1). These types of emotions are called mixed emotions. Little is known about mixed emotions in emotion research. While emotion recognition through facial expressions has been extensively studied from both psychology and affective computing perspectives in the past decades, most recognition systems have failed to estimate or recognize

mixed emotions. In this current study, we aim to develop a system that can estimate and target mixed emotions from generated facial expression images, rather than simple emotions. This will be explained in detail in the later section of this chapter.



(a) Ben Affleck (American Actor) during Oscar award

(b) Risako Kawai (Olympic Gold Medalist) during Rio Olympic

Figure 1.1: Example of mixed emotions. Photo credit:(a)CW News, (b)Mainichi News

1.2 Overview of Emotions, Facial Expressions and Their Relationship

Emotions play a significant role in social communication. When human beings communicate with each other, not only understanding the conversational contents but also recognizing the other party's emotions is essential. Understanding human emotions helps better communication since emotions influence the method of communication. On some occasions, emotion underneath the conversation is more important than what is being said. Naturally, human beings are raised and grown up in a social environment, they can sense and be aware of their own emotions as well as the emotions of other human beings by using social signals such as facial expressions, vocal tones, body language, speech, and many others.

Numerous psychologists have debated and discussed what human emotions are for centuries. Greek psychologist Aristotle defined that to have emotions, we have to experience pain, pleasure, or both. American Psychological Association (APA) describes emotions as a complex reaction pattern, including experiential, behavioral, and psychological elements. In the Cannon-Bard Theory of emotions, emotions are mental states and significant since emotions have been considered a physical response. In 1981, Kleinginna [2] listed the emotion definitions from various perspectives. Emotion definitions vary depending on the people and their standpoints. In this dissertation, emotions are the state of feeling or intangible psychological states towards an object, substance, event, or experience. To experience an emotion, a particular stimulus had to happen to trigger the emotions.

With advanced technology, emotion recognition is not limited to human beings but also in machines and robots from nonverbal signals. According to Figure 1.2, Hamilton's work explained the occurrences of nonverbal signals when emotion is triggered [3]. When a stimulus such as an event or experience happens, it stimulates a particular emotion. If the emotion is strong enough, it changes the hormone level inside the brain. These changes impact the Autonomous Nervous System, a part of the peripheral nervous system. The system connects

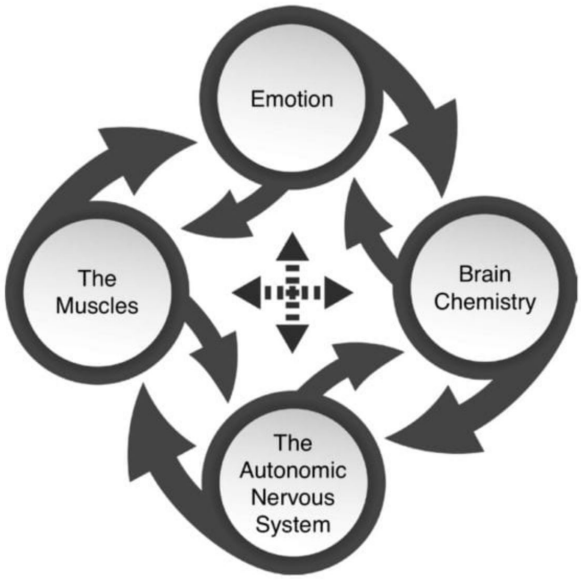


Figure 1.2: The four components of emotions by Hamilton David R. [3]

the brain and the internal organs through the blood vessels. As its name suggests, it involuntarily regulates the physiological process. Therefore, when the emotion happens, it is spread throughout the whole body by the autonomous nervous system. As a result, nonverbal signals such as changes in blood pressure, heartbeat, and facial expressions as responses to emotions have occurred.

Some debate that the occurrence of facial expressions is conscious to cover the true intention or used as a social response. For example, when we encounter our friends or colleagues, we greet them with a light smile, which is an intentional facial expression unrelated to emotions. However, Matsumoto et al. [4] showed that naturally blind athletes produced the same facial expressions as others in the context of winning or losing the games, although they did not know what the facial expression resembled. Matsumoto's work supported that facial expressions spontaneously happened as a response to emotions.

Table 1.1: Translation from facial expressions with groups of Action Units(AU) to emotions in Facial Action Coding Systems (FACS) [5]

Emotions	Group of Action Units (AU)	FACS Name
Anger	4 + 7 + 11 + 24	Brow Lowerer; Lip Tightener; Nasolabial Deepener; Lip Pressor;
Disgust	9 + 10 + 24	Nose Wrinkler; Upper Lip Raiser; Lip Pressor;
Fear	1 + 2 + 5 + 20 + 25	Inner Brow Raiser; Outer Brow Raiser; Upper Lip Raiser; Lip Stretcher; Lips Part;
Happiness	6 + 12 + 25	Cheek Raiser; Lip Corner Puller; Lips Part;
Sadness	1 + 4 + 6 + 11 + 15 + 17	Inner Brow Raiser; Brow Lowerer; Cheek Raiser; Nasolabial Deepener; Lip Corner Depressor; Chin Raiser;
Surprise	1 + 2 + 5 + 26	Inner Brow Raiser; Outer Brow Raiser; Upper Lip Raiser; Jaw Drop;

In addition, psychologist Ekman established the Facial Action Coding System (FACS) [5], which includes the movement of facial muscles due to the appearance of facial expressions in

correspondence with emotions. They are listed in Table B.1. The connection between emotions and activated action units is tabulated in Table 1.1. From the literature, it can be observed that facial expressions are useful signals and have a well-established relationship in defining human emotions.

1.3 Overview of Emotion Representations

Emotion representations can be mainly divided into two categories; discrete model and dimensional model. They are illustrated in Figure. 1.3 by Siritanawan's work [6].

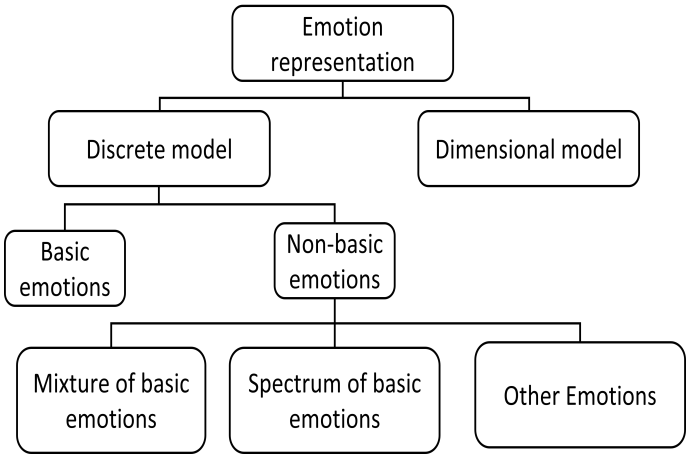
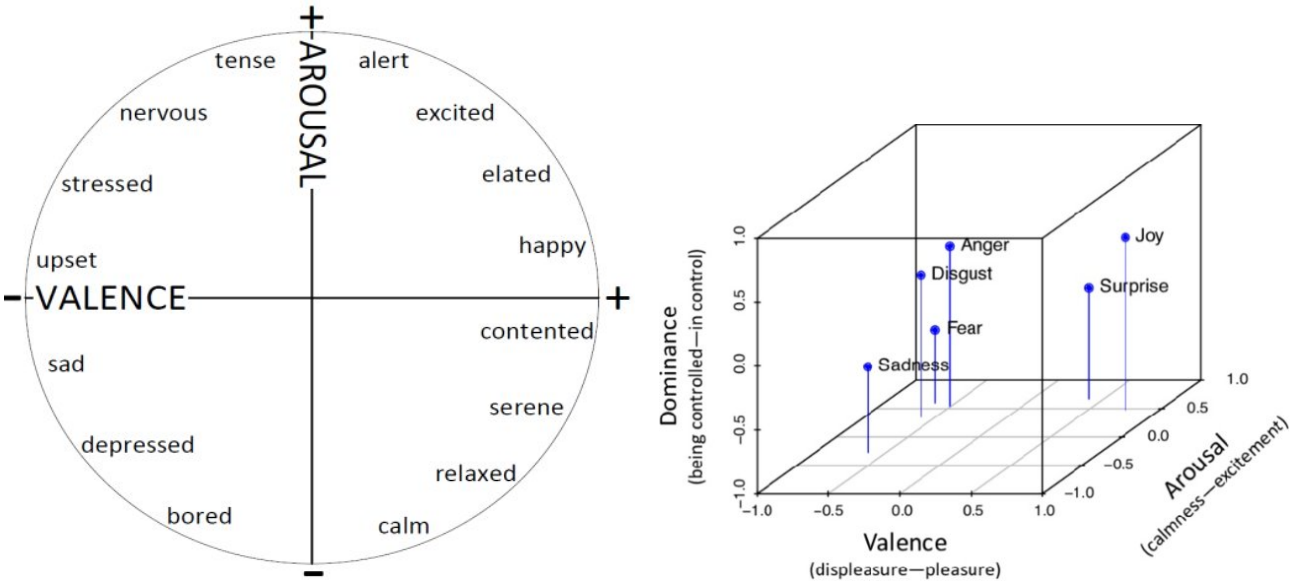


Figure 1.3: Emotion representation by Prarinya Siritanawan [6]

The dimension model defines the emotion representations based on the dimension parameters. For example, the famous work by Russell [7] described emotions based on the valence and arousal elements on the Cartesian coordinates. In Russell's model, happiness is located at a high level of both arousal and valence, whereas sadness is spotted at a low level of both arousal and valence. According to the circumplex model, happiness and sadness represent the two polar emotions and cannot co-exist. Extended from Russell's circumplex model, Mehrabian added dominance as a third dimension representing the level of control to describe emotions. Russell and Mehrabian dimensional models are shown in Figure. 1.4. Many researchers have discussed the number of dimension parameters. The four-dimensional model (Pleasure, Tension, Impulse, Confidence) [8] and five-dimensional model (Intensity, Pleasure, Control, Certainty, Tension) [8] can also be seen in the literature.

In the discrete model, the emotion representations are categorized into discrete emotion classes depending on their standpoint of emotion definitions. Psychologist Ekman categorized emotion representations into six classes: anger, disgust, fear, happiness, sadness, and surprise. Ekman stated that those emotions are primary or basic and cannot be decomposed to form another. Jack et al. [9] proposed that basic emotions are anger, fear, happiness, and sadness. In Plutchik's work [10], they described eight basic emotions: anger, fear, sadness, disgust, surprise, anticipation, trust, and joy.

Although basic emotions cannot be broken down to be another emotion, they can be mixed to be new, as in Figure. 1.3. The mixed emotions mean the emotions co-existed at the same time. Numerous psychologists debated this statement of mixed emotions. According to Russell's circumplex model, happiness and sadness cannot happen together since they are on two opposite sides of the valence dimension. Some psychologists stated that since happiness and



(a) Circumplex Model with Two Dimensions (Valence and Arousal) by Russell [7]

(b) Pleasure, Arousal, Dominance (PAD) Model by Mehrabian [11]

Figure 1.4: Examples of dimensional emotion models

sadness describe two extreme meanings of pleasure and displeasure, mixing these two emotions resulted in the neural emotion, as a synonymous scenario in mixing acid and base results in a neutral solution. Larsen et al. [12] discussed the mixture of happiness and sadness in their work by considering the bittersweet events that induced both emotions. Their work concluded that the mixture of those two emotions could happen simultaneously and opened the future direction of other mixed emotions beyond happiness and sadness.

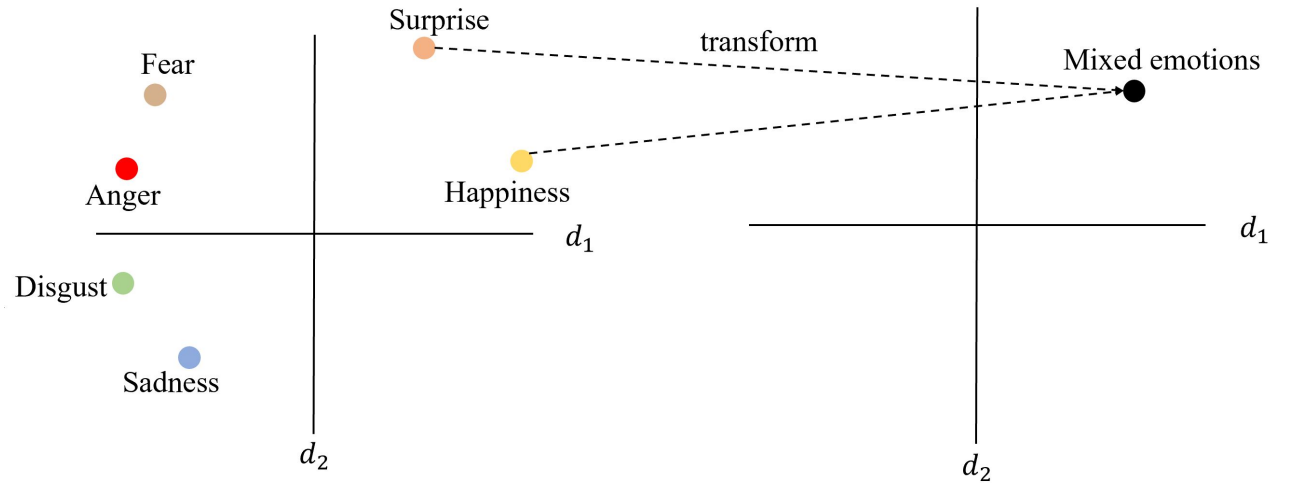


Figure 1.5: Example of complex emotions in 2D facial expressions feature space. Two dimensions (d_1 and d_2) are used for illustration purpose.

As previously stated, facial expressions can be utilized as signals to measure basic emotions. In the context of mixed emotions, Can facial expressions be represented to realize the mixed emotions? The series of works by Du et al. [13, 14] addressed this question. Their work showed the facial expressions of mixed emotions comprised of the underlying facial expressions for each

emotion in the mixture. Figure.1.5 illustrated the meaning of the mixed emotions targeted in this dissertation. Suppose that there exists a feature space that can represent the distinct facial expressions for each emotion. The mixed representations of two different emotions are non-linearly transformed in the feature space to represent the mixed emotions in this dissertation. Fifteen possible combinations of Ekman's six basic emotions will be addressed.

1.4 Research Problems and Motivation

At present, the recognition of human emotions is mostly restricted to simple and basic emotions. There are three motives behind it. The most significant advantage of Ekman's basic emotions is being simple and categorically different. Since each facial expression is distinctly different, the emotion representations can be easily classified into emotion classes. The second advantage is that facial expressions of basic emotions are universal. It was proved in the work [15] by studying the facial expressions of two cultures: Japanese and American. Since basic facial expressions are universal, every language has specific emotional words to describe basic emotions. It indicates that everyone can easily understand basic emotions in their own language.

As discussed earlier in this chapter, human emotions are complex. Although basic emotions are commonly applicable in realizing human emotions most of the time, it is not deniable that non-basic emotions can also happen in a particular situation (See Figure.1.1). In Du's work [14], the particular situations that triggered the mixture of basic emotions (compound emotions) are listed. For example, happily sadness mixed emotion occurs when we encounter events that bring pleasure and displeasure information to us. In other words, it is described as the new mixed emotion named bittersweet in [12, 14].

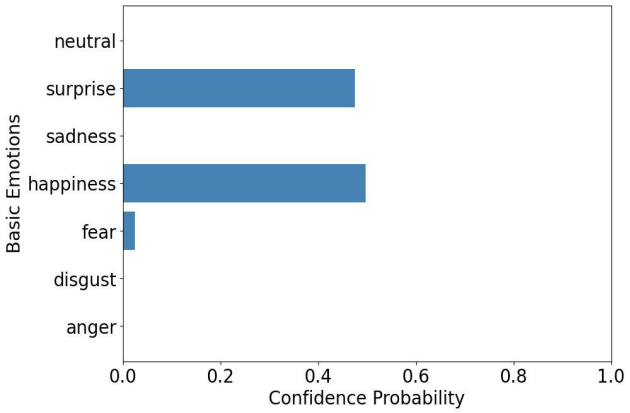
In previous work on mixed emotions by Du et al. [13, 14], the consistency among subordinate facial expressions for each basic emotion was analyzed. The facial expressions used in the mixed emotion literature are shown in Figure 1.6. It can be observed that there is a gap between voluntary and involuntary facial expressions. The involuntary facial expressions are blatant and intense, which differ from what we experience in real-life situations. Although the current state-of-the-art recognition system (PyFeat [16] used in the example Figure 1.6) correctly recognizes the expressed emotions in the Compound Facial Expressions of Emotions Database [13], it fails to recognize the emotions in actual events, for example, Figure 1.6(c), where although surprise emotion can be perceived, it is recognized as happiness. Additionally, the previous work defines mixed facial expressions as the linear combination of Action Units (AUs). However, mixed facial expressions might not be a linear process. This also highlights the challenging factor of the current state-of-the-art recognition system for mixed emotions.

1.5 Research Purpose and Its Significance

This research aims to create complex facial expression images in order to estimate mixed emotions using synthesized images. The analysis-by-synthesis (AbS) approach, initially proposed by Morris et al. [17] for speech recognition, is adopted to achieve the research objectives. The AbS approach is designed based on the cognitive processes of human beings. Humans can both speak (action) and listen (perception), as well as see (perception) and recognize what they see (action) in general.

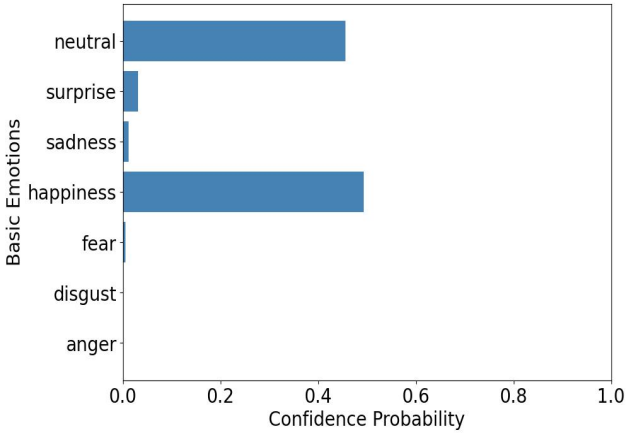
Why is the AbS approach better than the traditional way for image formation? To address this question, suppose a generation model exists that maps samples from the prior

(a) Non-spontaneous facial expression



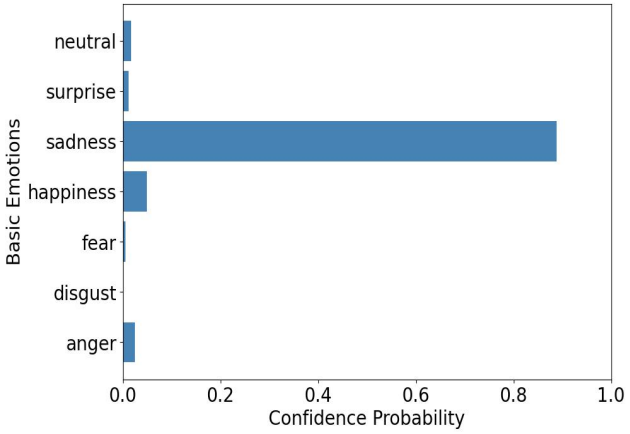
(b) Emotions in Figure.(a)

(c) Spontaneous facial expression



(d) Emotions in Figure.(c)

(e) Spontaneous facial expression



(f) Emotions in Figure.(e)

Figure 1.6: Difference between spontaneous and non-spontaneous facial expression for mixed emotion ("Happily Surprise"). (a) is the facial expressions presented in Compound Facial Expressions of Emotions Database [13]. (c) show the spontaneous facial expressions of actor Ben Affleck during his Oscar award, as captured by CW News. (e) shows the spontaneous facial expressions of Gold Medalist Risako Kawai during Rio Olympic, as captured by Mainichi News. (b), (d) and (f) show the confidence probability of emotions recognized in the respective figures using PyFeat: Python Facial Expressions Analysis Toolbox [16].

distribution onto image space, as shown in Figure 1.7. For instance, if we have a blue sample

in the latent space, it can be mapped onto image space by the model G. Now, if we modify the values of this sample and it becomes the gray sample in the latent space, the model G will map the new sample to a different image. This process introduces numerous variations in the image space, creating a rich environment for image formation that is not possible with the traditional approach. Therefore, the AbS approach is adopted in this dissertation. .

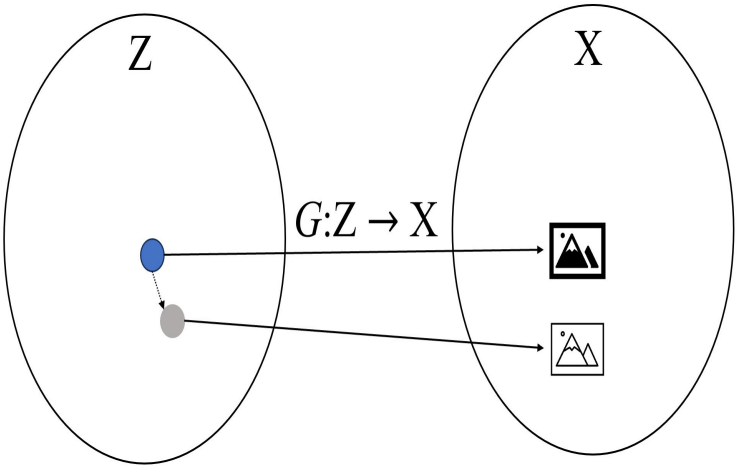


Figure 1.7: An example diagram to show how a generative model G map from a sample from the latent Space Z onto image space X

By realizing the research objective, the following milestones are expected.

1. Generate the complex facial expressions images for mixed emotions.
2. Estimate mixed emotions in feature space using generated images.

Compared to literature in mixed emotions research, the characteristics or significance of the current research can be defined as follows.

1. Strength and type of facial expressions in the image can be controlled by the parameters of the generation model.
2. Although the current research focuses on a mixture of two emotions at the moment, it is flexible and can easily be adapted to mix more than two facial expressions.
3. Mixed representations among emotions are done by non-linear transformation.
4. Generation provides a rich environment for image formation.

1.6 Dissertation Organization

This dissertation is organized as follows:

- **Chapter 1** introduces the psychological background of emotions, facial expressions, and their connection. It also includes the definition of the mixture of emotions for this current research. In addition, it includes the research's motivation, purpose, and goal.
- **Chapter 2** introduces the previous work on image generation methodology and models, not only limited to the traditional approach but also including deep learning concepts.

- **Chapter 3** introduces the essential processes for preparing the training data used in the experiments.
- **Chapter 4** contains the overview of the proposed methodology used to achieve the research objectives. It discusses proposed generative models and their evolution.
- **Chapter 5** evaluates the generated images by the proposed generative models from the perspective of image quality, data diversity, and feature disentangle property.
- **Chapter 6** discusses the emotion estimation models based on basic facial expressions images and generated mixed facial expressions images, including the analysis of facial expressions based on their discriminant properties.
- **Chapter 7** concludes the research by pointing out the research findings during experiments.

Chapter 2

Literature Review

2.1 Image Formation for Facial Expressions

This section reviews the methods to create facial expression images and divides them into two parts. Conventional Methods and Generative Adversarial Networks.

2.1.1 Conventional Methods

Influenced by the trendy beliefs of physiognomy in 1862, French neurologist Duchenne [18] observed the mechanism of human facial expressions to determine how facial muscles produce facial expressions. In his work, facial expressions are triggered by muscle contraction due to the stimulation of an electric probe. The induced facial expressions are documented with a recently invented camera. His work became the first documented work on human facial expressions illustrated with photographs. His experiments are revolutionary in facial expression study. He is credited for a physiological distinction between a forced smile and a genuine smile (also known as a Duchenne smile).

Although Duchenne's work became groundbreaking in facial expression research, induced facial expressions are non-spontaneous and appearance-based facial expressions, meaning they are unrelated to human emotions. Therefore, the study [19] used well-crafted romance narratives to stimulate 'jealousy'. During experiments, the participants were asked to circle the emotions introduced by the stories. Their responses were documented for the study of emotions.

Instead of the written stories, triggers to induce emotions might be varied. The study [20] used black and white photos depicting Ekman's emotions as the baseline. Different photographs of a single emotion, such as the top half of the photo illustrating sadness expressions and the bottom half depicting anger expressions, were combined using Macintosh software. The transformed photos are used as stimuli to study a mixture of emotions.

The common technique to collect facial expressions images is photography, in which facial expressions expressed by the participants are documented by a camera. The way to activate facial expressions can be diverse. One comes in written stories, and another can be photos that visualize the distinct facial expressions for each emotion. After the study between facial expressions and facial muscles by Ekman [5], their relationship is often used to train the participants to act facial expressions. The study [13, 14, 21] used the FACS [5] to train the participants before staging the facial expressions. Their responses were then photographed for further study. Most of the standard facial expressions datasets are collected using photography such as the Extended Cohn Kanade facial expressions dataset[22], Japanese Female Facial Expressions dataset[23], Multimedia Understanding Group facial expression dataset[24], and

many others.

Another technique to collect facial expressions images is image crawling with the respective keywords through the search engine. Compared to photography, it is a challenging approach as various factors such as lightning, head pose, and occlusion, are uncontrolled. Besides, the labels of these collected images are expensive as it needs human labelers to encode them. The conventional dataset such as Facial Expressions Recognition 2013 (FER 2013) [25] was created by searching the images with given a set of 184 emotion-related keywords in the Google search engine.

The last technique to create facial expressions images is 3D modeling for facial expressions. In the Facial Expression Research Group 3D Database (FERG-3D-DB) [26], the proposed system takes the 2D facial expression images of a human and learns the relationship between 2D images and character expressions to predict the 3D rig parameters of the character. The conventional methods for facial expression images are challenging as they are labor-intensive and require manual labeling. The quality of facial expressions also depends on the participants' performance to stage the facial expressions of emotions. Therefore, the automatic generation of facial expressions images to represent the mixture of emotions is desired. The next section discusses the deep generative models, especially generative adversarial networks (GANs) for image generation.

2.1.2 Generative Adversarial Networks and Its Variants

In this section, image generation by Generative Adversarial Networks (GANs) and their variation is discussed based on their individual characteristics and challenging factors.

Generative Adversarial Networks (GANs) were first introduced by Goodfellow et.al [27] as another generative modeling approach based on the game theory, where two different players compete against each other to gain their own profits. GANs include two different neural networks for the competition, named discriminator D and generator G. The idea of GANs is to learn a function that transforms the existing distribution into unknown or real distribution. Therefore, the generator takes the sample $\mathbf{z}$ from known prior distribution such as Gaussian distribution as its input and directly generates the imitated samples from the unknown distribution with learning parameters $\boldsymbol{\theta}_g$. Its opposite network, discriminator D behaves as a judging network and produces the confidence probability of being the input samples from the real distribution rather than from the generator model. Learning by GANs model is formulated as a function such as $v(\boldsymbol{\theta}_g, \boldsymbol{\theta}_d)$, in which D obtains the positive payoff whereas G receives the negative payoff. During learning, they both compete to obtain their maximum profits. Their objective function can be defined as follows.

$$\mathcal{L}_{\text{GANs}} = \min_{G} \max_{D} \; \mathbb{E}_{\mathbf{x}\sim P_X} \log D(\mathbf{x}) + \mathbb{E}_{\mathbf{z}\sim P_Z} \log(1 - D(G(\mathbf{z}))) \tag{2.1}$$

where $\mathbf{x}$ defines as the sample drawn from the real data distribution P_X. $\mathbf{z}$ refers to a sample from the existing prior distribution such as $\mathcal{N}(0, 1)$.

The advantage of GANs is that the learning process to estimate the real distribution does not require the inference function to approximate the gradients. However, training GANs are difficult and unstable in practice as simultaneously optimizing the respective cost function from two networks does not guarantee to have an equilibrium [28]. The way to train the GANs model remains the ongoing research problem in the literature.

In practice, GANs model does not take the supervised labels of the training data and creates customized labels to annotate the data samples from real and generated distributions. Therefore, practical GANs training can be denoted as an unsupervised learning approach.

The preposition behind unsupervised learning is to learn the structure existing in the training samples to infer the unknown labels [29]. As D predicts the probability of being the input sample following the training distribution, the study [29] revised the above objective function of GANs as follows.

$$\mathcal{L}_{\text{GANs}} = \min_G \max_D \mathbb{E}_{\mathbf{x}\sim P_X} \log p(y = 1|\mathbf{x}, D) + \mathbb{E}_{\mathbf{z}\sim P_Z} \log(1 - p(y = 1|G(\mathbf{z}), D)) \tag{2.2}$$

where the discriminator D is assumed to perform the binary classification problem with the sigmoid activation such as $p(y = 1|\mathbf{x}, D) = \frac{1}{1+e^{-D(\mathbf{x})}}$.

Compared to unsupervised learning, supervised learning assists the learning process by maximizing the mutual information between input and their corresponding labels and generalizing on unseen data. Therefore, in the semi-supervised learning of GANs [29, 30], the discriminator D is enforced to be a classifier for categorical recognition. As the generator G requires the discriminator to capture the distribution of the given input, D is made to perform $i + 1$ classes where the additional class refers to the classification of real and generated samples. The final activation of D is changed to softmax. The loss function of this approach can be seen as follows.

$$\mathcal{L}_{\text{SGANs}} = \min_G \max_D \mathbb{E}_{\mathbf{x}\sim P_X} \log p(y = k|\mathbf{x}, D) + \mathbb{E}_{\mathbf{z}\sim P_Z} \log(1 - p(y = k|G(\mathbf{z}), D)) \tag{2.3}$$

where p is the probability of being one of the categorical classes such as $p(y = k|\mathbf{x}, D) = \frac{e^{D_k(\mathbf{x})}}{\sum_{k=1}^{c_i+1} e^{D_k(\mathbf{x})}}$. k is the number of classes. $\mathbf{x}$ is the input samples from real data distribution P_X and $\mathbf{z}$ is the latent sample from P_Z.

2.1.2.1 Variants of Generative Adversarial Networks

After the invention of GANs by Goodfellow et.al. [27], various types of GANs models have been developed to overcome the challenges encountered in GANs' training and the common problems. They can be divided into two types in summary: architectural optimization and loss function optimization, which are discussed subsequently.

Architectural Optimization This section will discuss the important architectural revolutionaries in GANs.

Deep Convolution Generative Adversarial Networks (DCGANs) The previous vanilla GANs model is comprised of densely connected layers for both sub-models, challenging in scaling up for computer vision applications. Therefore, Deep Convolution Generative Adversarial Networks (DCGANs) is proposed by Radford et.al [31], in which densely connected layers are replaced by convolution layers to keep the spatial relationship among pixels in filtering and training stability. With convolution, nonlinear activation such as rectified linear unit (ReLU) is used for the hidden layer and tanh function for the layer in G, whereas Leaky ReLU is used in D. Besides, a batch normalization layer is added to increase the diversity among generated samples [32]. Visualizing the convolution filters provides insights into what the models see and comprehension of GANs learning. The authors also observed the generated space by linear interpolation and arithmetic operation. Although DCGANs is the major evolution from vanilla GANs, training stability, high-resolution image generation, and latent space observation remains the open research problem in GANs.

Conditional Generative Adversarial Networks (cGANs) It was proposed by Mirza et.al [33] to control the generation process by providing the extra information as a condition. The additional information can be class labels and/or different forms of data modality. Conditioning the generation process has proven to avoid the common mode collapse problem. However, it requires the labeled dataset in the case where labels are given as the condition during training.

Auxiliary Classifier GANs (ACGANs) It is an extension from cGANs and proposed by Odena et.al[34]. Although it is extended from cGANs, it does not take the labels as the condition. Instead, the discriminator is modified to perform as an auxiliary decoder to predict the probability of the image classes. As it is a variant of GANs, the decoder still requires to perform the original task to classify the distribution of the given samples. Therefore, the objective function of ACGANs consists of two parts as follows.

$$\mathcal{L}_S = \mathbb{E}[\log p(S = \text{real}|\mathbf{x})] + \mathbb{E}[\log p(S = \text{fake}|G(c, \mathbf{z}))] \tag{2.4}$$

where $\mathcal{L}_S$ is the loss function for the log-likelihood of the correct distribution of the given samples. $\mathbf{x}$ is the real sample drawn from the training distribution. $\mathbf{z}$ is the latent sample from the prior distribution. c is the label of image classes. $G(c, \mathbf{z})$ is the generated sample of the given prior latent and respective class, in other words, a fake sample.

$$\mathcal{L}_C = \mathbb{E}[\log p(C = c|\mathbf{x})] + \mathbb{E}[\log p(C = c|G(c, \mathbf{z}))] \tag{2.5}$$

where $\mathcal{L}_C$ is the loss function for the log-likelihood of the correct label.

$$\mathcal{L}_D = \max(\mathcal{L}_C + \mathcal{L}_S),\ \mathcal{L}_G = \max(\mathcal{L}_C - \mathcal{L}_S), \tag{2.6}$$

where $\mathcal{L}_D$ and $\mathcal{L}_G$ are the loss function for the discriminator D and generator G, respectively.

The modified function has proved to stabilize the GANs training and scale up to 128×128 images[34]. It has also proved that high-resolution generated images improve the distinctness among images.

Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets (InfoGANs) Chen et.al [35] proposed the InfoGANs to disentangle feature representation by maximizing the mutual information between unlabelled observation and latent codes. As the GANs take the latent samples as input for the generator without any restrictions, the semantics of the generated data is tangled in the latent space. Therefore, similar to cGANs, supplementary information such as latent code is given for the semantics structure of the generated data. Unlike to cGANs, the latent codes are learned in an unsupervised manner by regularizing the GANs' objective with mutual information between data and latent codes as follows.

$$\mathcal{L}_{\text{InfoGANs}} = \min_G \max_D \mathbb{E}_{\mathbf{x}\sim P_X} \log D(\mathbf{x}) + \mathbb{E}_{\mathbf{z}\sim P_Z} \log(1 - D(G(\mathbf{z}))) - \lambda I(\mathbf{c}; G(\mathbf{z}, \mathbf{c})) \tag{2.7}$$

where $\mathcal{L}_{\text{InfoGANs}}$ is loss function for InfoGANs. $\mathbf{x}$ is the real sample from the distribution P_X. $\mathbf{z}$ is the latent sample from the distribution P_Z. I is the mutual information between latent codes $\mathbf{c}$ and the generated samples by the generator G. λ is a hyperparameter to be learned.

In practical training, the above objective is hard to achieve as it requires the posterior distribution $P(\mathbf{c}|\mathbf{x})$. Therefore, it is revised with an auxiliary distribution to approximate the posterior distribution. The revised function can be seen as follows.

$$\mathcal{L}_{\text{InfoGANs}} = \min_{G}\max_{D}\ \mathbb{E}_{\mathbf{x}\sim P_X}\log D(\mathbf{x}) + \mathbb{E}_{\mathbf{z}\sim P_Z}\log(1 - D(G(\mathbf{z}))) - \lambda\mathcal{L}_I(G, Q) \tag{2.8}$$

where Q is the auxiliary distribution and is represented as a neural network in practice.

Progressive Growing of GANs (ProGANs) High-resolution images are demanded since they support the discriminability among generated samples, meaning that the higher-resolution images are easier to be distinguished. However, GANs training is well-known for its instability. To tackle this problem, Karras et.al [36] proposed the ProGANs with the new training approach in a progressive manner. The main idea of ProGANs training is that most of the training is done in low-resolution image space and stabilized before adding the new layers to construct the new resolution images. As the benefits of executing progressive training, it saves training time and stability, which is one of the most common problems in GANs training. Besides, ProGANs can generate high-quality images and achieve an inception score of 8.8 in the CIFAR-10 dataset.

A Style-Based Generator Architecture for Generative Adversarial Networks (StyleGANs) Since latent are sampled from the continuous prior distribution without restriction, the semantics feature representations of the generated sample are entangled in the latent space. To handle this entanglement, Karras et.al [37] proposed StyleGANs to learn different styles of data. In StyleGANs, the latent are mapped onto intermediate latent space and changed into style via affine transformation. Then, the style or semantics representation is used in the convolution layer through the adaptive instance normalization layer. Therefore, the generator can control image generation by drawing the samples of each style through the mapping network. StyleGANs achieved the Frechet Inception Score (FID) of 5.06 in Celeb-HQ [36] and 4.40 in FFHQ [37] datasets.

Loss Function Optimization As opposed to the architecture modification of GANs, many research focuses on GANs' objective function to tackle the training instability problem. The source of training instability comes from the Jensen Shannon divergence, which measures the similarity among distributions, in which the discriminator D is often better than the generator G. Therefore, other research formulated the GANs objective with another measurement and a few of them will be reviewed in this section.

Wasserstein Generative Adversarial Networks (WGANs) Arjovsky et.al [38] proposed the new objective function of GANs based on Wasserstein distance or Earth mover distance as follows.

$$\mathcal{W}(P_r, P_g) = \inf_{\gamma\in\Pi(P_r,Pg)}\mathbb{E}_{(x,y)\sim\gamma}\|x - y\| \tag{2.9}$$

where Wasserstein distance between real distribution P_r and generated distribution P_g is defined over the minimum cost function γ for moving the mass at point x to y.

Unlike GANs' objective, the Wasserstein function gives the distance value to measure the two distributions P_r and P_g. Therefore, changes need to be done in the discriminator to execute the new objective. The final activation is changed from sigmoid to linear to provide the real value instead of the confidence probability score of the input samples. Besides, weights are fixed in the range of -0.01 and 0.01 to ensure the parameters lie in the compact space. The original study also showed that Wasserstein distance-based function is better in other measurements

including Jensen Shannon divergence when both distributions lie in low dimensional manifolds and provides the continuous gradients useful for optimization. Due to its benefits, WGANs are one of the most adopted GANs variations in the literature.

Improved Training of Wasserstein GANs (WGANs-GP) The Wasserstein objective enforces the Lipschitz constraint on the discriminator or critic by clipping the weights into a certain range such as [-0.01, 0.01]. Although weight clipping is easy to implement in practice, it is not a good way to enforce the constraints and causes either vanishing or exploding gradients problems. Therefore, Gulrajani et.al [39] proposed the regularization term to enforce the constraints on the Wasserstein objective in replace of weight clipping. The new function is defined as follows.

$$\mathcal{L}_{\text{WGANs-GP}} = \mathbb{E}_{\mathbf{z}\sim P_Z} D(G(\mathbf{z})) - \mathbb{E}_{\mathbf{x}\sim P_X} D(\mathbf{x}) + \lambda(\| \nabla_{\hat{\mathbf{x}}} D(\hat{\mathbf{x}})\| - 1)^2 \tag{2.10}$$

where D is the discriminator or critic and G is the generator. The first two terms refer to the Wasserstein distance which takes real samples $\mathbf{x}$ and fake samples $\mathbf{z}$ synthesized by G. λ is the gradient penalty coefficient and $\hat{\mathbf{x}}$ is the sample form the distribution $P_{\hat{\mathbf{x}}}$ by uniformly sampling along the straight line between P_X and P_Z.

This modification no longer needs batch normalization in both sub-models, which is used to stabilize the adversarial training, as it independently penalizes the norm of the gradients of D regarding each input. Besides, it also improves image quality and supports faster convergence over the WGANs objective [39].

Least Squares Generative Adversarial Networks (LSGANs) The discriminator of the vanilla GANs used the sigmoid activation for classifying the distribution of the given samples, often leading to the vanishing gradient problem in updating the generator with the fake samples that are far from the real samples. In LSGANs [40], it was conceptualized in the two-dimensional space where the discriminator acts like a decision boundary for separating the real and fake images, whereas the generator samples the fake images in that space. However, when the fake samples are drawn far from the real samples but still on the correct side of the boundary, the choice of cross-entropy sigmoid loss in the discriminator provides very little gradient information for those fake samples, leading to a vanishing gradient problem. To tackle this problem, Mao et.al [40] replace the cross entropy sigmoid loss with least squares loss to penalize the fake samples located far from the real distribution. The objective functions for each sub-model are defined as follows.

$$\mathcal{L}_D = \min_D \Big(\frac{1}{2}\mathbb{E}_{\mathbf{x}\sim P_X}[(D(\mathbf{x}) - b)^2] + \frac{1}{2}\mathbb{E}_{\mathbf{z}\sim P_Z}[(D(G(\mathbf{z})) - a)^2] \Big) \tag{2.11}$$

where $\mathcal{L}_D$ defines as a minimized loss for the discriminator D. a and b are the labels of real and fake samples.

$$\mathcal{L}_G = \min_G \frac{1}{2}\mathbb{E}_{\mathbf{z}\sim P_Z}[(D(G(\mathbf{z})) - c)^2] \tag{2.12}$$

where $\mathcal{L}_G$ is the minimized loss for the generator G. c is the label to indicate that G wants D to believe that fake samples as real.

We can observe that both loss functions are penalized by the squared function to encourage the generator to synthesize the fake samples closer to the decision boundary, producing higher gradients and preventing the vanishing gradient problem.

2.2 Facial Expressions Recognition

This section includes the literature on emotion recognition through facial expressions. It can be broken down into two parts which are conventional methods and deep learning-based methods. The overview of a typical facial expressions recognition model can be seen in Figure.2.1.

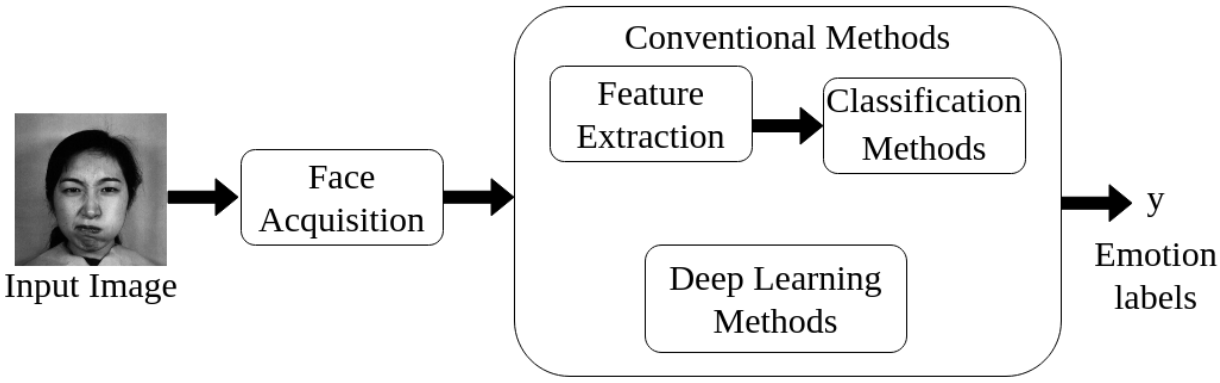


Figure 2.1: Overview of facial expressions recognition models in literature

2.2.1 Conventional Methods

In the last decade, various feature extraction methods were proposed and had been discussed in the facial expression recognition literature. A handful of baseline extraction methods used for facial expressions recognition will be discussed in this section.

As shown in Figure 2.1, conventional methods for emotion recognition typically involve two major steps: feature extraction and classification. However, it is important to note that pre-processing is also essential as conventional feature extraction methods are highly sensitive to noisy data.

One standard pre-processing step is locating the face in the image and extracting the facial region. Extensive research on face detection has been conducted, resulting in the proposal of thousands of different methods in the literature so far. Among them, the Viola-Jones face detection algorithm [41] is a common practice and worth to be mentioned. It was proposed by Viola and Jones [41] to efficiently detect faces in real time. The algorithm breaks down the image into several patches using a moving window approach and identifies special features defined by Haar-like functions within each patch. The window is moved across the entire image to check different features in different positions, as an image can include multiple faces, leading to high-dimensional feature sets. To efficiently detect faces, the Adaboost method is employed to select important features from the complete set and reduce the feature dimensions. Subsequently, a cascade of face classifiers is trained on these features. At each stage of the cascade, if a simple feature is not detected in a particular window, it implies that faces do not exist in that window, allowing for the quick elimination of irrelevant windows. This enables the algorithm to detect the regions of interest rapidly.

However, Viola and Jones's algorithm was designed to detect frontal faces and provides the best results on those faces. Therefore, if the images contain side faces or facing upwards or downwards, the other methods based on neural networks can be employed [42, 43, 44]. After face extraction, other pre-processing steps such as unifying the resolution, adjusting face orientation, enhancing pixel intensity, and improving contrast can be done to enhance the recognition performance in the later stage.

The next step involves the extraction of the positive features concerned with facial expressions in a given image. This step requires domain-level knowledge and expertise to analyze the underlying patterns within the extracted features. Therefore, the features obtained through this approach are commonly referred to as handcrafted features. Extraction methods vary on the type of facial expression features that are geometric or shape-based and appearance-based features. Geometric features refer to information related to shape and facial landmarks that are used to define facial structure. Distinct facial landmarks to define the facial structure of a particular person involve eyebrows, nose, mouth, eyes, chin, and cheek. However, geometric-based feature extraction methods often disregard facial texture or complexion. Many approaches use Active Shape Model (ASM)[45], Active Appearance Model (AAM)[46], Constraint Local Model (CLM)[47] and their variations to track and extract the geometric representations. The appearance-based methods include Local Binary Pattern (LBP)[48], Scale Invariant Feature Transform (SIFT)[49], Histogram of Gradient (HoG) [50, 51], Gabor filter or wavelets [52, 53, 54]. Hybrid features can be created to take advantage of the respective types of features [55].

Before the final classification, dimensionality reduction is an optional step to select the important or significant features. Principal Component Analysis (PCA)[56] is one of the several dimension reduction methods. In the final step, those handcrafted features are employed to train the classifier to recognize the emotion labels. Commonly used classification methods in facial expressions recognition systems include Support Vector Machine (SVM)[57], Linear Discriminative Analysis (LDA)[48], Random Forest (RF)[58], Adaboost [59], Decision Trees[60], and Naive Bayes[61].

2.2.2 Deep Learning Methods

As stated in the previous section, conventional methods are vulnerable to noisy data and rely on domain expertise. However, real data comes with high variance, and conventional methods are not sufficient to handle this type of data. Additionally, facial expression recognition systems based on conventional methods are not end-to-end systems, meaning the entire process from feature extraction to classification is not seamless, and manual intervention is necessary to enhance recognition performance. To address these issues, deep learning-based approaches have been proposed. With the advances in technology in computing devices and resources, countless deep learning-based networks had been developed so far. Therefore, this section discusses the fundamental networks used in the facial expressions recognition system.

Convolution Neural Networks (CNNs) were proposed by LeCun et al. [62] in 1998. The properties of convolution filters used in CNNs, such as sparse interaction, translation equivalence, and parameter sharing, contribute to the success of image recognition tasks. A typical CNNs includes down-sampling operations such as pooling or stride convolution, as well as a fully connected layer at the end for classification, which results in a seamless recognition system. The initial part of the CNN is responsible for the feature extraction process, and subsequent layers, such as the pooling layer, perform dimensionality reduction. Finally, fully connected neurons perform non-linear transformations on the feature maps filtered by the convolution operations. Architectural variations of CNN applied in facial expressions recognition can be seen in [63, 64, 65].

Deep Belief Networks (DBNs) were proposed by Hinton et al.[66] to address issues such as getting stuck at the local minimum due to the slow learning rate in deep networks during back-propagation. DBNs are unsupervised graphical models designed to capture complex patterns in training data. They consist of a series of Restricted Boltzmann Machines (RBMs), which

are stochastic generative neural networks that learn the input distribution. The output of one RBM is fed as input to another RBM in the series. Unlike RBMs, there is no communication among nodes within layers. The greedy learning algorithm is used to pre-train the DBN. In contrast to Convolution Neural Networks (CNNs), which extract basic features such as edges, shapes, and facial parts at early layers and more task-specific features at later layers, each layer in a DBN learns representations of the entire input. DBNs can be employed in applications of image recognition tasks such as facial expressions recognition [67, 68, 69].

Deep Auto Encoder (DAE) is an unsupervised generative model which involves symmetrical networks, named encoder and decoder. The objective of DAE is to encode the data in a compact space to reduce the dimensions and learns the significant representation from the input. The learning is regulated through the reconstruction error by the decoder which re-ensembles the latent from the low-dimensional space into the input sample. DAE overcomes the shortage of label data and is able to learn patterns in input in an unsupervised way with the use of a decoder network. Therefore, it can be applicable to many applications such as noise reduction, dimensionality reduction, and recognition tasks such as facial expressions[70, 71]. It is also important to note that another generative model, Generative Adversarial Networks (GANs) broadly discussed in the previous section, can be used together with DAE [72] that uses the representations learned by the discriminator as a baseline for reconstruction. Once the training had been finished, the pre-trained discriminator can be fine-tuned to classify the facial expressions.

Data can be not only static images but also image sequences or sequential data. Recurrent Neural Networks (RNNs) are specifically designed to handle temporal data by leveraging their internal memory to retain information or patterns from previous time steps, enabling them to make predictions for future steps. Within RNNs, nodes possess feedback loops that enable them to consider not only the current input but also information from recent previous inputs. Except for their internal memory, it is similar to the feedforward networks and can be trained through back-propagation through time (BPTT)[73]. RNNs are well-suited for tasks involving time series data, such as natural language processing for sentiment analysis, as well as computer vision tasks like real-time facial expression recognition or video analysis.

During the training of RNNs, they typically encounter exploding gradients or vanishing gradients. To address this problem, Long Short Term Memory (LSTM)[74] is extended from RNNs including their expanded internal memory. Cell state is regulated through the use of three gates such as an input gate to allow or modify the state by the input signal, an output gate to enable or restrict the cell state to influence the other neurons, and a forget gate to modulate the recurrent connection or erase the previous state. Due to their design, LSTM had been widely employed in handling sequential data including video facial recognition tasks.

Chapter 3

Data Preparation

This chapter explains the type of dataset and essential pre-processing step required to train the proposed methodology in the next chapter.

3.1 Dataset

In this research, three benchmark datasets are used to test the proposed model during the experiment. They are Extended Cohn Kanade Facial Expressions Dataset (CK+)[22], Japanese Female Facial Expression Dataset (JAFFE)[23] and the Multimedia Understanding Group (MUG) Facial Expression Dataset[24].

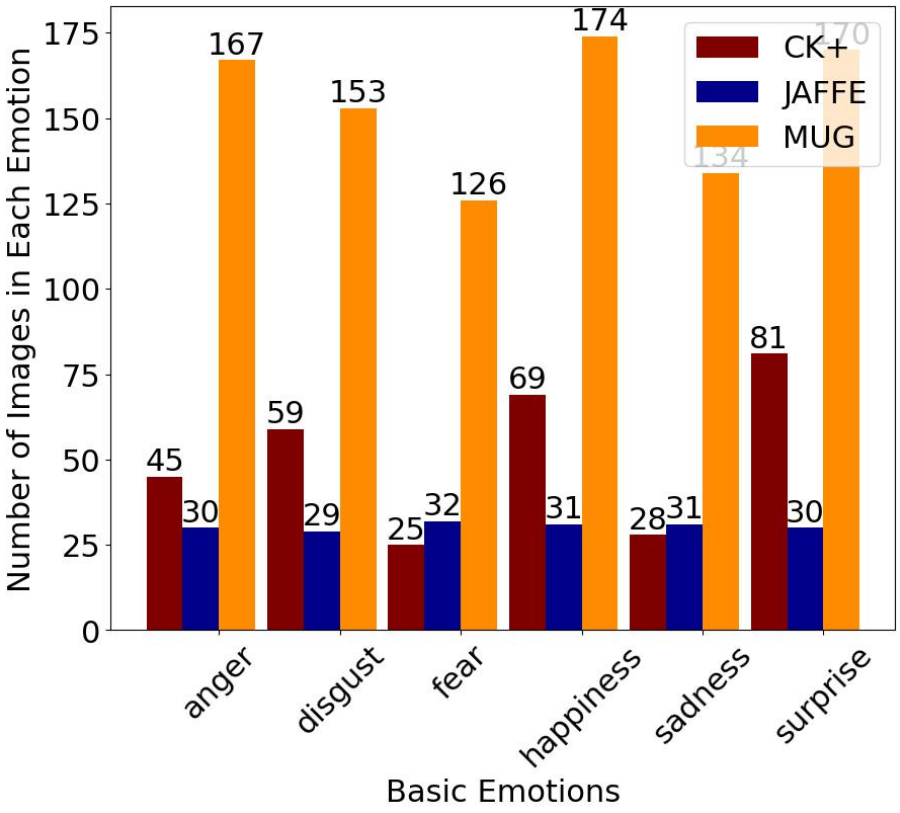
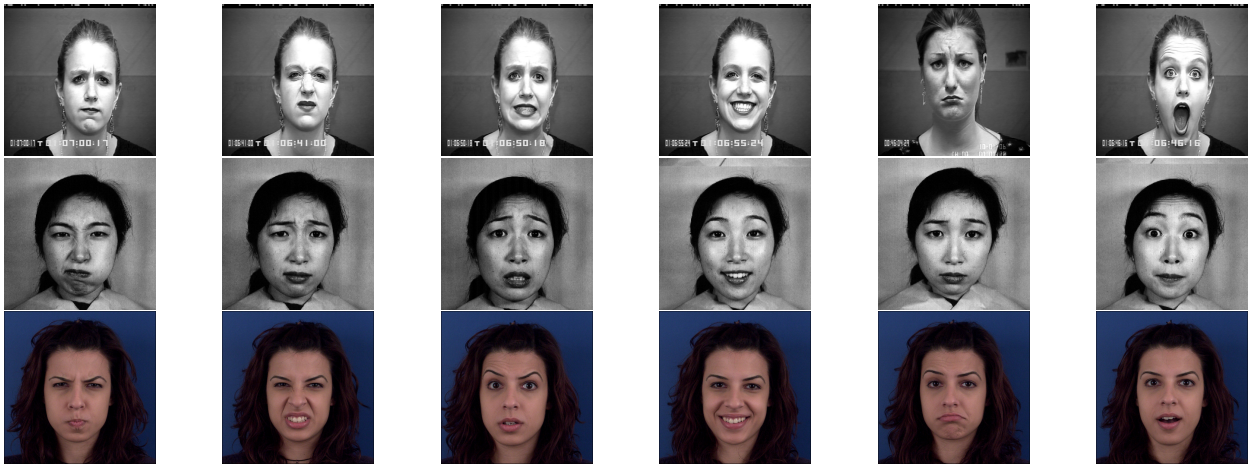


Figure 3.1: Number of peak images in each basic emotion in CK+, JAFFE and MUG dataset

The CK+ dataset includes 123 subjects who are 81% from Euro-American, 13% from Afro-American, and 6% from others. Ages span from 18 to 50 years and 69% are female. Each

sequence started with a neural face and ended with a peak expression. The peak frame is coded with FACS[5] for action units (AUs) which can later be used for emotion recognition such as anger, contempt, disgust, fear, happiness, sadness, and surprise.

The JAFFE dataset contains 10 subjects who are Japanese females to simplify the experiments. Occlusion such as facial hairs is excluded to show expressive facial area while being photographed under uniform illumination. Emotion labels are graded by 92 external Japanese female undergraduates for the degree of facial expressions of each basic emotion such as anger, disgust, fear, happiness, sadness, and surprise on a Likert scale.



(a) Anger (b) Disgust (c) Fear (d) Happy (e) Sadness (f) Surprise

Figure 3.2: Examples of six basic emotions from the CK+ (upper row), JAFFE (middle), and MUG (lower row)

The MUG dataset includes videos of 86 subjects who are in the age range of 20 to 35 years and belong to the Caucasian race. Out of all subjects, 59% are male. Data from 52 subjects are permitted to be used by external users. All frame sequences are captured under uniform light conditions and no occlusion is included. The subjects are instructed on how to exhibit facial expressions for the six basic emotions, as per the FACS manual guide [5]. The order of sequences is neutral, onset, apex, offset, and back to neutral again. The number of images that contributed to each six basic emotions is shown in Figures. 3.1. Sample images for each dataset are illustrated in Figure.3.2.

3.2 Pre-processing

The initial step in face-related tasks is face extraction to remove extraneous information from the input data, which is not useful for recognition tasks. Processing of raw input images is necessary because they include additional details like background and time steps that are not useful in the task. This research uses the pre-processing stages, which are depicted in Figure.3.3.

Pre-processing includes locating the face region in the input image (face detection), outlining the detected face (face localization), and retrieving the localized area (face extraction). There are various face detection algorithms, including classical and deep learning-based approaches. Classical algorithms utilize various feature representations, including the Histogram of Gradients (HoG) feature descriptor[75], rectangular Haar-like features[41], Active Shape Models (ASM)[45], and Active Appearance Models (AAM)[46], along with classification methods such as linear Support Vector Machine (SVM) and cascade classifiers. However, classical

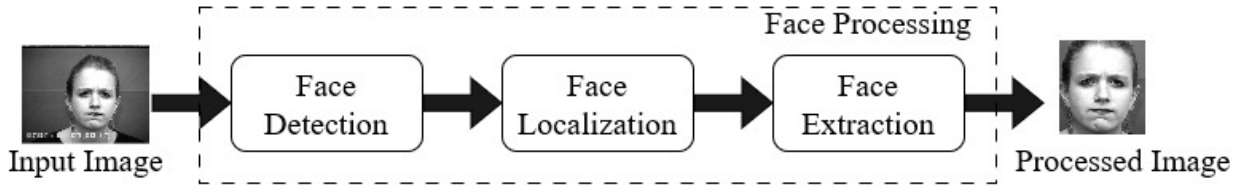
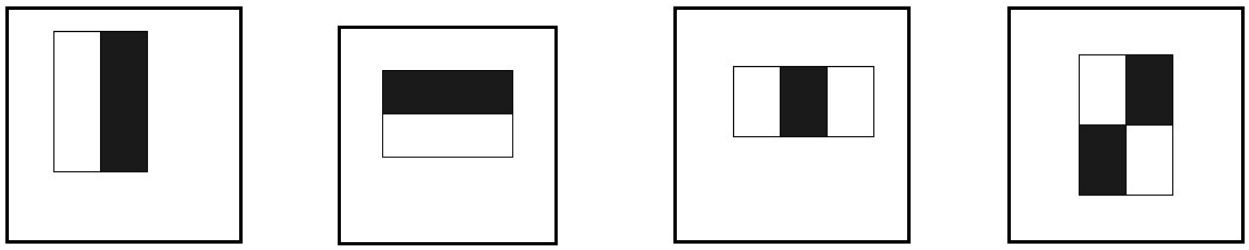


Figure 3.3: Overview of face processing stages

algorithms are sensitive to face orientation, lighting conditions, and occlusion. To address these challenges, deep learning-based algorithms like Multi-Task Cascaded Convolutional Neural Networks (MTCNN)[76], and Single Shot Multibox Detectors (SSD)[77] have been developed.



(a) (b) (c) (d)

Figure 3.4: Three types of Haar's kernel functions defined in Viola and Jones' algorithm[41], (a) and (b) are 2 rectangles Haar's features for vertical and horizontal edges, (c) is 3 rectangles feature for lines, and (d) is 4 rectangles features for complex structure.

Although deep learning-based face detection algorithms have advantages over the classical approach, the images used in this research are simple frontal face images associated with emotional labels. Therefore, the classical approach, named Viola and Jones' face detection algorithm[41] is sufficient enough to detect faces in the images. It defines features using Haar's basic functions[78] as the window that transverses across the image. They are 2 rectangles for edges, 3 rectangles for lines, and 4 rectangles for any structure where there are changes in pixel intensities as in Figure.3.4.

Given an image patch $\mathbf{I}$, the Haar values for the edge can be calculated as follows.

$$\Delta_{\text{edge}} = \sum_u \sum_v \mathbf{I}_b(u,v) - \sum_u \sum_v \mathbf{I}_w(u,v) \tag{3.1}$$

where b refers to the black and w refers to the white region in the detecting window. u and v is the coordinate of image path $\mathbf{I}$. Δ_{edge} is the difference of the summed values of pixel intensities between the black and white region in the given image patch.

For line features,

$$\Delta_{\text{line}} = \sum_u \sum_v \mathbf{I}_b(u,v) - \left(\sum_u \sum_v \mathbf{I}_{w_1}(u,v) + \sum_u \sum_v \mathbf{I}_{w_2}(u,v) \right) \tag{3.2}$$

where w_1 and w_2 are the two white rectangles, whereas b is the black rectangle in the detecting windows. Δ_{line} indicates the presence of changes along the line by detecting a darker region in the image path surrounded by lighter regions from both sides.

For any structural changes across a diagonal, Haar values can be computed as follows.

$$\Delta_{\text{complex}} = \left(\sum_u \sum_v \mathbf{I}_{b_1}(u,v) + \sum_u \sum_v \mathbf{I}_{b_2}(u,v) \right) - \left(\sum_u \sum_v \mathbf{I}_{w_1}(u,v) + \sum_u \sum_v \mathbf{I}_{w_2}(u,v) \right) \tag{3.3}$$

where Δ_{complex} indicates the presence of changes in intensity values across the diagonal.

The Viola and Jones algorithm[41] utilizes the integral image to speed up the feature extraction process. The integral image is a cumulative sum of pixel values above, left, and including the values at the current point (x,y), as defined in Equation.3.4.

$$\mathbf{I}'(u,v) = \sum_{u' \leq u, v' \leq y} \mathbf{I}(u', v') \tag{3.4}$$

where $\mathbf{I}'$ refers to the integral image and $\mathbf{I}$ is the original image.

This allows us to retrieve Haar features for any rectangular window with fewer operations. However, the resulting feature set is over 180,000 and is infeasible to use directly. To address this issue, the AdaBoost technique is used to select important features from the feature set. It employs a weak classifier for each feature and selects features that produce a lower error to form the final set of important features. This reduces the feature set to over 6,000 important features.

The classifier is then trained in a cascade fashion, with 38 stages in a detecting window for face detection. If a simple feature is not detected in the early stage of a particular window, it is discarded. Otherwise, it moves to the next stage. This cascade classifier significantly reduces processing time for finding a face, as most windows are discarded in the earlier stages, allowing for fast computation of face detection. The Viola and Jones algorithm is supported by many open-source tools. In this research, we used OpenCV[79] for the implementation. The output of face processing is shown in Figure.3.5.

(a) Input image (b) localization (c) extraction

Figure 3.5: Face processing results, (a) is an input image, (b) is a result of face localization which bounded the detected face region returned by Viola and Jones' algorithm[41], (c) is the result of face extraction for the bounded area.

3.3 Peak Frame Selection

Sometimes data comes in a sequence to express facial expressions, for example, the CK+ and MUG datasets. In this research, we focus on working with still images. Therefore, we need to

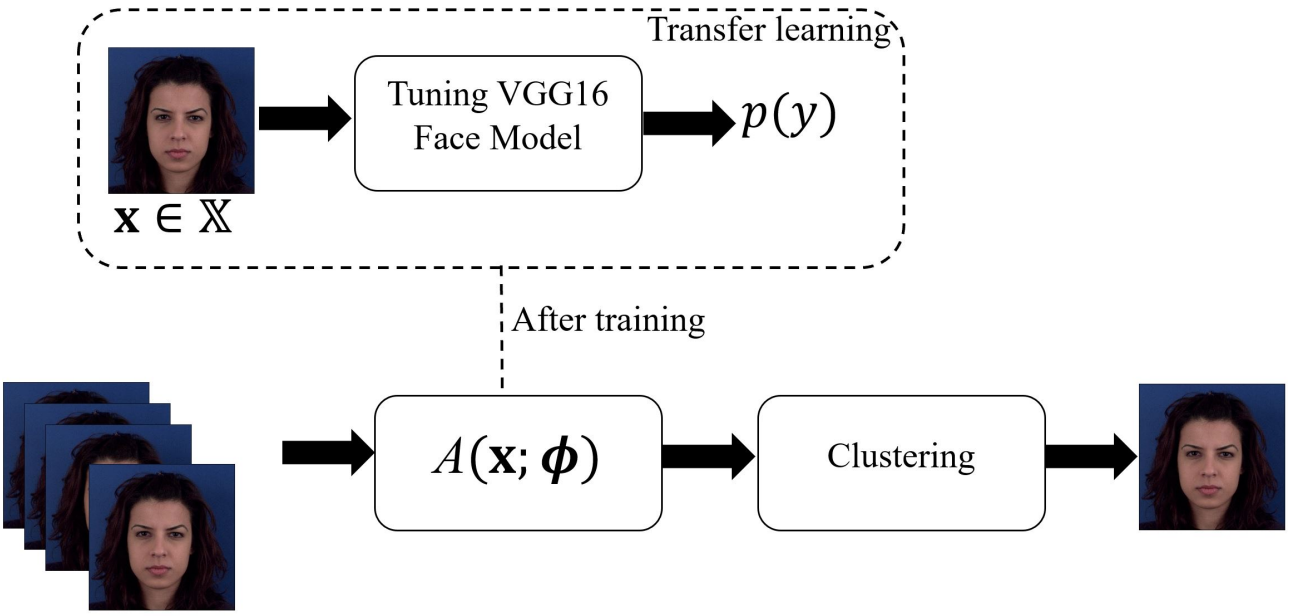


Figure 3.6: Overview of peak frame selection framework. A represents recognition model parameterized by learned weights $\boldsymbol{\phi}$ to extract the facial expression features. $\mathbf{x}$ represents the image from the training set $\mathbb{X}$. $p(y)$ represents the confidence probability of being label y.

pick the frame that has the most expressive facial expressions. In the CK+ dataset, all sequences start with a neutral expression and end with the peak expression for each emotion. However, not every dataset follows the same temporal pattern. In the MUG dataset, all sequences start with a neutral state, followed by onset, apex, offset, and neutral temporal patterns. Therefore, selecting the peak frame is necessary.

The peak frame can be defined as the image that has features drastically dissimilar to the ones in the neutral state. Previous research on peak frame selection has had initial requirements, such as the presence of a neutral frame. Therefore, the peak frame can be identified by finding the frame with the maximum difference among feature values. One study [80] computed the dissimilarity matrix for all frames based on chi-squared distances. The distance values were calculated among Local Phase Quantization (LPQ) features, and the frame with the maximum distance score was selected as the peak frame. Similarly, another study [81] used three different features (constrained local model, binary local pattern, and optical strain) on the region of interest and searched for the peak frame using binary search.

The conventional approach requires a neutral frame for reference. However, this condition cannot always be satisfied when dealing with real data. Moreover, feature extraction with traditional methods can be challenging. Therefore, we propose a deep learning-based methodology for peak frame selection [82]. We hypothesize that the probability of a particular emotion class being represented in the peak frame is higher than the score in the neutral frame since the peak frame includes the most expressive facial expressions. The selection process can be categorized into two stages: training a facial expression recognition model (transfer learning) and clustering the extracted facial expression features for the peak frame selection. It is depicted in Figure.3.6. Each stage is discussed in sequential order.

Training a facial expression recognition model Transfer learning is the training technique to transfer the knowledge that has been learned in the past in solving the new task [28]. Knowledge gained from previous training is adjusted based on the new domain and the final

layers of the model are tuned for the new task. For example, if the model learns the representations of visual categories such as faces as in the previous task, it can share the low-level features such as edges, shapes, texture, and changes in lightning for the next task with a few training examples. This technique can significantly reduce the training time and computational resources required for training a model from scratch, and can often lead to better performance on the new task, especially when the new dataset is small or similar to the original dataset used from previous training.

Due to the benefits of transfer learning, the model that was previously trained on face domain is adapted to train on facial expressions recognition. According to the past research on transfer learning from previous face recognition setting [83, 84, 85, 86], the deep neural network, named VGG16 [87] that was trained with large face dataset to identify the identities provides relatively good performance on the facial expressions recognition task. VGG16 face model comprises five major convolution blocks before the pooling layer and three dense layers at the end. To determine the suitable layer in tuning the model for facial expressions recognition, the first 9 feature maps of final convolution layers from each block are visualized as a convolution output for interpretation in Figure 3.7.

As per the illustration results, each convolution filter focuses on the different features of a given input image. For example, at the early stage, convolution filters capture the details of the image. Some filters highlight on different facial parts of the image such as eyes, mouths, cheeks, forehead, or face shape while others focus on the image background. As moving into deeper layers, they capture the general features to make the final classification of the target task. Based on feature maps from the final block, filters emphasize more on facial areas such as the eyes, forehead, chin, bridge of the nose, nose, and temple area in the face. Similar to face recognition tasks, facial expression recognition also requires to detect the different prominent facial parts as changes are made in those parts. Therefore, the last three dense layers after the final convolution blocks are considered as candidate layers to tune the model.

Two new dense layers are added after each candidate layer of the pre-trained model. A thousand images are randomly selected from the MUG dataset for each emotion for training. Given the training set $\mathbb{X}$, the new weights are updated through the Adam optimizer over 64 training batches. The softmax activation in Equation 3.5 is applied in the last layer to ensure that the output sums up to 1, following the probability distribution.

$$P(y|\mathbf{v}) = \frac{e^{v^{(j)}}}{\sum_{c=1}^{6} e^{v_c}} \tag{3.5}$$

where e is the standard exponential function applied at each attribute j of input vector $\mathbf{v}$. c is the index for emotion classes. y is the output class.

Their performance is measured in terms of accuracy in Equation 3.6 where the value of accuracy a is computed over frequency ω of correct prediction per class c_i and total number of samples $\mathbb{X}$ used in the evaluation.

$$a_{c_i} = \frac{\omega_{c_i}}{|\mathbb{X}_{c_i}|} \tag{3.6}$$

Table 3.1 shows the performance of the model tuned after each potential layer. Their average accuracy for each tuned model is calculated as in Equation. 3.7.

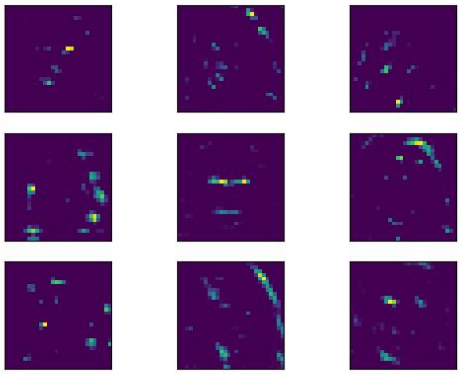
$$\hat{a} = \frac{1}{6}.\sum_{i=1}^{6} a_{c_i} \tag{3.7}$$

(a) Input image

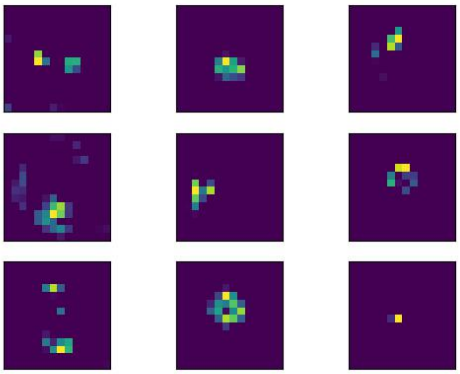
(b) Convolution block 1

(c) Convolution block 2

(d) Convolution block 3



(e) Convolution block 4



(f) Convolution block 5

Figure 3.7: Visualization of first 9 feature maps from final convolution layers of each block in VGG face model

The tuned model after FC7 gives the best performance among the three candidates, and therefore, it is selected for further processing.

Table 3.1: Accuracy of tuned VGG face model after last three dense layers (FC6, FC7, FC8) for facial expressions recognition task

Accuracy a_{c_i}				
i	class c_i	FC6	FC7	FC8
1	anger	76%	95%	26%
2	disgust	64%	82%	15%
3	fear	75%	62%	0%
4	happiness	79%	91%	28%
5	sadness	71%	82%	28%
6	surprise	79%	64%	29%
Average accuracy ($\hat{a}$)		74%	**79%**	21%

Clustering Clustering is an unsupervised learning algorithm to group samples that have similar features or characteristics together in the same groups. The goal of clustering is to identify the underlying pattern presented in the features. Our objective is to categorize the image sequences into two distinct groups - peak-like and neutral-like clusters, depending on the level of facial expressions exhibited in the images. Therefore, with clustering, we can gather the instances that have similar facial expressions intensity levels. Suppose that facial expressions features extracted by the tuned model are

$$\mathbf{f}_i^{(j)} = [f_i^{(1)}, f_i^{(2)}, f_i^{(3)}, ..., f_i^{(j)},], j \in \mathbb{R}^{128} \tag{3.8}$$

where i is the index for the number of feature instances, and j is the index for the number of feature attributes returned by the tuned model A, which is parameterized by $\boldsymbol{\phi}$.

Kmeans++ clustering [88] is performed on those sets of facial expressions features. The number of clusters, k is set to 2 as the features are clustered into two groups that are peak-like and neutral-like clusters. Then, the first center, $\mathbf{c}_1$ is uniformly selected from instances as follows.

$$\mathbf{c}_1 \in_R \{\mathbf{f}_1^{(j)}, \mathbf{f}_2^{(j)}, \mathbf{f}_3^{(j)}, ..., \mathbf{f}_i^{(j)}\} \tag{3.9}$$

where R represents randomization. The next center $\mathbf{c}_2$ is initialized based on the distance between the data instances $\mathbf{f}_i^{(j)}$ and the first center $\mathbf{c}_1$ with the probability of $\mathbf{f}_i^{(j)}$ being proportional to d^2. It is formulated as follows:

$$\mathbf{c}_2 = \max_i \sqrt{\sum_{j=1}^{128} (f_i^{(j)} - c_1^{(j)})^2} \tag{3.10}$$

After the initialization of the centers for each cluster, the distance between instances and each center is computed. Each instance is assigned into the cluster with minimum distance values as follows.

$$\min_i \left(\sqrt{\sum_{j=1}^{128} (f_i^{(j)} - c_1^{(j)})^2}, \sqrt{\sum_{j=1}^{128} (f_i^{(j)} - c_2^{(j)})^2} \right) \tag{3.11}$$

Afterwards, the algorithm re-calculates new centers for every cluster by taking into account the instances that are contained within each respective cluster. It is formulated as follows.

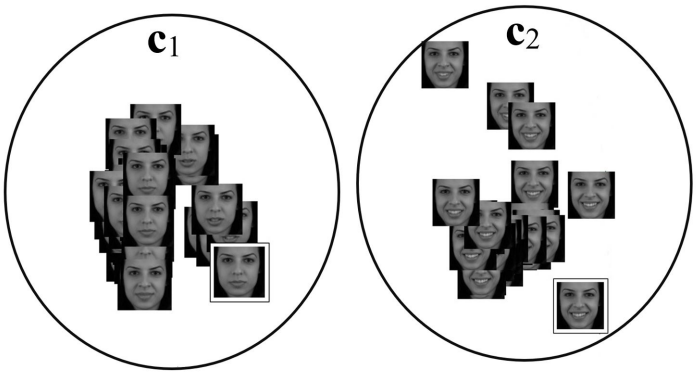
$$\mathbf{c}_k = \frac{1}{|\mathbf{c}_k|} \sum_{i=1}^{n} \mathbf{f}_i^{(j)} \tag{3.12}$$

where i is the index for instances and n is the total number of instances contained in the cluster $\mathbf{c}_k$, indicating $\mathbf{f}_i^{(j)} \in \mathbf{c}_k$. Equations 3.10 and 3.12 is executed until cluster centers remain unchanged.

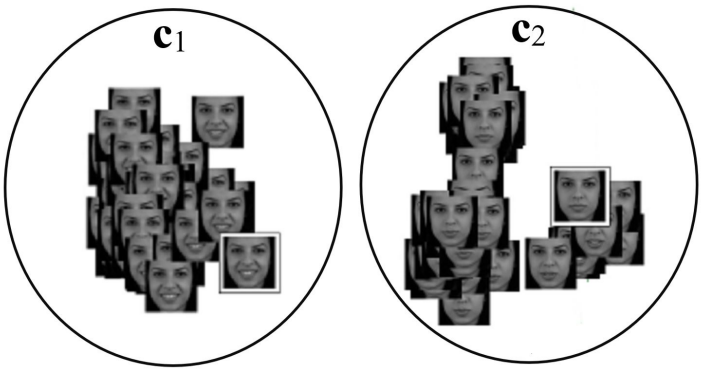
Next, the data instance closest to each center is chosen as key frame as in Equation. 3.13, which gives two key frames to represent their own group.

$$\mathbf{f}_{i,\mathbf{c}}^{(j)} = \min_i \sqrt{\sum_{j=1}^{128} (f_i^{(j)} - c^{(j)})^2}, \ \mathbf{c} \in \{\mathbf{c}_1, \mathbf{c}_2\} \tag{3.13}$$

where $\mathbf{f}_{i,\mathbf{c}}^{(j)}$ refers the nearest sample to center $\mathbf{c}$ in its respective group.



(a) Results on the image sequences captured during the *first take* for subject ID 001.

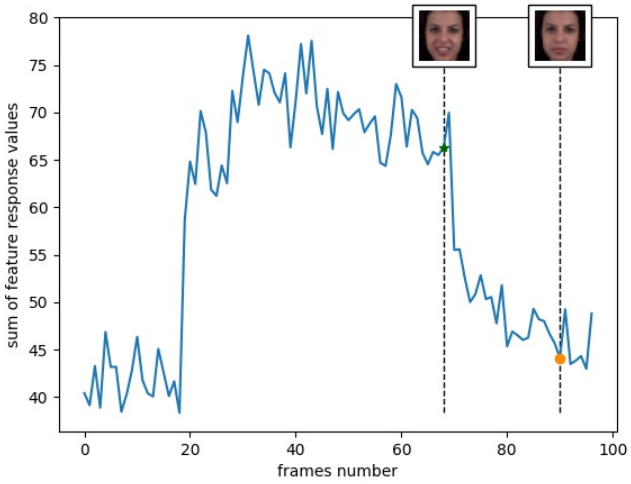


(b) Results on the image sequences captured during the *second take* for subject ID 001.

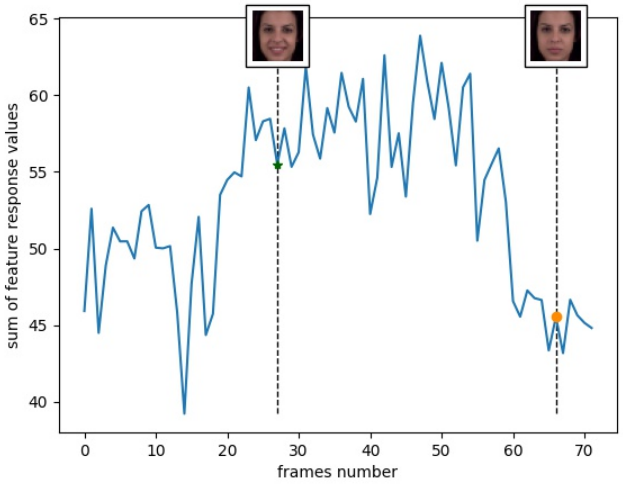
Figure 3.8: Visualization the clustering results, $\mathbf{c}_1$ refers to first cluster and $\mathbf{c}_1$ refers to second cluster. The rectangular box represents the key frame of the corresponding cluster.

The clustering results and their corresponding key frames are shown in Figure 3.8. As depicted in Figure 3.8a, the first cluster $\mathbf{c}_1$ contains neural-like expressions, while the second cluster $\mathbf{c}_2$ contains peak-like expressions. However, Figure 3.8b presents these interpretation in the opposite manner. Our hypothesis is that the key frame that corresponds to the peak group can be identified by comparing the total feature values between the peak key frame and the neutral key frame, with the expectation that the former would have a higher total value.

Figure 3.9 displays the total response values of the features for two image sequences. Based on the figure, total values are low at the beginning and end of sequences. High values are



(a) Results on the image sequences captured during the first take for subject ID 001.



(b) Results on the image sequences captured during the second take for subject ID 001.

Figure 3.9: Visualization the total feature response values with two key frames from each cluster. ★ refers to peak key frame. ● refers to neutral frame.

secured in the middle as the sequences from the MUG dataset follows the temporal pattern such as neutral, onset, apex, offset, neutral in that order. Therefore, as a next step, total features response values are computed in Equation.3.14 which returns two summation values for peak-like and neutral-like clusters.

$$s_{\mathbf{c}_k} = \sum_{j=1}^{128} f^{(j)} \tag{3.14}$$

where j represents an index of attribute of the 128-dimension features. s is the summation result for its corresponding cluster $\mathbf{c}_k$.

Next, the total values are compared and the key frame with higher values are defined as peak-like cluster that gathers the frames with higher features intensity. It is formulated as follows.

$$\mathbf{c} = \max_{\mathbf{c}}(s_{\mathbf{c}_1}, s_{\mathbf{c}_2}) \tag{3.15}$$

where $\mathbf{c}$ refers to the peak-like cluster. The key frame belong to this class is chosen as peak frame and used in further processing.

3.4 Image Pair Selection

In our experiments, we focused on selecting peak images with expressive facial expressions for each emotion. Since the last frame of the CK+ sequences is labeled as the peak frame, we only selected the last image for further processing. For the MUG dataset, we used the proposed framework from the previous section to select the peak image. All of the examples from the JAFFE dataset were employed because they were all peak frames. However, apart from CK+, the other datasets contained six basic facial expression images from the same subject. Therefore, we paired different facial expression images from CK+ based on the similarity score of the facial features.

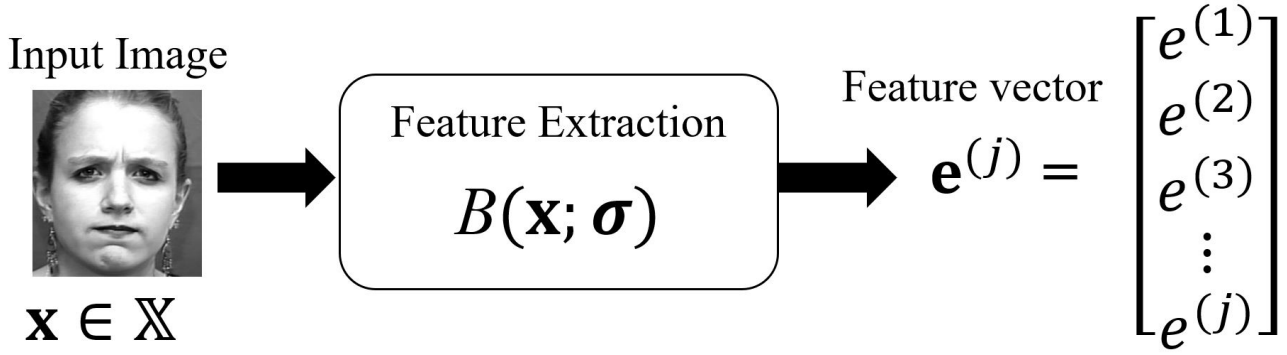


Figure 3.10: Overview of facial feature extraction by pre-trained model B parameterized by trained weights $\boldsymbol{\sigma}$[89]. $\mathbb{X}$ is the set of training images. j is the feature attribute where $j \leq 128$.

Since CK+ images contain both color and grayscale images, all the images are converted into grayscale to maintain consistency. Before the selection, a set of candidate images for a particular image is chosen based on gender, such as male-to-male or female-to-female. Afterwards, the selection process involves two stages: feature extraction (Figure.3.10) and similarity measurement (Figure.3.11). In this research, we use the pre-trained model reported by Geitgey[89] for the feature extraction process, as it achieved high accuracy (up to 99%) on challenging faces datasets such as the Labeled Faces in the Wild. The implementation is supported by open-source tools, named Dlib [90].

Given an image $\mathbf{x}$ from the candidate set $\mathbb{X}$, the pre-trained model B with trained parameter $\boldsymbol{\sigma}$ [89] encodes the faces into 128-dimensional feature space, where similar faces are located closer to each other. It is depicted in Figure.3.10. All facial image from CK+ are encoded in this way. After encoding the faces, degree of similarity among features are computed as follows.

$$s_{k,l} = \frac{\mathbf{e}^j_{k,c_1}.\mathbf{e}^j_{l,c_2}}{\|\mathbf{e}^j_{k,c_1}\|.\|\mathbf{e}^j_{l,c_2}\|} \tag{3.16}$$

where $\mathbf{e}$ is the encoded features in j dimensions where $j \leq 128$. c_i represents the emotion class where $c_1 \neq c_2$. k and l are the image indices from the classes c_1 and c_2 respectively. s is the similarity score with the range of -1(dissimilarity) and +1(similarity).

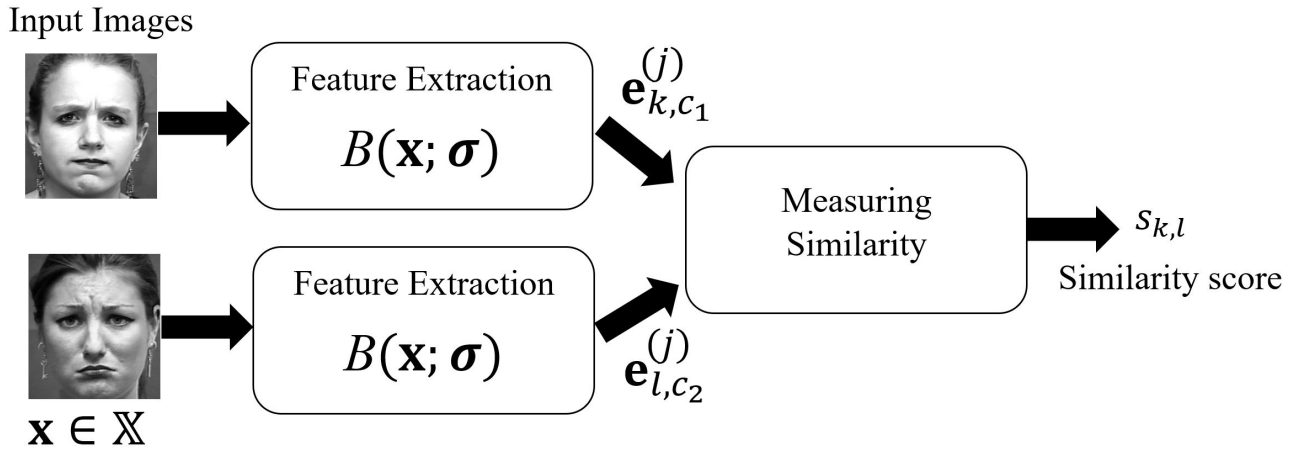


Figure 3.11: Overview of measuring the similarity among facial features. Pre-trained model B parameterized by $\boldsymbol{\sigma}$ takes training sample $\mathbf{x}$ and returns 128-dimensional facial features where $j \leq 128$. c_i represents the emotion class where $c_1 \neq c_2$.

For a particular image $\mathbf{x}_k$ in class c_1, all instances from a different class c_2 are taken into consideration. The image with the minimum score from c_2 is reported as a match for that image in c_1. This can be formulated as follows:

$$l = \underset{l}{\operatorname{argmin}}\{s_{k,l} | \forall s \in S, l \leq M\} \tag{3.17}$$

where s is the similarity score from the scores set S in the class c_2 for a particular image $\mathbf{x}_k$ from c_1. M is the total number of image instances in c_2. Detail processing steps are explained in Algorithm.1.

3.5 Morphing

Given two images $\mathbf{x}_{k,c_1}$ and $\mathbf{x}_{l,c_2}$ ($\mathbf{x} \in \mathbb{X}$), morphing creates the intermediate image $\mathbf{I}$ between $\mathbf{x}_{k,c_1}$ and $\mathbf{x}_{k,c_2}$ by warping the pixels of the same local region from each respective input.

$$\mathbf{I}(u,v) = (1-\alpha)\mathbf{x}_{k,c_1}(u,v) + \alpha\mathbf{x}_{l,c_2}(u,v); \tag{3.18}$$

where $\mathbf{I}$ is the morphed image between $\mathbf{x}_{k,c_1}$ and $\mathbf{x}_{l,c_2}$. α is the control parameter for pixels information to be morphed at the location (u, v). k and l are the image indices in the corresponding classes c_1 and c_2.

Detailed steps for image morphing are described in Algorithm 2. First, distinct facial features such as eyes, and nose are located by a pre-trained landmark detector (Dlib[90]) that returns coordinates of 68 facial landmarks. Additional points at four corners and their middle are added to the list, making a total of 76 landmarks as shown in Figure.3.12. Those points are used as vertices to find the Delaunay triangles.

Delaunay triangulation divides the image plane into several small triangles and rejects the skinny triangle during division. This property helps the morphing process not to process the region that is not very perceptible. Triangles from each corresponding image are transformed and warped. Warped images are used for morphing.

The morphing process is controlled by a parameter α, which ranges from 0 to 1, inclusive. It controls the amount of information from each input to be morphed. For example, when α is 0, much pixels information at position (u and v) from $\mathbf{x}_{c_1}$ is applied in the morphing process

Algorithm 1: Algorithm to select the compatible different facial expressions images with the most similar facial features

Input: Candidate images $\mathbf{x} \in \mathbb{X}$ for classes c_1 and c_2. Pre-trained model $B(\mathbf{x}; \boldsymbol{\sigma})$ to encode faces into features into feature space.

Output: Indices of compatible image pair for morphing

$S_1 \leftarrow \phi$;
$k \leftarrow 0$;
while $k < N$ **do**
    /* N is the number of images in c1 class */
    $\mathbf{e}^{(j)}_{k,c_1} \leftarrow B(\mathbf{x}_{k,c1})$ /* $j \in \mathbb{R}^{128}$ */
    $S_1 \leftarrow S_1 \cup \mathbf{e}^{(j)}_{k,c_1}$
end
$S_2 \leftarrow \phi$;
$l \leftarrow 0$;
while $l < M$ **do**
    /* M is the number of images in c2 class */
    $\mathbf{e}^{(j)}_{l,c_2} \leftarrow B(\mathbf{x}_{l,c_2})$
    $S_2 \leftarrow S_2 \cup \mathbf{e}^{(j)}_{l,c_2}$
end
for $k \leftarrow 0$ **to** N **by** 1 **do**
    $S \leftarrow \phi$;
    for $l \leftarrow 0$ **to** M **by** 1 **do**
        $s_{k,l} = \frac{\mathbf{e}^{(j)}_{k,c_1} . \mathbf{e}^{(j)}_{l,c_2}}{\|\mathbf{e}^{(j)}_{k,c_1}\| . \|\mathbf{e}^{(j)}_{l,c_2}\|}$ /* Cosine similarity metric */
        $S \leftarrow S \cup s_{k,l}$
    end
    $l = \underset{l}{\operatorname{argmin}}\{s_{k,l} | \forall s \in S, l \leq M\}$;
end
return k, l

Figure 3.12: Detected landmarks by Davis [90] with additional points

and the result looks exactly like $\mathbf{x}_{c_1}$. When α is 1, the result has the same appearance as $\mathbf{x}_{c_2}$. In this research, α is set as the value of 0.5.

Algorithm 2: Algorithm for image morphing

Input: Two images $\mathbf{x}_{k,c_1}$ and $\mathbf{x}_{l,c_2}$ from the training set $\mathbb{X}$, facial feature detector $B(.;\sigma)$ provided by dlib open source library [90]. $E(.)$ is the function to find the delaunay triangulation that returns a list of triangles to morph. $W(.)$ is a warping function for affine transformation.

Output: Morphed image $\mathbf{I}$

$\mathbf{e}_{k,c_1}(u,v) \leftarrow C(\mathbf{I}_{k,c_1})$

$\mathbf{e}_{l,c_2}(u,v) \leftarrow C(\mathbf{x}_{l,c_2})$;

`/* e is the detected landmark features by B. u,v is pixel coordinates. */`

$\mathbf{e}(u,v) \leftarrow (1-\alpha)\mathbf{e}_k(u,v) + \alpha\mathbf{e}_l(u,v)$

$T_1 \leftarrow D(\mathbf{e}_k(u,v))$

$T_2 \leftarrow D(\mathbf{e}_l(u,v))$

$T \leftarrow D(\mathbf{e}(u,v))$

`/* T1, T2, T are the list of delaunay triangles. */`

for $m \leftarrow 0$ **to** $|T_1|$ **by** 1 **do**

  $T_{1,m} = A.T_m^{\mathrm{T}}$ `/* A is affine transformation matrix */`

  $\mathbf{x}'_{k,c_1,m} = W(T_{1,m})$

end

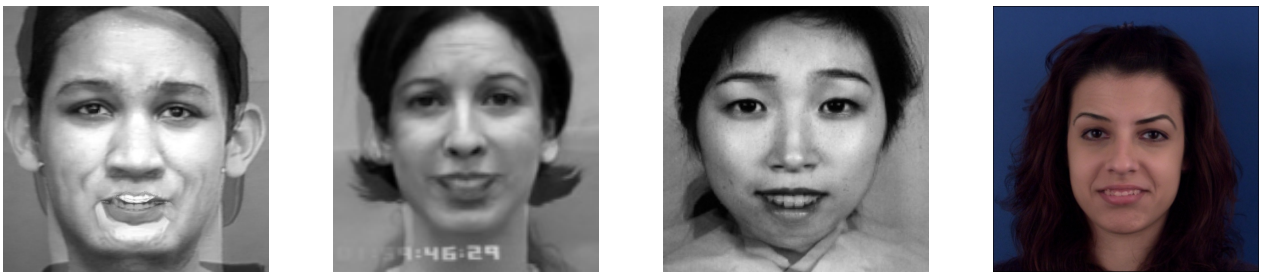
for $m \leftarrow 0$ **to** $|T_2|$ **by** 1 **do**

  $T_{2,m} = A.T_m^{\mathrm{T}}$

  $\mathbf{I}'_{l,c_2,m} = W(T_{2,m})$

end

$\mathbf{I}(u,v) = (1 - \alpha)\mathbf{x}'_{k,c_1}(u,v) + \alpha\mathbf{x}'_{l,c_2}(u,v)$



(a) Random selection (CK+)

(b) Similarity based selection (CK+)

(c) Identity-based selection (JAFFE)

(d) Identity-based selection (MUG)

Figure 3.13: Output of morphing

Figure.3.13 illustrates the output of the morphing process using different approaches for image pair selection. Since the subjects in JAFFE and MUG have images for different emotions, images from those datasets are paired based on their subjects' identity. Images from CK+ are paired based on their similarity scores, as described in the previous section. Compared to the results of random pairing, the results of morphing seem better when using similarity-based selection, especially when performed on opposite emotions such as happiness and sadness. The morphed images are used as training data to train the proposed emotion generative adversarial networks model in the next chapter.

Chapter 4

Generating Complex Facial Expressions Images for Mixed Emotions

The objective of this dissertation aims to estimate mixed emotions using synthesized images. To accomplish this objective, we propose the methodology using the analysis-by-synthesis approach described in Figure.4.1. This chapter discusses the proposed generation models to generate the mixed facial expressions images and the next chapter includes the evaluation of those synthesized mixed facial expressions images. Afterward, the facial expressions recognition models will be discussed to estimate the mixed emotion labels.

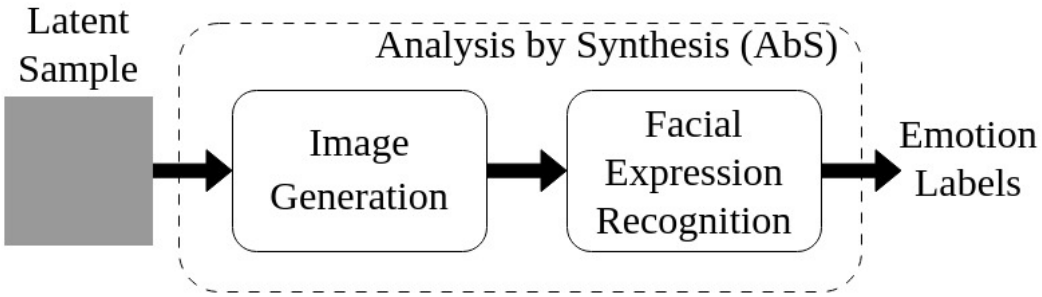


Figure 4.1: Overview of the proposed model to estimate mixed emotion labels through analysis-by-synthesis approach

In this chapter, we present four different proposed models and methodologies during the study. It can be categorized into two main groups as in Figure.4.2: Unsupervised and Supervised. The first three models are designed to solve the training stability with different loss functions and improve image quality. The last model is designed to control the image generation process. All models include their network configurations, loss functions used during training, and their respective challenges. They are described in the following order.

- Emotion Generative Adversarial Networks (EmoGANs)[91]
- EmoGANs with Wasserstein distance-based objective function (EmoGANs1)[92]
- methodology for generating a mixture of facial expressions images for complex emotion (EmoGANs2)[93]
- Conditional EmoGANs with identity preservation (EmoGANs3)

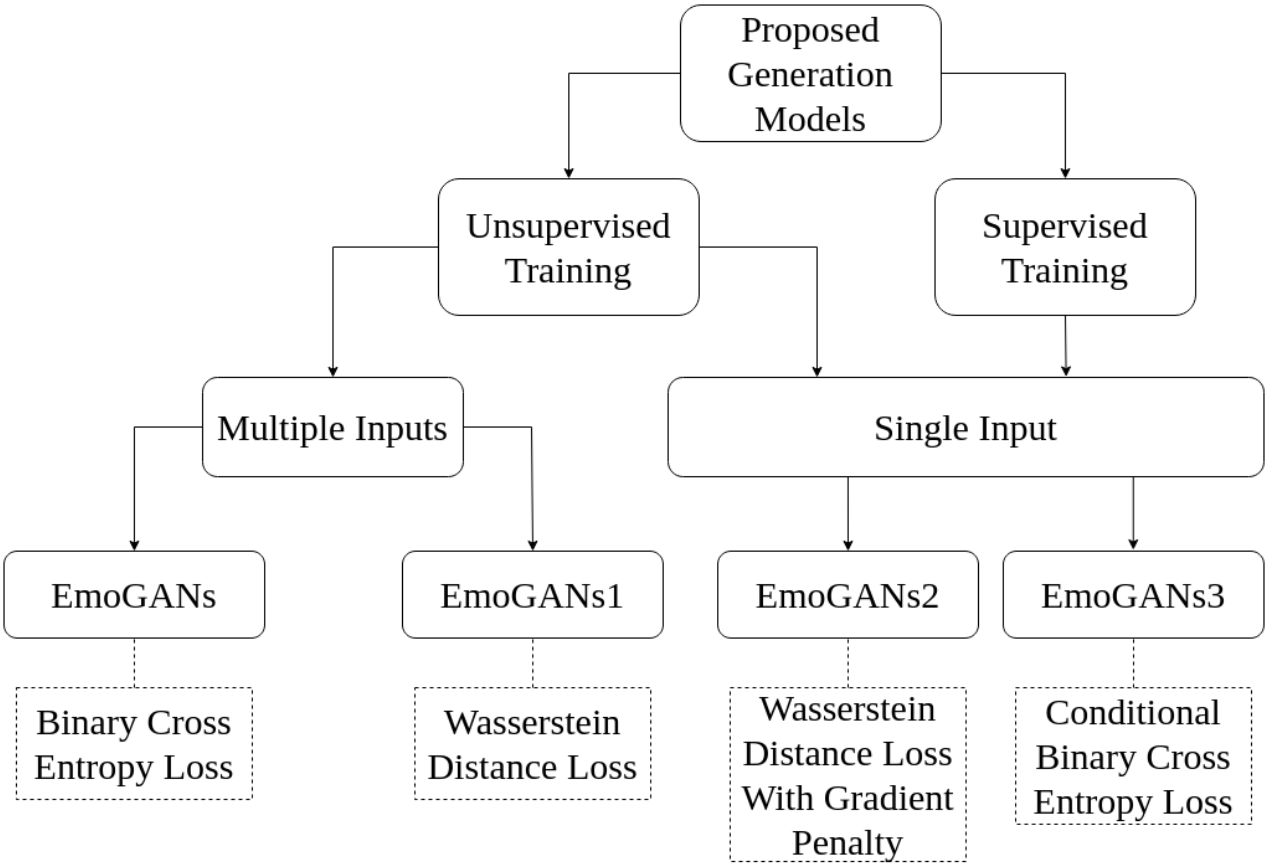


Figure 4.2: Overview of proposed image generation models

4.1 Unsupervised Training

This section includes the respective network configurations and challenging factors covering their advantages and disadvantages for the first three models: EmoGANs, EmoGANs1, and EmoGANs2.

4.1.1 Emotion Generative Adversarial Networks (EmoGANs)

This section presents the proposed Emotion Generative Adversarial Networks (EmoGANs)[91]. The background architecture of EmoGANs is based on Deep Convolution Generative Adversarial Networks (DCGANs) [31] that use the convolution layers unlike the vanilla GANs by [27].

Vanilla GANs are composed of fully connected layers in two sub-models (Generator, G and Discriminator, D). The challenging part of vanilla GANs is that G must not be over-updated without updating D and weights updates should be harmonized because it uses a single loss function (minimax loss) among models. Over-updating often leads to a mode collapse problem, in which G fails to cover the full dataset and focuses on generating the best outputs for particular class instances without considering diversity. It is also referred to as the Helvetica scenario in [27]. Besides, it used densely connected layers as its architecture, making it computationally expensive for unstructured high-dimensional data such as images. DCGANs cover these challenges by replacing dense layers with stride convolutions.

4.1.1.1 Exploring the advantages of convolution layers over densely connected layers

Suppose that $\mathbf{x}$ is the sample from training data distribution $\mathbb{X}$, represented by 3x3 matrix as follows.

$$\mathbf{x}^{(j)} = \begin{bmatrix} x^{(1)} & x^{(2)} & x^{(3)} \\ x^{(4)} & x^{(5)} & x^{(6)} \\ x^{(7)} & x^{(8)} & x^{(9)} \end{bmatrix} \tag{4.1}$$

where j is an attribute contributed in the sample $\mathbf{x}$ from distribution $\mathbb{X}$.

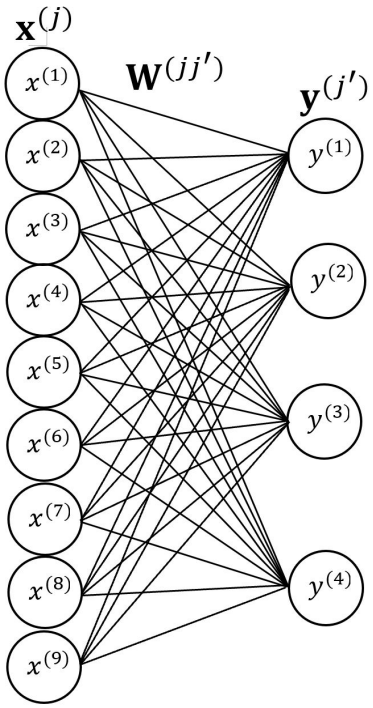


Figure 4.3: A simple example of a fully connected network with input and output layers, $\mathbf{x}^{(j)}$ is the input neuron and $\mathbf{y}^{j'}$ is the output neuron. $\mathbf{W}$ is the weighted matrix. j and j' are the feature indices for $\mathbf{x}$ and $\mathbf{y}$ respectively.

An example fully connected network (FNN) is constructed with two layers (input and output) in Figure.4.3, which has 9 input neurons at input layer and 4 neurons at output layer. The value of output neuron $\mathbf{y}^{(j')}$ can be calculated as follows.

$$\mathbf{y}^{(j')} = \mathbf{W}^{(jj')} \ . \ \mathbf{x}^{(j)} \tag{4.2}$$

Based on the FNN example, the input sample is flattened and each input neuron is connected to every single neuron at the output layer. Values of output neurons are computed through weighted transformation by input neurons in Equation 4.2, indicating every neuron at the input layer impact every output neuron regardless of their relation. When input data comes to FNN, they are flattened without considering their spatial position. Flattening might not have an impact on tabular data, but it affects image data because adjacent pixel values are meaningful in the image. In the example, input data have 9 dimensions for illustration purposes. However, image data are in high dimensions. Working with high-dimensional images is computationally expensive in FNN because it includes every input in the calculation for a single output.

For an example of a convolution neural network (CNN), suppose that the weighted kernel is defined by 2x2 matrix as follows.

$$\mathbf{W} = \begin{bmatrix} w^{(1)} & w^{(2)} \\ w^{(3)} & w^{(4)} \end{bmatrix} \tag{4.3}$$

The convolution can be formulated by using the weighted kernel as follows.

$$\mathbf{y}^{(j')} = \mathbf{W} \ * \ \mathbf{x}^{(j)} \tag{4.4}$$

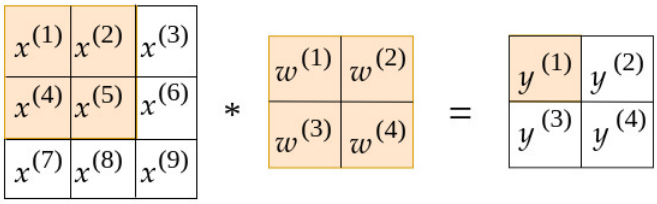


Figure 4.4: Matrix form of convolution operation in Convolution Neural Network with input and output layers

Figure.4.4 illustrates the convolution process described in Equation.4.4, in which each output is calculated based on the position of adjacent pixel values in a window sliding across the input matrix. In addition, weighted kernel can be assigned any matrix which is smaller than the input matrix. When the input is an image, the weighted matrix for FNN is exponentially growing, whereas in CNN, the small matrix can still be used by sliding across the input for calculation and output dimension can significantly reduced by pooling operations. To summarize, CNN is suitable for high-dimensional image data over FNN.

Due to the benefits of CNN over FNN for image data, DCGANs replace the dense layer with a strided convolution layer, that learns its own spatial downsampling in discriminator D and upsampling in generator G. Asides from convolution operations, it uses batch normalization that normalizes the input to have zero mean and unit variance to help the network training and avoid the poor weight initialization problem during training. The benefit of how batch normalization helps deep models during their training is discussed in [94].

In DCGANs, batch normalization is applied to all layers except D's input layer and G's output layer. Adding batch normalization helps the deep generator for gradient flows and prevents it from the mode collapse problem. Instead of max-out activation in vanilla GANs, DCGANs employ bounded activation functions such as Rectified Linear Unit (ReLU) in G and Leaky ReLU in D. Once DCGANs had been trained, the trained G and D can be reused as the feature extractors for the supervised task. It is proven to have about 82% accuracy by using trained models for classifying CIFAR 10 dataset in [31] and outperformed the Kmeans-based method as a feature extractor. The proposed model, Emotion Generative Adversarial Networks (EmoGANs)[91], derives benefit from adopting the structure of DCGANs and further modifying it to align with the research objectives.

4.1.1.2 Network configurations of Emotion Generative Adversarial Networks

Like other GANs, EmoGANs[91] consist of two sub-models, generator G and discriminator D to compete for a zero-sum game. Unlike the others, G takes three inputs, such as two images with different facial expressions to represent distinct emotions and a random latent sampled from the Gaussian distribution as in Figure. 4.5. The zero padding layer is employed onto two inputs to

control the dimension reduction after applying convolution filters in the next layer and to have a uniform dimension for feature fusion at the concatenation layer. Each convolution layer is followed by a dropout layer to regulate the training and avoid overfitting. On the other hand, the random latent is upsampled with transposed convolution followed by dropout operation. Once features from all inputs had been extracted, they are combined at the concatenation layer and upscaled by transposed convolutions until the desired output resolutions, that is 64×64.

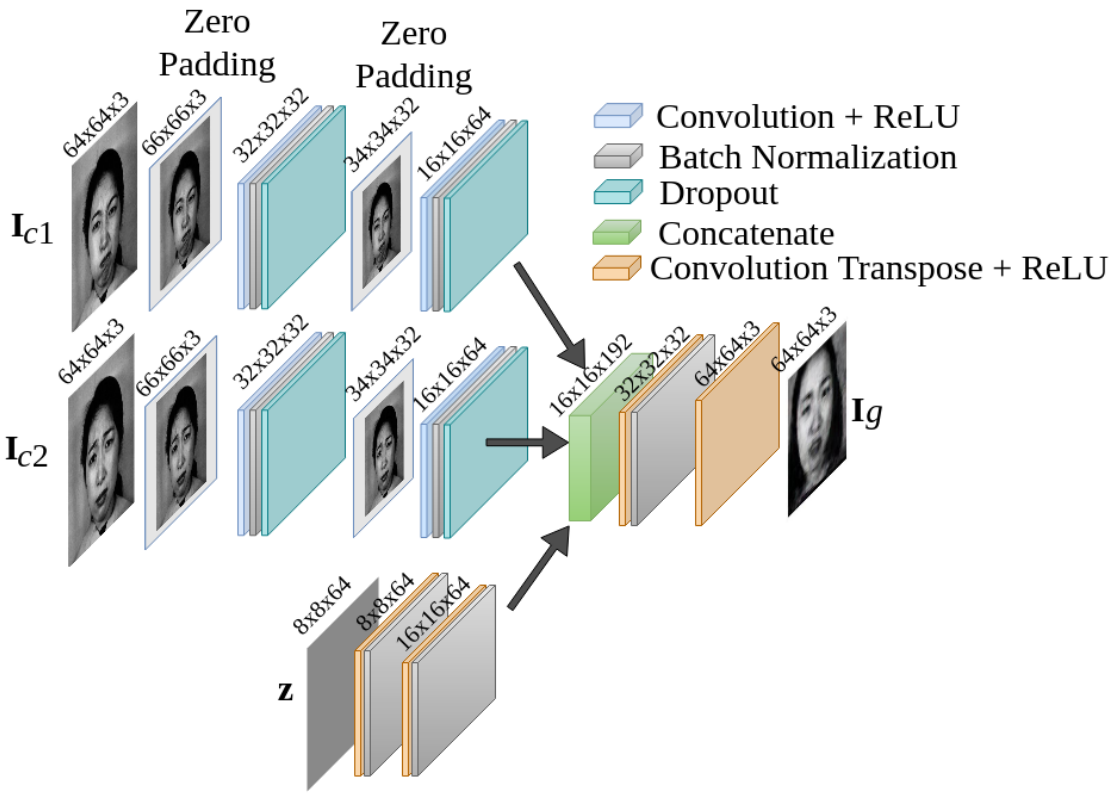


Figure 4.5: Overview of EmoGANs' generator

The configuration of D is a simple 2-layered convolution neural network as in Figure.4.6 as the datasets are simple and do not include sophisticated facial expressions images such as lateral facial or occluded images. D takes two types of input images such as one comes from real data distribution, P_X and the other comes from generated data distribution P_g by G. The objective of D is to predict the probability distributions of input samples by applying sigmoid activation, yielding the value 1 to represent P_X and 0 for P_g.

4.1.1.3 Network Training

The objective of Generative Adversarial Networks (GANs)[27] is to train two models in a way that neither can decrease its own cost function without modifying the parameters of the other model in competition as in Equation 4.5. Their weight updates can be done by any gradient-based optimization algorithm. In vanilla GANs, it was achieved by momentum-based stochastic gradient descent optimizer to accelerate the convergence and avoid local minima.

$$\min_G \max_D \mathrm{V}(D,\ G) = \mathbb{E}_{\mathbf{x}\sim P_X}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim P_Z}[1 - \log D(\,G(\mathbf{z}))] \tag{4.5}$$

where $\mathbf{x}$ is the sample from training distribution P_X and $\mathbf{z}$ is the latent sample from Gaussian distribution P_Z.

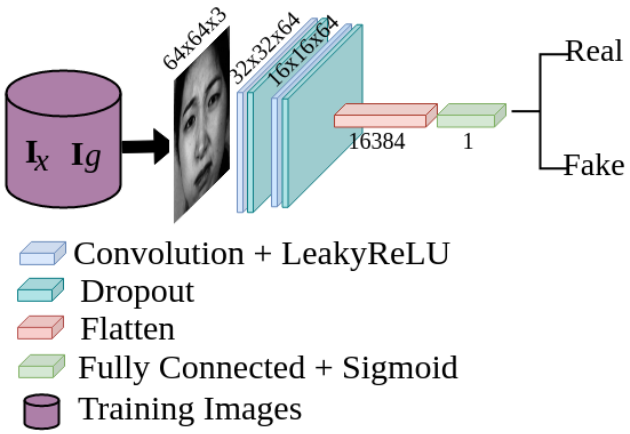


Figure 4.6: Overview of EmoGANs' discriminator

Gradient descent is an optimization technique that leverages the first-order derivatives or the slope of an objective function to discover the global minimum within the search space by traversing along the negative direction of the slope. Due to its reliance on the function's gradient or slope, the search process may oscillate within the search space, occasionally resulting in upward movements that can delay the realization of the global minimum. To overcome this issue, a hyper-parameter called momentum γ can be introduced to accelerate the search. This parameter controls the amount of past gradients during updates and its values range from 0.0 to 1.0.

$$\mathbf{v}^{(t)} = \gamma.\mathbf{v}^{(t-1)} + \lambda.J^{'}(\boldsymbol{\theta}) \tag{4.6}$$

where γ is the momentum to accelerate the search. λ is the learning rate. $J^{'}$ is the first order derivative of the objective function parameterized by weights $\boldsymbol{\theta}$. $\mathbf{v}$ is the velocity.

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \mathbf{v}^{(t)} \tag{4.7}$$

where $\boldsymbol{\theta}$ is the set of learning weights and $\mathbf{v}$ is the velocity to accelerate the gradient towards convergence.

For instance, $J(\boldsymbol{\theta})$ represents an objective function parameterized by a set of parameters $\boldsymbol{\theta}$. The update rules for parameter updates using momentum are outlined in Equations 4.6 and 4.7. Equation 4.6 calculates the current position $\mathbf{v}^t$ at time step t by combining the previous position weighted by the momentum γ and the gradient multiplied by the learning rate or step size λ. The current position is later used in the weights update step, as specified in Equation 4.7. Since momentum accumulates past gradients over iterations, it can help to reduce search oscillation and enable the search to make progress even in flat search spaces, where the gradient is zero. One disadvantage of using momentum-based gradient descent is that as the search approaches the global minimum, it tends to oscillate within and around the minimum due to its use of high momentum.

Therefore, as an alternative, Adaptive Moment Estimations, known as Adam optimizer [95] is used in DCGANs[31] for weight updates. Similar to the momentum gradient descent (MGD) algorithm, Adam is also designed to accelerate the search process by adapting the learning rate λ for each parameter. Steps for Adam optimizer are described by two functions named InitializeParameters and UpdateParameters in Algorithm 3. The Adam optimizer involves two-moment vectors that are the first and second moments of the gradient for an objective

function, symbolically represented as $\mathbf{m}$ and $\mathbf{v}$, which are initialized as zeros at the beginning of the search.

During the search, $\mathbf{m}$ is updated with hyper-parameter β_1 weighted by partial gradient g. $\mathbf{v}$ is also updated based on the hyper parameter β_2 multiplied by g^2, which is the element-wise square of g. Since both $\mathbf{m}$ and $\mathbf{v}$ employ zero-based initialization, both values are corrected by dividing their respective hyper-parameters decayed over time. In the final stage, the weights are updated based on the biased-corrected moment vectors. The advantages of Adam are computational efficiency and effective performance with larger parameters. Moreover, in contrast to MGD, it decelerates progress and prevents oscillation when approaching convergence at the minimum. Therefore, it becomes a default optimizer in updating the weights of the deep neural networks, which contain millions of parameters. Due to its benefits, this research utilizes the Adam optimizer and updates the weights over mini-batch samples.

To align with the objectives of the research, the cost function of EmoGANs is updated as in Equation 4.8.

$$\min_G \max_D \mathrm{V}(D,\ G) = \mathbb{E}_{\mathbf{x}\sim P_X}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim P_Z}[1 - \log D(\ G(\mathbf{x}_{c_1}, \mathbf{x}_{c_2}, \mathbf{z}))] \tag{4.8}$$

where $\mathbf{x}$ is the sample from real data distribution P_X. $\mathbf{x}_{c_1}$ and $\mathbf{x}_{c_2}$ are training samples from $\mathbb{X}_1$ and $\mathbb{X}_2$ for corresponding classes c_1 and c_2. $\mathbf{z}$ is the latent sample from Gaussian distribution P_Z.

As illustrated in Figure.4.5, G takes three inputs, in which two inputs ($\mathbf{x}_{c_1}$, $\mathbf{x}_{c_2}$) are training samples for emotion classes c_1 and c_2. The third input $\mathbf{z}$ is the sample from the prior distribution. For D, it takes the morphed images as training samples. During batch production, the sample indices are randomly generated and recorded. Based on those indices, the morphed images ($\mathbf{x}$) and their corresponding image pairs ($\mathbf{x}_{c_1}$, $\mathbf{x}_{c_2}$) are selected and used to update weights over those samples.

4.1.1.4 Discussion

The EmoGANs model is trained on 64 × 64 images, which are normalized before training. It takes latent samples from a Gaussian distribution $\mathcal{N}(0, 1)$. The models are updated through the Adam Optimizer over batches of size 128. During hyperparameter tuning, the default learning rate of the Adam optimizer is not suitable for the model and causes the mode collapse problem, where a small subset of images is repeatedly produced by the generator and the generation does not cover the variation of the entire dataset.

Since a larger learning rate updates a larger amount of weights, it makes the model adapt to the problem faster and might overlook the optimal points. To remedy the problem, we set a smaller learning rate of 5e-06 to make smaller updates for each update. However, this smaller learning rate requires longer training time. Therefore, we train the model for 5000 training iterations. The same settings are applied for all datasets. In spite of being the same setting, the quality of generated images is different. Based on our trial experiments during model training, the model produces good images for CK+ after completing the set training iteration, whereas the intermediate models, after some iteration, work better for JAFFE and MUG compared to the end model. Those trained models are selected for image generation.

To assess the quality of the generated images, a handful of random latent and random image pairs were generated to synthesize the images using the trained generator. The output is shown in Figure 4.7. The trained generator mapped the input samples onto a 64×64 image space. From the results, it is observed that the generated images are noisy, causing the facial

Algorithm 3: EmoGANs' mini-batch stochastic training with Adam optimizer

Input: e, b, n, $\mathbf{z}_b$, $\mathbf{x}_b$ /* e is number of iterations. b is the mini-batch size for each update. n is the number of mini-batch sizes. z is the random latent drawn from P_g. x is training sample from P_X. $R$is the random function that takes start and end values and returns a list of random numbers with size b. */

Function InitializeParameters() **is**

- $\mathbf{m}^{(0)} \leftarrow 0$; /* m is first moment. */
- $\mathbf{v}^{(0)} \leftarrow 0$; /* v is second moment. */
- $t \leftarrow 0$; /* t is time step. */
- **return** $\mathbf{m}^{(0)}$,$\mathbf{v}^{(0)}$, t

end

Function UpdateParameters(θ^t, f, $\beta_1 = 0.9$, $\beta_2 = 0.999$) **is**

- $t \leftarrow t + 1$
- $g^{(t)} \leftarrow f'(\theta^{(t-1)})$ /* g is slope of f */
- $\mathbf{m}^{(t)} \leftarrow \beta_1.\mathbf{m}^{(t-1)} + (1 - \beta_1).g^{(t)}$ /* β_1,β_2 are decay rates */
- $\mathbf{v}^{(t)} \leftarrow \beta_2.\mathbf{v}^{(t-1)} + (1 - \beta_2).g^{2(t)}$ /* $g^2 = g \odot g$ */
- $\hat{\mathbf{m}}^{(t)} \leftarrow \frac{\mathbf{m}^{(t)}}{1-\beta_1^t}$ /* β_1^t is the β_1 with power t */
- $\hat{\mathbf{v}}^{(t)} \leftarrow \frac{\mathbf{v}^{(t)}}{1-\beta_2^t}$ /* β_2^t is the β_2 with power t */
- $\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{\theta}^{(t-1)} - \lambda.\frac{\hat{\mathbf{m}}^{(t)}}{\sqrt{\hat{\mathbf{v}}^{(t)}}+\epsilon}$ /* λ is the learning rate. */
- **return** $\boldsymbol{\theta}^{(t)}$

end

$\boldsymbol{\theta}_d \leftarrow$ InitializeParameters(); /* initialize D's parameters */

$\boldsymbol{\theta}_g \leftarrow$ InitializeParameters(); /* initialize G's parameters */

foreach iteration e **do**

- **foreach** mini batch n **do**
  - $\mathbf{L} \leftarrow R(0, |\mathbb{X}|), b)$ /* random numbers between 0 and b */
  - $\mathbb{Z} = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_b\}$ /* for latent */
  - $\mathbb{X}[\mathbf{L}] = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_b\}$ /* for morphed images */
  - $\mathbb{X}_{c_1}[\mathbf{L}] = \{\mathbf{x}_{c_1,1}, \mathbf{x}_{c_1,2}, ..., \mathbf{x}_{c_1,b}\}$ /* for emotion c_1 */
  - $\mathbb{X}_{c_2}[\mathbf{L}] = \{\mathbf{x}_{c_2,1}, \mathbf{x}_{c_2,2}, ..., \mathbf{x}_{c_2,b}\}$ /* for emotion c_2 */
  - $f_d = \frac{1}{b}\sum_{l=1}^{b}[\log D(\mathbf{x}_l) + \log(1 - D(\,G(\mathbf{x}_{c_1,l}, \mathbf{x}_{c_2,l}, \mathbf{z}_l)))]$
  - UpdateParameters($\boldsymbol{\theta}_d^t, f_d$) /* Update D's parameters */
- **end**
- $\mathbb{Z} = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_b\}$ /* for new batch of latent */
- $f_g = \frac{1}{b}\sum_{l=1}^{b}\log(1 - D(\,G(\mathbf{x}_{c_1,l}, \mathbf{x}_{c_2,l}, \mathbf{z}_l)))$
- UpdateParameters($\boldsymbol{\theta}_g^t, f_g$) /* Update G's parameters */

end

expressions in the generated images cannot be clearly visible. However, it can be perceived that the generated images resemble their corresponding dataset.

In comparison to other datasets, JAFFE includes only ten subjects and it is easy to determine variations in image generation. Based on the results shown in Figure. 4.7 (middle row), the image generation for JAFFE covers the entire range of data variations. Analyzing the results for CK+ in Figure 4.7 (upper row), the results include mixed genders and diverse

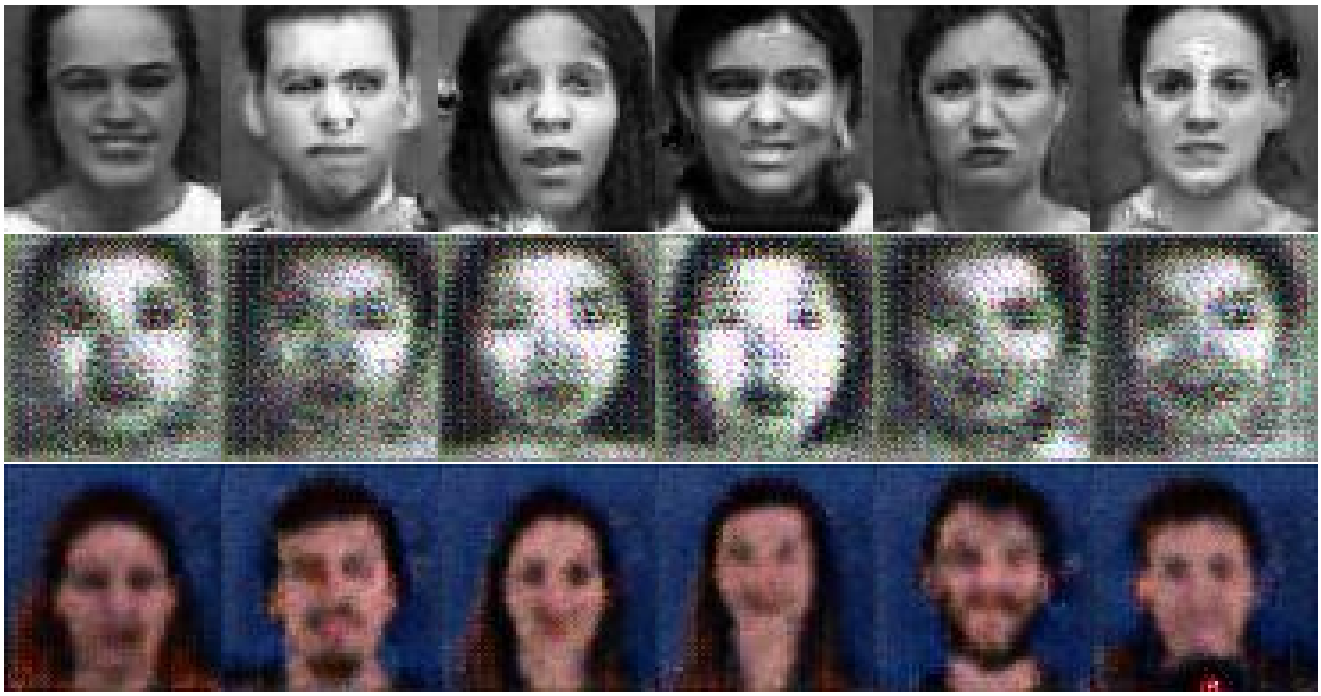


Figure 4.7: Example images generated by EmoGANs' generator using EmoGANs' objective function. Each row represents the generated samples from CK+, JAFFE, and MUG in sequential order.

facial characteristics. Compared to CK+, MUG is more homogeneous, including the Caucasian race. Although facial occlusion is significantly reduced, some male subjects in the MUG dataset have facial hair, such as beards or mustaches, which cover the upper lips, philtrum (the area between the upper lips and nostrils), and chin. Furthermore, the generated samples of MUG in Figure.4.7 (last row) demonstrate a resemblance to the characteristics present in the dataset.

4.1.2 Wasserstein Distance Based Emotion Generative Adversarial Networks (EmoGANs1)

This section proposed the EmoGANs model[92] with the Wasserstein distance cost function to overcome the training instability. Before the proposed model, the problems in GANs training will be discussed.

4.1.2.1 Problems in GANs Training

Difficult to achieve convergent training GANs training is built on finding the Nash equilibrium in a non-cooperative game, where each model wants to optimize its profits. The Nash equilibrium is hard to achieve in practice as concurrently updating their own cost function by gradient descent optimization does not guarantee to reach the equilibrium where both models obtain their optimal outcome[28]. Suppose that the objective function of a two-player game is xy, where a player controls the variable x and the other manages the variable y. For the first player, its respective cost function is xy, whereas, the other wants to minimize it that is $-xy$. Therefore, the gradient for their respective cost becomes

$$\frac{\partial(xy)}{\partial x} = y, \ \frac{\partial(-xy)}{\partial y} = -x \tag{4.9}$$

where the objective function for the first player is xy and the other is $-xy$.

During optimization by gradient decent with a small learning step η, their corresponding updates become

$$\hat{x} = x - \eta.y, \;\; \hat{y} = y + \eta.x \tag{4.10}$$

where $\hat{x}$ and $\hat{y}$ are the updated values. η is the learning step for going down toward the minimum value.

Based on their gradient decent updates, x and y have different signs, causing oscillation and instability rather than converging at the origin, where $x = y = 0$.

Training instability The real data distribution could exist in a low-dimensional manifold embedded in the high-dimension space during manifold learning [96]. The generated data distribution also lies in a low-dimensional manifold, where the random numbers from latent space which has a small number of dimensions hardly fill up the pixels in image space. Therefore, when both real data distribution and generated distribution lie in low-dimensional manifolds, the supports of the distributions are disjoint [97]. When they have disjoint supports, the discriminator is able to distinguish the real and generated samples from those distributions and becomes perfect. Therefore, the discriminator could not provide meaningful gradients back to the generator, leading the unreliable training.

Jensen Shannon (JS) divergence is not continuous In GANs' training, the distance between real data distribution and generated distribution is measured by Jensen Shannon (JS) divergence. In some circumstances, the JS divergence is not continuous. Suppose that there are two distributions P_0 and P_θ having the corresponding value of 0 and θ in the 1^{st} dimension and follow the Gaussian distribution $\mathcal{N}(0, 1)$ in the 2^{nd} dimension as in Figure.4.8.

Assume that P_0 is the real distribution that we wanted to approximate and P_θ is the prior distribution. The parameters of P_θ are learned to approximate the real distribution P_0, meaning that θ will be moved closer to zero. During approximation, the vanilla GANs[27] use the JS divergence to measure how close these two distributions are which is defined as follows.

$$\text{JS}(P_0, P_\theta) = \frac{1}{2}\text{KL}(P_0, P_m) + \frac{1}{2}\text{KL}(P_\theta, P_m) \tag{4.11}$$

where P_m is the mixture of the two distributions $(P_0,\ P_\theta)$ which is defined as $\frac{1}{2}(P_0) + \frac{1}{2}(P_\theta)$. KL represents the Kullback-Leibler (KL) divergence defined in the following equation.

$$\text{KL}(P_0, P_\theta) = \int_{x,y} \log\Big(\frac{P_0(x,y)}{P_\theta(x,y)}\Big) P_0(x,y)dy\, dx \tag{4.12}$$

From the KL divergence equation, it is asymmetric. If $P_\theta(x, y) = 0$ and $P_0(x, y) > 0$ at any point of (x, y), KL divergence becomes $+\infty$. Similarly, the same value holds for $\text{KL}(P_\theta, P_0)$. Therefore, it becomes

$$\text{KL}(P_0, P_\theta) = \text{KL}(P_\theta, P_0) = \begin{cases} +\infty & \text{for } \theta \neq 0 \\ 0 & \text{for } \theta = 0 \end{cases} \tag{4.13}$$

where θ is the two-dimensional parameter space such as x and y.

To define the JS divergence between them, the mixture distribution P_m is

$$\text{KL}(P_0, P_m) = \int_{x,y} \log\Big(\frac{P_0(x,y)}{P_m(x,y)}\Big) P_0(x,y)dy\, dx = \log 2 \text{ where } (x, y) \neq 0 \tag{4.14}$$

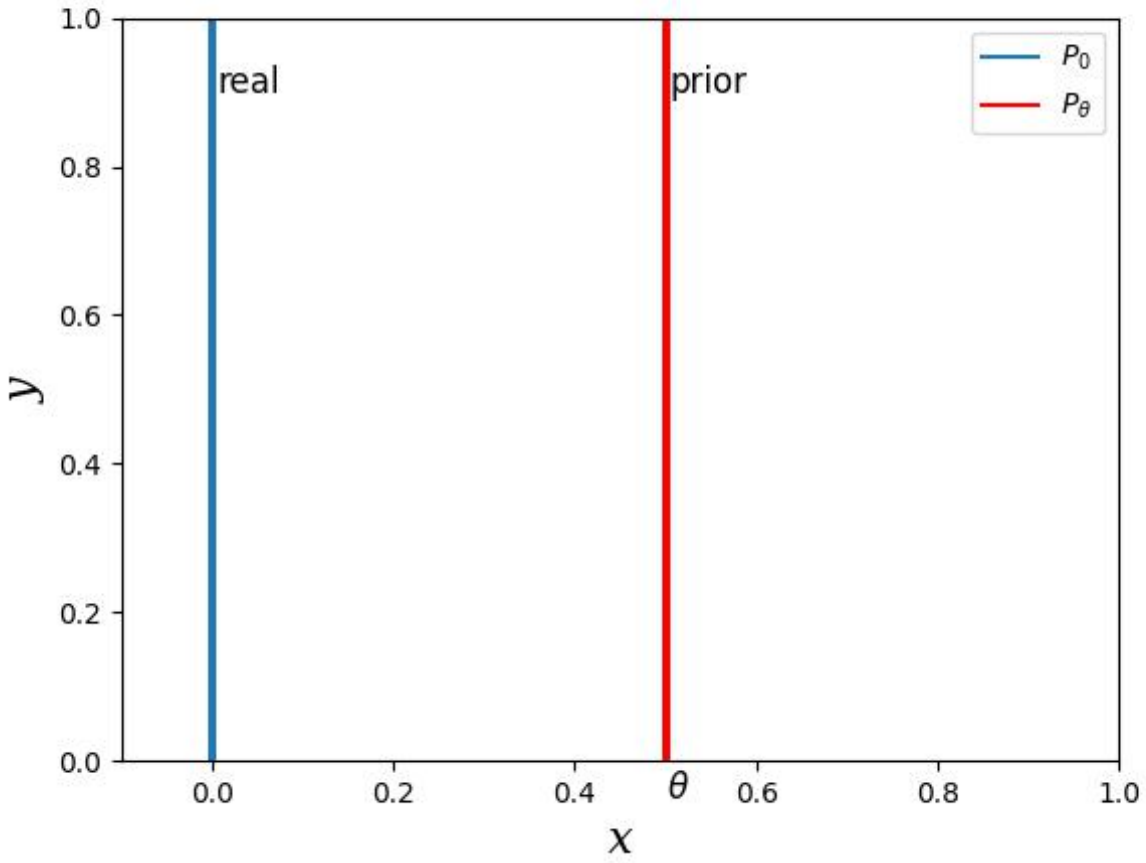


Figure 4.8: An example case where Jensen Shannon (JS) divergence is not continuous. P_0 is the distribution such that $(0, \mathcal{N}(0,1))$ and P_θ is the prior distribution such that $(\theta, \mathcal{N}(0,1))$. θ is the parameter of the distribution.

Similar condition holds for $\mathrm{KL}(P_m, P_0)$. Therefore, JS divergence of P_0 and P_θ is

$$\mathrm{JS}(P_0, P_\theta) = \begin{cases} \frac{1}{2}\log 2 + \frac{1}{2}\log 2 = \log 2 & \text{for } \theta \neq 0 \\ 0 & \text{for } \theta = 0 \end{cases} \tag{4.15}$$

From the above equation, we can observe that the JS divergence jumps to zero from the value of log2 when θ becomes zero and can be concluded that it is not continuous.

4.1.2.2 Wasserstein Distance or Earth Mover Distance

Wasserstein distance defines the probability distribution by moving the transported mass at each point with minimal effort during the transformation from the distribution P_r into P_g. Effort can be defined as the product of transported mass m and distance γ from moving the point from x to y. Each γ can be considered an optimal transportation plan defined as infimum over (P_r, P_g). To execute the transportation plan γ, the mass m need to be moved from the point x to y. Therefore, the amount of mass m leaving from the point x becomes $\int_y \gamma(x,y)dy$ and the amount of mass m entering the point y is $\int_x \gamma(x,y)dx$, which leads to the effort as follows.

$$\mathcal{W}(P_r, P_g) = \int_x \int_y \gamma(x,y)||x-y||dy\,dx = \mathbb{E}_{(x,y)\sim\gamma}[||x-y||] \tag{4.16}$$

where P_r and P_g are the real distribution and generated distribution, respectively. $\gamma(x,y)$ defines the transportation plan from point x to y whose ends up with the corresponding marginals of P_r and P_g.

From the example where JS divergence is discontinuous (Figure.4.8), the Wasserstein distance between P_0 and P_θ is just moving the mass from the point (θ, y) to the point $(0,y)$ along the straight line. Therefore, it can be defined as follows.

$$\mathcal{W}(P_0, P_\theta) = \begin{cases} |\theta| & \text{for } \theta \neq 0 \\ 0 & \text{for } \theta = 0 \end{cases} \tag{4.17}$$

where θ represents the parameters x and y where $x, y \in \mathbb{R}^2$. From this equation, we can observe that the Wasserstein distance is continuous and useful to calculate the meaningful gradients during optimization[38].

4.1.2.3 Network Configurations of EmoGANs1

From the previous discussion, we can observe that the Wasserstein distance has advantages over JS divergence in estimating probability distribution. Therefore, the EmoGANs model[91] is modified based on this Wasserstein distance. The proposed models can be seen in Figures.4.9 and 4.10.

The overview configuration of the generator G can be viewed in Figures.4.9. Similar to EmoGANs, it takes three inputs such as two images that represent different basic emotions and a prior latent sample. Their respective facial expressions features are filtered through convolutional kernels and passed through rectified linear unit activation, which are later concatenated with latent samples to form the baseline features in generating the image with a mixture of emotions. Its loss can be defined as follows.

$$\mathcal{L}_G = -\mathbb{E}_{\mathbf{z}\sim P_Z}[D(G(\mathbf{x}_{c1}, \mathbf{x}_{c2}, \mathbf{z}))] \tag{4.18}$$

where $\mathbf{x}_{c1}$ and $\mathbf{x}_{c2}$ represent the images with different facial expressions for basic emotions. $\mathbf{z}$ is the latent samples from the prior distribution P_Z.

With respect to the configuration of the discriminator D, we can observe that the deeper convolution layer extracts generic features which are more useful for the final classification than those in earlier convolution in Figure.3.7. Therefore, more convolutional layers are added in D compared to previous EmoGANs. Besides, linear activation is used in the final layer of D instead of sigmoid as in previous EmoGANs as the task of D is calculating the Wasserstein distance d between real data distribution and generated distribution. Its loss can be defined as follows.

$$\mathcal{L}_D = \mathbb{E}_{\mathbf{x}\sim P_X}[D(\mathbf{x})] - \mathbb{E}_{\mathbf{z}\sim P_Z}[D(G(\mathbf{x}_{c1}, \mathbf{x}_{c2}, \mathbf{z}))] \tag{4.19}$$

where $\mathbf{x}$ is the morphed sample that is used as the training data. $\mathbf{x}_{c1}$ and $\mathbf{x}_{c2}$ represent the images with different facial expressions for basic emotions. $\mathbf{z}$ is the latent samples from the prior distribution P_Z. $\mathcal{L}_D$ provides the score for the distance between the training distribution and the generated distribution.

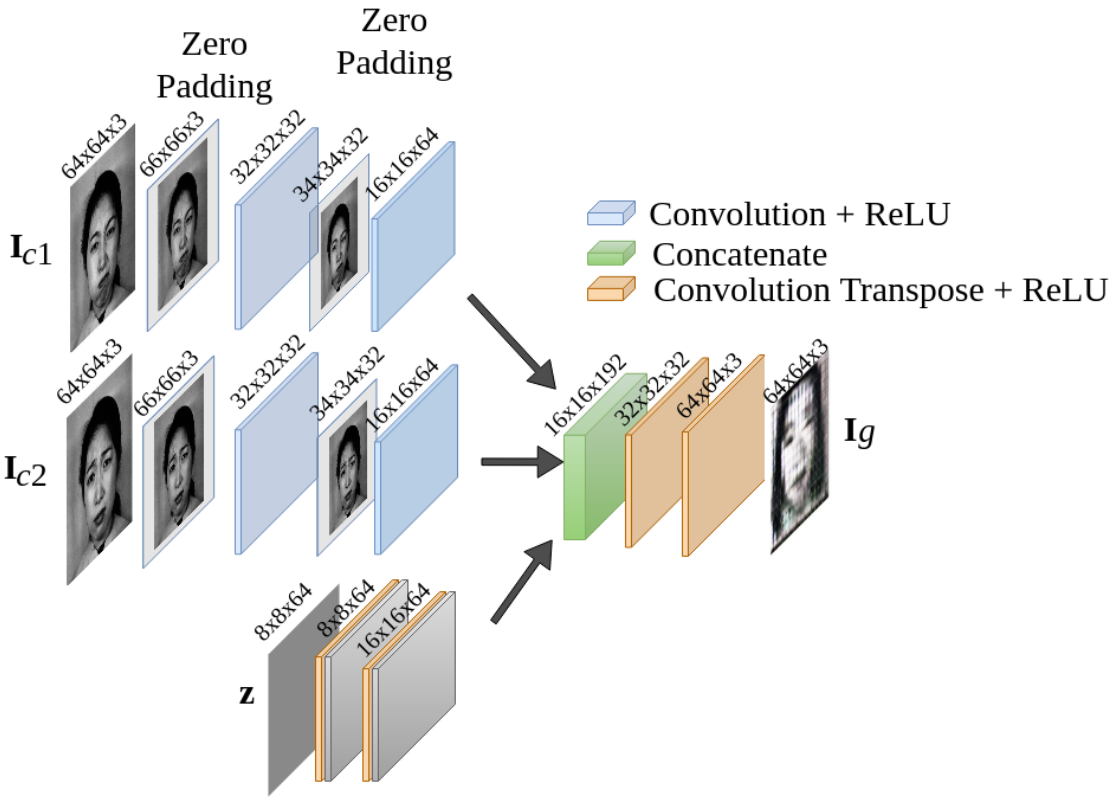


Figure 4.9: Overview of EmoGANs1' generator

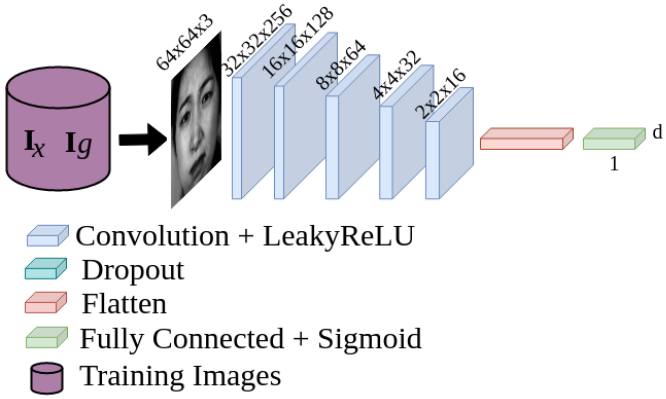


Figure 4.10: Overview of EmoGANs1' discriminator

4.1.2.4 Network Training

In the study [38], Wasserstein Generative Adversarial Networks (WGANs) work well with Root Mean Squared Propagation (RMSProp) gradient descent optimization proposed by Geoffrey Hinton. Therefore, the proposed model is trained with RMSProp optimization, which chooses the different learning rates for each parameter during the update by computing the average of the partial gradient to dampen the gradient oscillations. For each parameter, the update rule of RMSProp is defined as follows.

$$\mathbf{v}_t = \beta \mathbf{v}_{t-1} + (1 - \beta) g_t^2 \tag{4.20}$$

where $\mathbf{v}_t$ is the decaying average of the square of the partial gradient for the current iteration. $\mathbf{v}_{t-1}$ is also the same but it was from the previous iteration. g_t^2 is the squared of the partial gradient. β is a hyper-parameter.

Then, the custom step size is defined based on this average value as follows.

$$\triangle \boldsymbol{\theta}_t = \frac{\lambda}{\sqrt{\mathbf{v}_t + \epsilon}} g_t \tag{4.21}$$

where λ is the initial learning rate and $\triangle \boldsymbol{\theta}_t$ is the current step size which is reduced by the decayed average value. ϵ is the small constant to prevent the division error. g_t is the partial gradient for the current iteration t.

The weight update for the next iteration is as follows.

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \triangle \boldsymbol{\theta}_t \tag{4.22}$$

Similar to previous EmoGANs training, RMSprop is executed with a mini-batch stochastic approach with size 64 for 100 iterations. The initial learning rate is set the value of 5e-05. The values of the hyperparameter are set at the default value offered by the TensorFlow library[98].

4.1.2.5 Discussion

Similar to EmoGANs, the current model, EmoGANs1, also operates on a 64×64 image space, and the training data is normalized before training. The model takes latent samples from a Gaussian distribution with zero mean and unit variance. Following the suggestion in the study by Arjovsky et. al [38], the model is optimized using RMSProp with updates performed over mini-batches of size 64. Additionally, the weights of the discriminator are clipped to the range of [-0.01, 0.01]. The learning rate is set to 5e-05 for both sub-models.

Unlike EmoGANs, the discriminator in EmoGANs1 executes more updates compared to the generator, depending on the specified value of the parameter, which is set to 10 in our experiment. These settings remain consistent across all datasets. Based on the experimental training results, the model for CK+ is trained for 5000 training iterations since the final model produces good images compared to the intermediate models. However, for the other two datasets, the models are trained for the same number of iterations, but the intermediate results are superior to the final ones. The selection of models for further processing is based on their visual quality during training. To evaluate the quality of the generated images, we provide random latent vectors from the prior distribution and random image pairs to the trained generator, projecting them onto a 64x64 image space. The results are depicted in Figure 4.11.

It can be observed that EmoGANs1 convergences faster than EmoGANs due to being trained with a higher learning rate. As discussed in the previous section, this suggests that the Wasserstein-based objective allows for continuous and usable gradients for the weights updates, at the point where vanilla GANs cannot support. Compared to the images generated by EmoGANs, the images produced by EmoGANs1 are less noisy, especially with JAFFE dataset. The synthesized images display facial resemblances to the corresponding dataset, indicating that the generated distribution approaches closer to the ground truth distribution.

Based on the illustrated Figure 4.11, the image generation for JAFFE produces the best-synthesized images compared to the other two datasets. This is attributed to JAFFE having a lower diversity of facial characteristics and a simpler distribution due to the subjects sharing the same gender and race. However, the facial expressions are less visible in the generated images, especially for MUG and CK+ as they contain diverse and heterogenous facial characteristics and their underlying distribution is more complex than JAFFE.

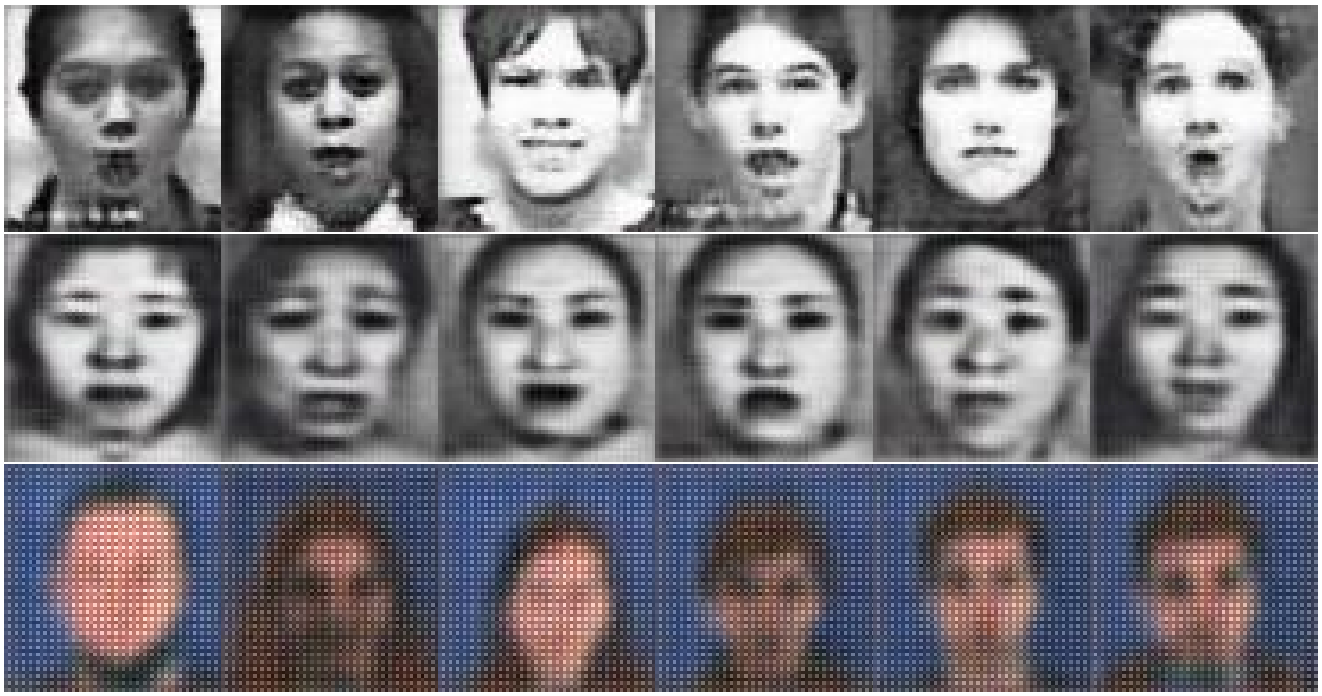


Figure 4.11: Example images generated by EmoGANs1' generator using Wasserstein objective function. Each row represents the generated samples from CK+, JAFFE, and MUG in sequential order.

Furthermore, the resolution of the generated images is low, making it difficult to assess the facial expressions in the synthesized images produced by both EmoGANs and EmoGANs1. The trade-off between image resolution and training stability is the challenging factor of GANs and their variants. In both EmoGANs models, the generator incorporates additional data, such as image pairs, to construct the baseline features that represent a mixture of facial expressions. During the trial experiments, we observed that the discriminator is capable of classifying the given samples after a few training iterations, requiring a slower learning process by reducing the learning rate and allowing sufficient time for the generator to update itself.

Compared to the discriminator, the generator in both of EmoGANs models has the dual tasks of extracting the facial expression features and projection these features onto image space. Consequently, training instability is undeniable during the training of EmoGANs. To address this issue, the dual tasks of the generator are separated into different models. The updated version of the EmoGANs model will be discussed in the next section.

4.1.3 Methodology for Generating Mixture of Facial Expressions Images for Complex Emotions (EmoGANs2)

This section proposes a new framework for generating complex facial expressions images to represent the mixture of emotions[93]. The overview framework is depicted in Figure.4.12. It is composed of three main parts: feature extraction, feature conversion, and image generation. In this framework, Generator G's tasks are divided and performed separately (feature extraction and image generation), whereas, in the previous models, they are performed within the same model. The conversion process in the middle transforms the features into latent space. Each part is explained in sequential order.

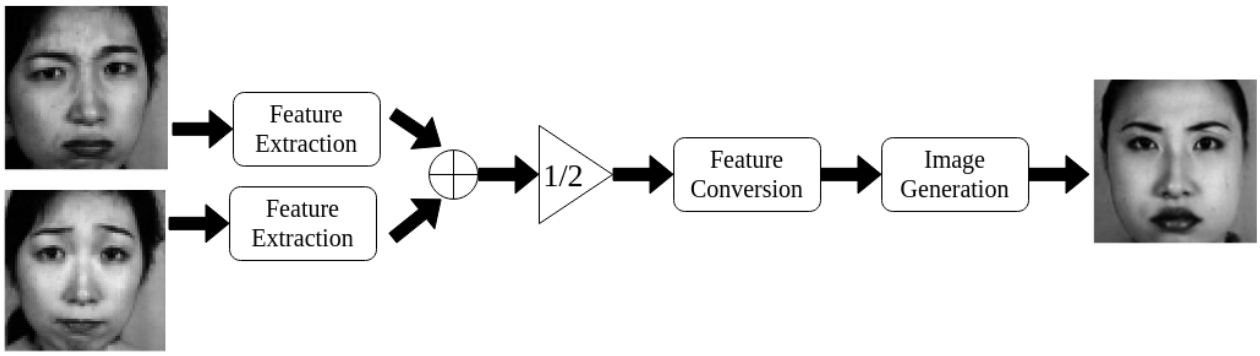


Figure 4.12: Overview of EmoGANs2 framework

4.1.3.1 Feature Extraction

As we discussed in the previous section, convolution neural network (CNN) has advantages over fully connected layers for processing high dimensional data such as images. In this research, we use the simple CNN with three convolution layers that are trained to recognize the basic emotions, and pre-trained CNN is employed as a feature extractor.

Convolution neural network (CNN) is first proposed by LeCun et.al[99]. It is a type of neural network that uses linear operation, named convolution, in its network layers to process grid-like data such as time-series data, and images. Convolution operation improves the machine learning system by having the properties such as sparse interactions, parameter sharing, and equivariant translations [28].

Properties of Convolution Neural Networks (CNN)

Sparse interactions Traditional neural networks, such as FNN, use matrix multiplication, as shown in Equation 4.2, to describe the interaction between neurons of input and output layers. This implies that every neuron is involved in computing each output, making it computationally expensive. In contrast, CNN has the property of sparse interactions, which can be achieved by creating smaller kernels than the input size (See in Figure.4.13). When dealing with high-dimensional data, such as images with thousands of pixels, a convolution kernel can detect small and useful features such as edges by moving through the image. This typically converts the thousand of raw pixels into fewer pixels by extracting the essential features, which reduces memory usage and improves efficiency. In the example, nine-pixel values are reduced to one new meaningful pixel.

Parameter sharing Parameter sharing means using the same set of parameters in more than one place in a model, also known as tied weights. This is because the weights applied to one input are tied or shared with other inputs. In FNN, once the weights are used in computing the output, they are not revisited. However, in CNN, the same set of weights can be applied in different places in the input. For example, the same values of a convolution kernel (highlighted with a blue box in Figure.4.13) will be applied to all input pixels by moving from left to right and top to bottom to get the feature maps of the whole input.

This property reduces the memory requirement to store the model's parameters compared to the traditional network. In the example, the input image is the 8x8 wide grayscale image and the output is the 6x6 image after convolution. The transformation requires 6x6x3 or 108

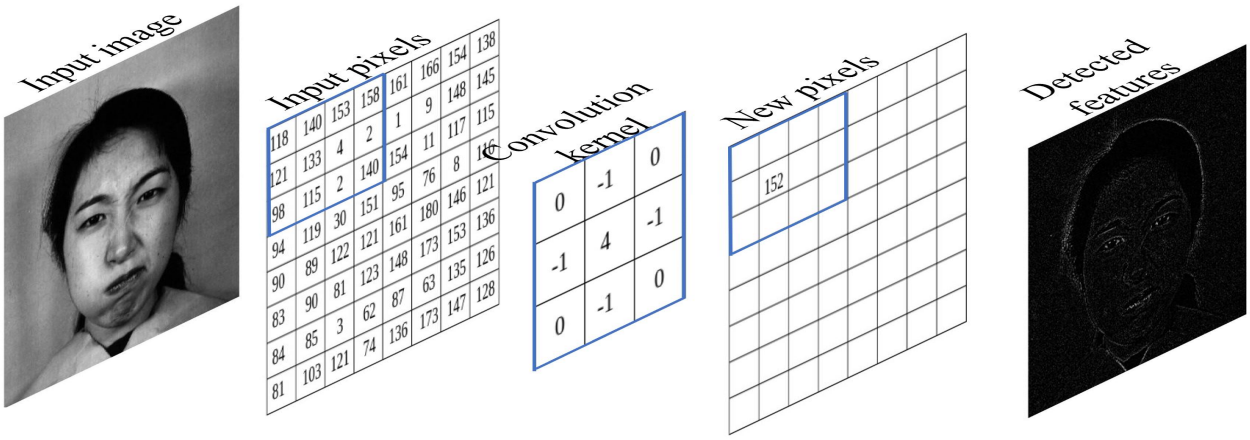


Figure 4.13: An example of sparse interactions or connectivity in convolution neural network. The input image is resized into 8x8x1 and the edge filter is used as a convolution kernel for illustration purposes.

floating-point operations. 3 refer to two addition and one multiplication to perform convolution. In a traditional network, it requires 8x8x6x6 or 2304 operations for matrix multiplication. Therefore, convolution improves network efficiency.

Translation equivariance Due to its sharing of the parameters across inputs, it induces another property, named translation equivariance. Theoretically, a function f is equivalent to a function g if it satisfies the following condition [28].

$$f(g(\mathbf{x})) = g(f(\mathbf{x})) \tag{4.23}$$

where f and g are equivariant.

Suppose that a function f is a convolution function with edge kernel applied on an input image and function g is the function to shift the 25 pixels to the right and 50 pixels down. We can observe that performing convolution after translation yields the same result as performing convolution before translation in Figure.4.14. It implies that if an input changes, the result also changes. It is helpful to detect an object in an image regardless of its position in the image.

Convolution Neural Network Configurations A general convolution layer of a convolution neural network involves three stages: convolution stage, detector stage, and pooling stage [28]. In the convolution stage, several linear operations that are convolution, are performed on the input and pass through the non-linear activation function, generally rectified linear activation (ReLU) in the detector stage. In the last stage, a pooling function is executed to summarize the values of feature maps according to their neighborhood pixel values.

For example, the max pooling function introduced by the study [100] selects the maximum value within the neighborhood window. This induces an additional property called translation invariant, as it uses the summary of neighborhood pixels and resists small translation changes in that neighborhood. Network configuration varies on the objectives of its usage.

In this research, the convolution neural network is constructed by applying three typical convolution layers. The convolutional feature maps are reshaped to flatten out and fully connected before recognizing six basic emotions: anger, disgust, fear, happiness, sadness, and surprise. The complete configuration is illustrated in Figure.4.15. Since the input data are simple frontal facial expressions, the current configuration of CNN is sufficient to achieve the

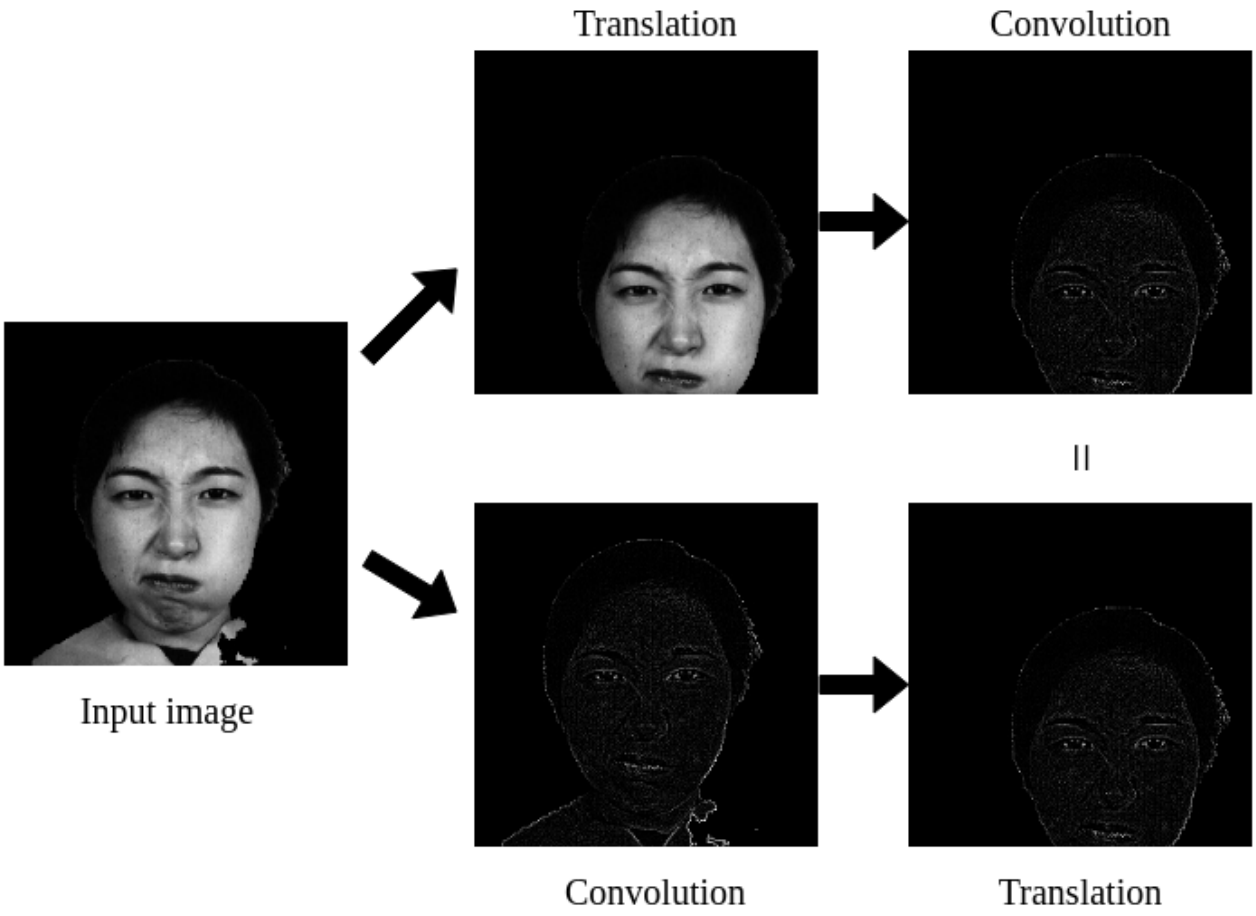


Figure 4.14: An example of translation equivariance in convolution neural network.

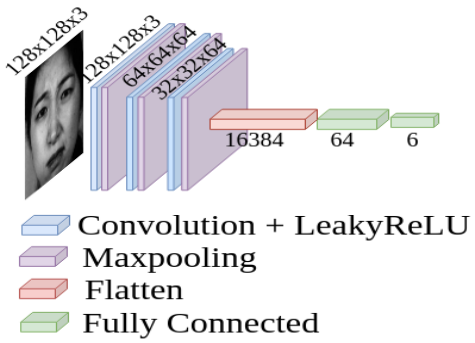


Figure 4.15: Network configurations of Convolution Neural Network (CNN) for basic emotion recognition

accuracy of about 80% in CK+, JAFFE, and 92% in MUG (Table.4.1). The training progress of CNN can be seen in Figure.4.16. The trained model is evaluated for each emotion and major samples from the test set are correctly classified in Figure.4.17.

Table 4.1: Accuracy of Convolution Neural Network (CNN) in 10 fold cross validation set for basic emotion recognition

Accuracy a			
Fold No.	Datasets		
	CK+	JAFFE	MUG
1	87.1%	94.7%	95.7%
2	80.6%	100.0%	91.4
3	83.9%	84.2%	94.6%
4	80.6%	100%	96.8%
5	90.3%	77.8%	96.7%
6	90.3%	83.3%	91.3%
7	80.6%	72.2%	90.2%
8	66.7%	88.9%	90.2%
9	66.7%	94.4%	90.2%
10	83.3%	72.2%	89.6%
Average accuracy ($\hat{a}$)	81.0%	86.8%	92.6%

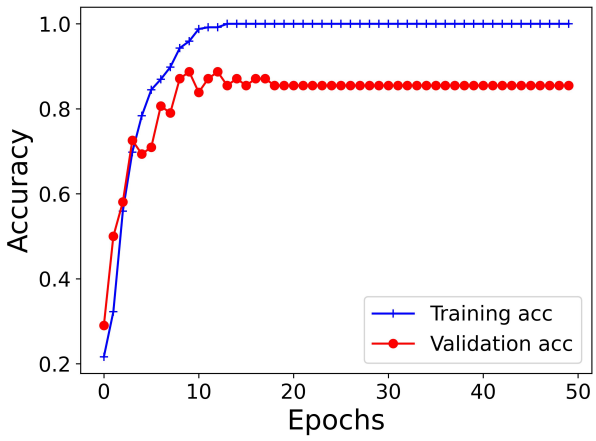
4.1.3.2 Image Generation

In the previously proposed models, constructing the features that represent the mixture of emotions, and up-sampling them for image generation is built within the same model G. It overloads G's task and makes the training unstable where D performs easy assessment on its input distribution. Besides, it also encountered the mode collapse problem during training, in which the generator cannot synthesize images that have many variations in them. To overcome it, the previous models require a slower step size in their optimization, and make the training slow, even for low-resolution 64x64 images. Therefore, the construction of baseline facial expression features for the mixture of emotions and mapping them onto image space are separated in this new framework.

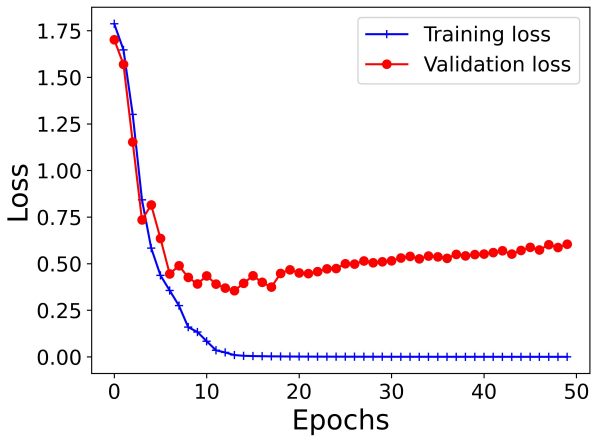
In this new framework, the generator model G aims to synthesize the high-resolution images from given features, that are easier to assess the facial expressions in them. To achieve this goal, the Progressive Growing of Generative Adversarial Networks (PGGANs) proposed by Karras et.al[36], is adopted for image generation. Characteristics of PGGANs are going to be discussed subsequently.

Progressive Growing of Generative Adversarial Networks (PGGANs) Progressive Growing of Generative Adversarial Networks (PGGANs) [36] is designed to synthesize high-resolution images up to 1024 pixels by progressively training the network. Vanilla GANs [27] has a trade-off between image resolution and training stability, because the discriminator model D can easily reject the generated images apart from the training images, especially when the high resolutions require detailed features to be filled in the generated images and lead to exaggerating the training gradients. Moreover, training for higher resolutions also requires large memory usage. Therefore, PGGANs assemble their sub-models G and D progressively in their configurations.

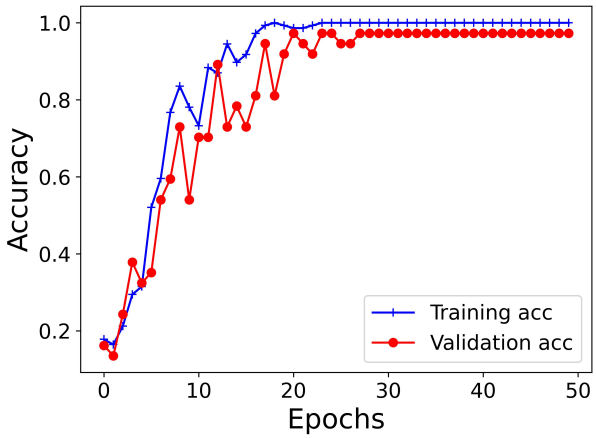
PGGANs started with 4x4 low-resolution images and set both sub-models G and D to correspond to each other. After training the initial models with 800k training images, they expand the networks by gradually adding new layers onto their initial networks. The way to add the new layers for higher resolution in PGGANs is shown in Figure.4.18. It shows the



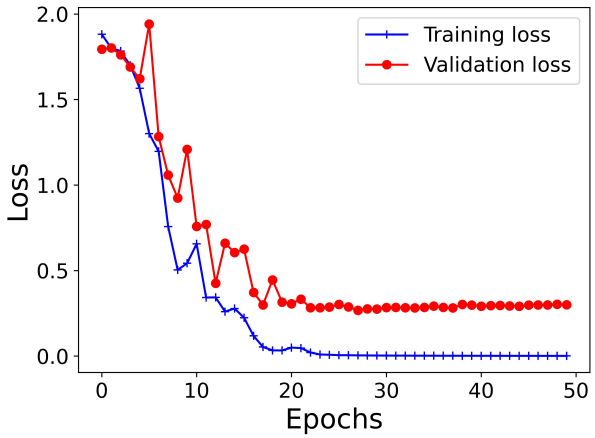
(a) CK accuracy plot



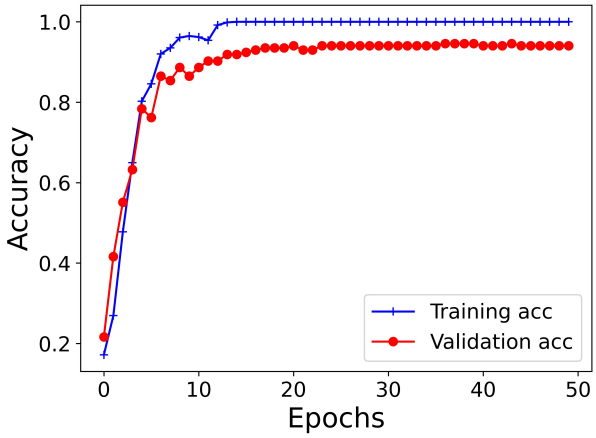
(b) CK loss plot



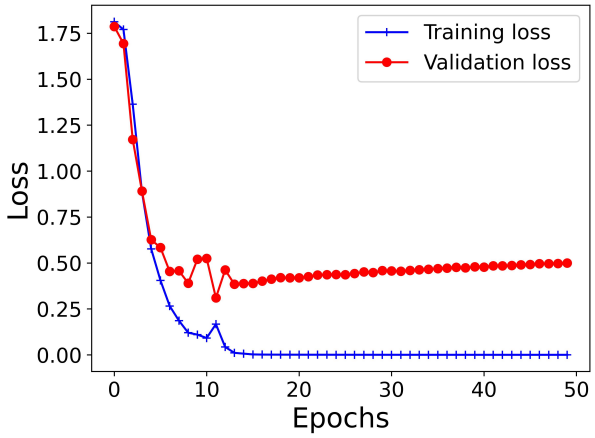
(c) JAFFE accuracy plot



(d) JAFFE loss plot



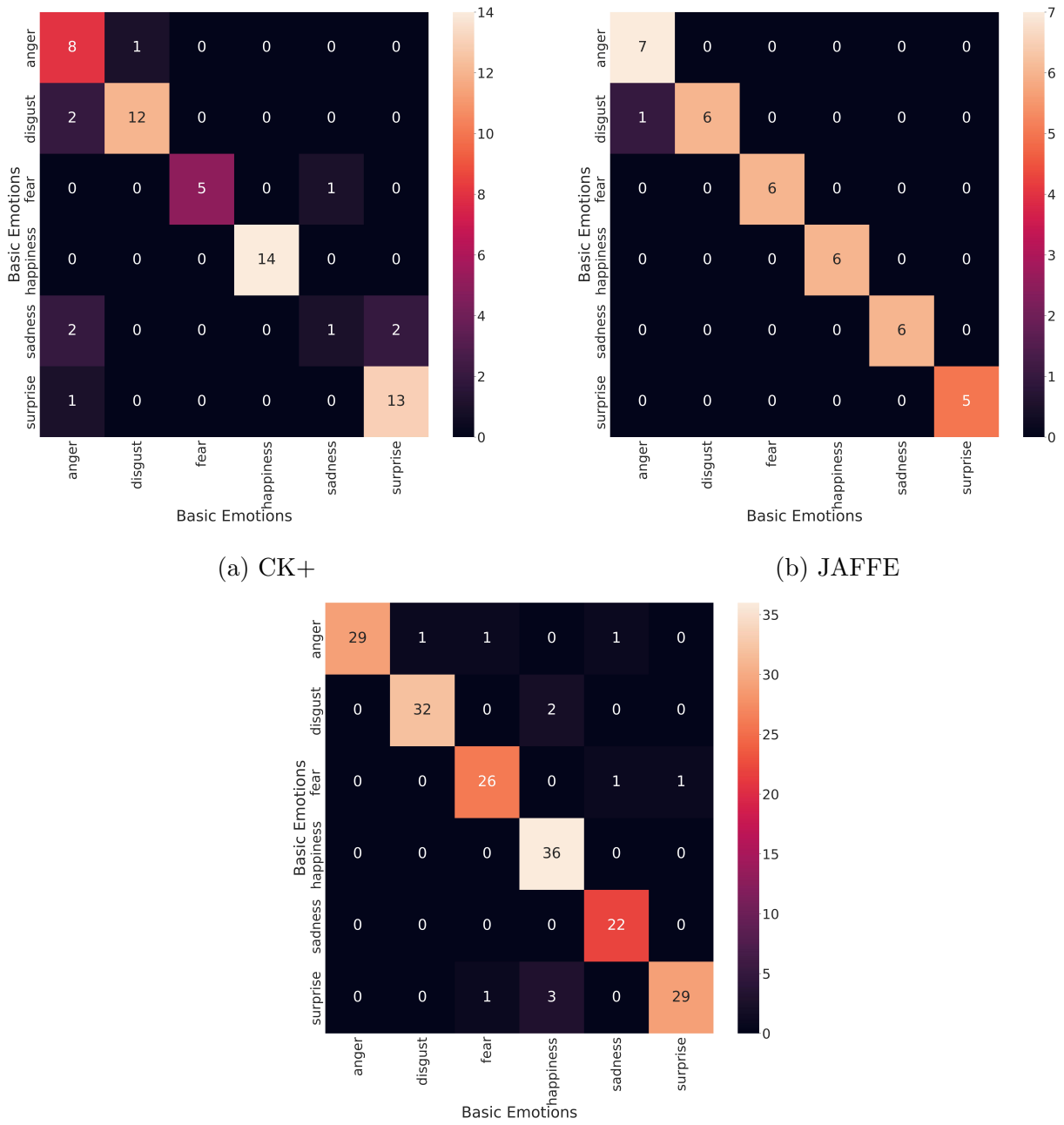
(e) MUG accuracy plot



(f) MUG loss plot

Figure 4.16: Plots for training a convolutional neural network (CNN) with Adam stochastic gradient descent optimization for every 16 batches. The learning rate is set to 1e-3.

layer transition from 16x16 resolution to 32x32 images. The initial network is trained first to establish stable weights in Figure. 4.18(a) with one convolution layer having a 1x1 convolution kernel in each sub-model. Then, the new convolution layer is added by increasing the resolution



(a) CK+

(b) JAFFE

(c) MUG

Figure 4.17: Performance of convolutional neural network (confusion matrix over a test set)

twice by the nearest neighbor filtering in G and halving the resolution by average pooling in D in Figure. 4.18(b). Both the new and the existing layers are smoothed out linearly using the parameter α, and then trained until they reach a stable state in Figure. 4.18(c). This gradual training technique provides two key benefits: training stability and the ability to generate high-resolution images.

Moreover, PGGANs also added counter measurements to handle the GANs' mode collapse problem, in which the generator G always generates a small subset of the output distribution believing this set can deceive the discriminator D. Theoretically, when the mode collapse problem is about to happen, the gradients of D go towards the same direction for the similar samples, resulting in directing all output towards a single point where D considers it as the real

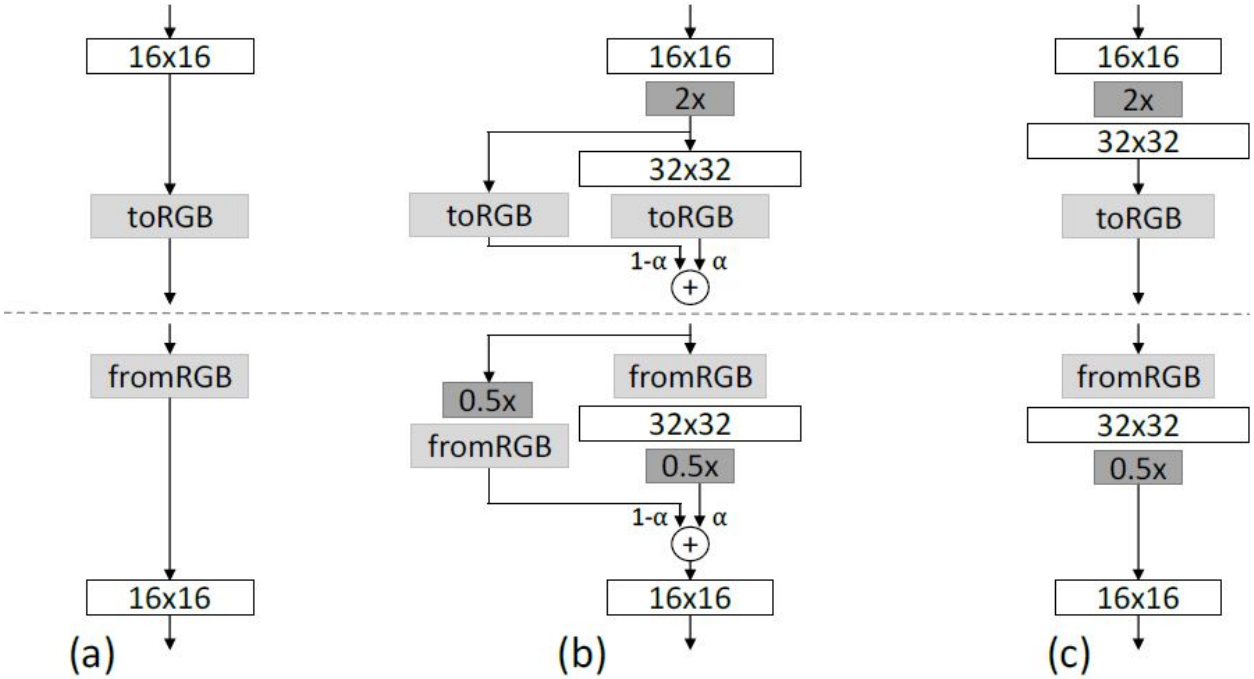


Figure 4.18: An example of adding new layers onto the initial models G and D in Progressive Growing GANs [36]. (a) is the initial model for 4x4 resolution. (b) is the transition phase for adding the new layers to the initial models. (c) is the finalized model for 32x32 resolution.

distribution [101]. Therefore, it cannot provide useful feedback to G, which keeps giving similar latent samples while believing that they can deceive D. The strategy to avoid this problem is to show D more dissimilar examples. In PGGANs, mini-batch standard deviation is used, in which the constant feature map computed from the mean and standard deviation of each feature in each spatial location in feature maps is added to D, resulting in encouraging G to produce dissimilar images with large variations in their pixel values.

Since both sub-models are competing against each other, GANs' training is widely known as unstable by escalating the magnitude of their gradients, which causes a longer time to converge, often resulting in poor-quality images. The other research employs one-sided label smoothing[101] or variants of batch normalization techniques [94] such as virtual batch normalization[101], weight normalization[102], and layer normalization[103] to deescalate the magnitudes and support stable training. Unlike the above-mentioned techniques, PGGANs normalize the weights at each layer by dividing them with a layer constant derived from He's initializer [104] during run time. This ensures keeping the weights at the same dynamic range and having the same learning speed at each layer. Since it maintains the same range, it makes independent of the scale of the weights and becomes efficient when weights are too big or too small. Another normalization used in PGGANs is pixel-wise normalization, which normalizes the values of each feature map to a unit length to avoid exploding gradients. It was applied after each convolution layer in the generator.

Loss function In PGGANs, Wasserstein distanced-based adversarial loss with gradient penalty (WGANs-GP)[39] is employed, where gradient norm is used to enforce the Lipschitz continuity. The regularization term can be added as follows.

$$\mathcal{L}_{\text{GP}} = \lambda \mathbb{E}_{\hat{\mathbf{x}} \sim P_{\hat{\mathbf{x}}}} [(\|\nabla_{\hat{\mathbf{x}}} D(\hat{\mathbf{x}})\|_2 - 1)^2] \tag{4.24}$$

where λ is the coefficient of gradient norm. $\hat{\mathbf{x}}$ is the sample from the distribution $P_{\hat{\mathbf{x}}}$ obtained by uniformly sampling from P_X and P_Z, which is described as in the following equation.

$$\hat{\mathbf{x}} = \epsilon \mathbf{x} + (1 - \epsilon) G(\mathbf{z}), \epsilon \sim U[0, 1] \tag{4.25}$$

where ϵ is a random number from the uniform distribution which ranges from 0 to 1, inclusive. $\mathbf{x}$ is the sample from the real distribution P_X. $\mathbf{z}$ is the latent sample. $G(\mathbf{z})$ is the sample from the generated distribution.

Besides it, PGGANs also adds another regularization to D's loss, named as drifting term, to keep D's output to stay close to zero.

$$\mathcal{L}_{\text{Drift}} = \epsilon_{\text{Drift}} \mathbb{E}_{\mathbf{x} \sim P_X} [D(\mathbf{x})^2] \tag{4.26}$$

where drifting constant ϵ is set to a small value of 0.001.

By combining them, the loss of D becomes

$$\mathcal{L}_D = \Big(\mathbb{E}_{\mathbf{z} \sim P_Z} [D(G(\mathbf{z}))] - \mathbb{E}_{\mathbf{x} \sim P_X} [D(\mathbf{x})] \Big) + \mathcal{L}_{\text{GP}} + \mathcal{L}_{\text{Drift}} \tag{4.27}$$

where the first loss refers to the adversarial loss defined in WGANs.

For the generator loss, it can be defined as a single adversarial loss as follows.

$$\mathcal{L}_G = \mathbb{E}_{\mathbf{z} \sim P_Z} - D(G(\mathbf{z})) \tag{4.28}$$

4.1.3.3 Feature Conversion

The third component of the new framework for complex facial expressions images is the ordinary least-squared linear regressions model which is responsible to transform the baseline features into latent features, which are the input of the generator G. Linear regression is the supervised learning algorithm to model the relationship between the response variable $\mathbf{z}$ and one or more explanatory variables $\mathbf{v}$.

Suppose that there are N instances of baseline features to represent mixtures of emotions as explanatory variables. Each instance has its response variable. It can be defined as

$$\{\mathbf{z}, \mathbf{v}\}_{n=1}^{N}, n \leq N \tag{4.29}$$

where $\mathbf{v}_n$ is the explanatory variable which has j attribute for n^{th} instance, which can be defined as $[v_n^{(1)}, v_n^{(2)}, ..., v_n^{(j)}]^{\mathrm{T}}$. $\mathbf{z}_n$ is the response variable which has k attribute for n^{th} instance which is represented as $[z_n^{(1)}, z_n^{(2)}, ..., z_n^{(k)}]^{\mathrm{T}}$.

The linear regression between $\mathbf{z}$ and $\mathbf{v}$ can be modeled for n^{th} instance in the following equation.

$$z_n^k = \psi^{(1)} v_n^{(1)} + \psi^{(2)} v_n^{(2)} + ... + \psi^{(j)} v_n^{(j)} + \varepsilon_n^k \tag{4.30}$$

where j is the attribute of the feature vector to represent baseline facial expressions features for the mixture of emotions. k is the attribute of latent vector $\mathbf{z}$. ψ is the regression coefficient. ε is the error for n^{th} instance.

It can be represented in vector form as follows.

$$\mathbf{z}_n = \mathbf{v}_n^{\mathrm{T}} . \boldsymbol{\psi} + \boldsymbol{\varepsilon}_n \tag{4.31}$$

where $\mathbf{z}$ and $\boldsymbol{\varepsilon}$ represent 1 vectors for the n^{th} instance. $\mathbf{v}$ is $1\times j$ feature vectors and $\boldsymbol{\psi}$ is the $j \times k$ coefficient matrix for the regression model.

The goal of the regression model is to find the values of coefficients that produce the minimum error in the following equation.

$$\boldsymbol{\varepsilon} = \mathbf{z} - \mathbf{v}^{\mathrm{T}}.\boldsymbol{\psi} \tag{4.32}$$

Therefore, the objective function of the model is to find the optimal coefficient in $\boldsymbol{\psi}$ by minimizing the squared of the residual error as follows.

$$\min_{\boldsymbol{\psi}} \sum_{n=1}^{N} (\mathbf{z}_n - \mathbf{v}_n^{\mathrm{T}}.\boldsymbol{\psi})^2 \tag{4.33}$$

Evaluation of linear regression model Previously latent features are estimated based on the baseline facial expressions for the mixture of complex emotions by the linear regression model. To validate that the linear regression model can transform the features into latent space, cosine similarity among features before and after conversion is measured. The flow of the evaluation framework is depicted in Figure.4.19.

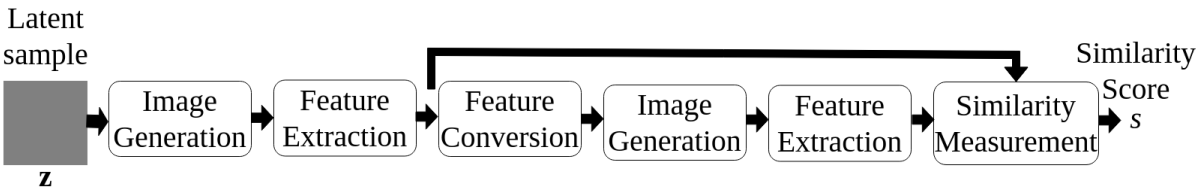


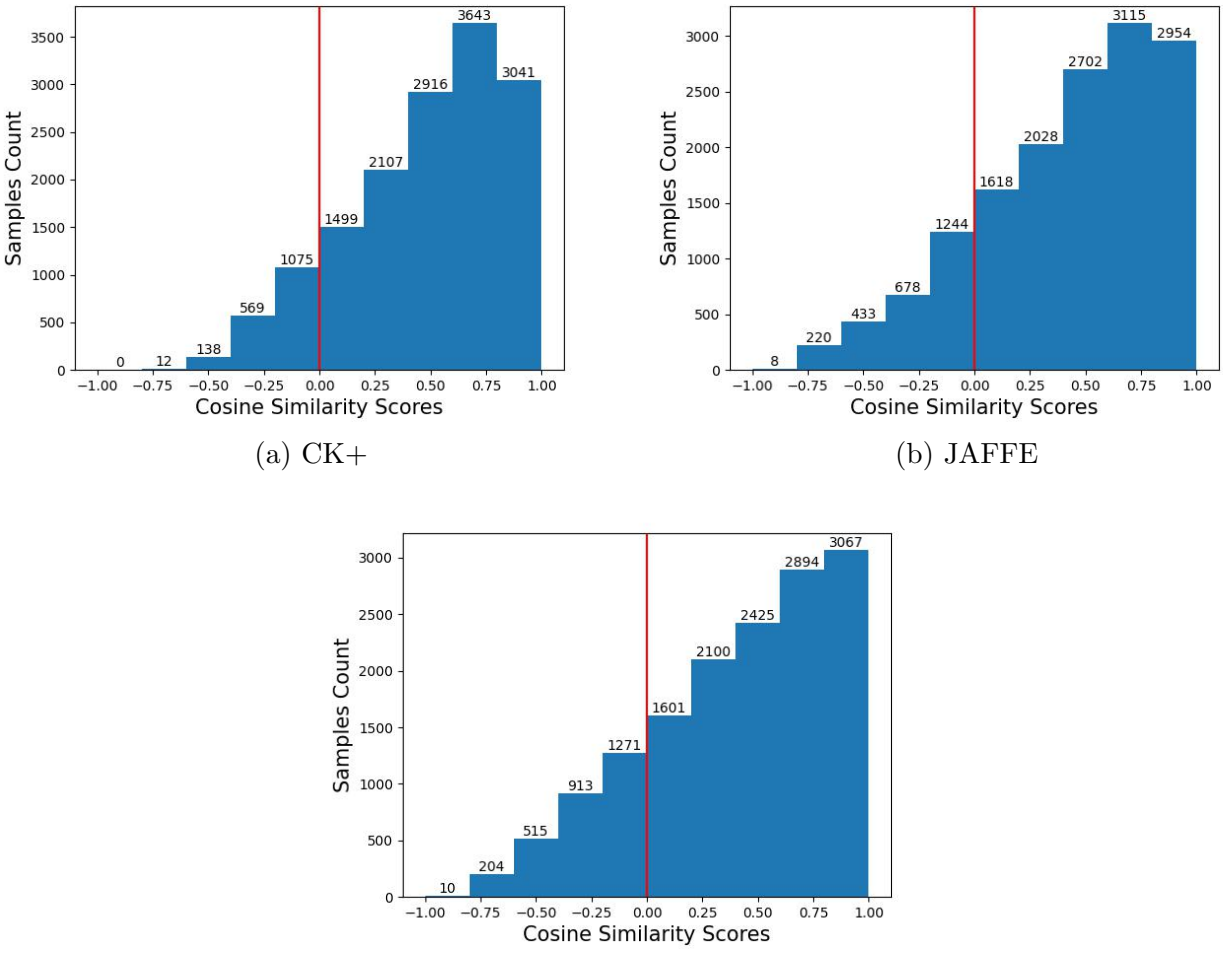
Figure 4.19: Evaluation framework to validate the converted features

First, the random latent sampled from the Gaussian distribution $\mathcal{N}(0, 1)$ is given to the generator G, which is trained with the facial expression datasets. The synthesized images are taken by their respective recognition model for feature extraction. Those features are used to model the regression model to estimate the latent samples. The same generator is used to produce images using the estimated latent. Again, facial expression-related features are extracted from those images. At last, the similarity (Equation. 3.16) score is computed on those two features set to evaluate if the regression model can convert the features into latent space.

The evaluation framework is executed on a test set randomly sampled from the Gaussian distribution, containing 15k samples. The output of similarity measurement on test samples distribution is shown in Figure.4.20, where we can observe that the majority of test samples demonstrated the similarity in features space. For visual interpretation, the images synthesized on initial latent and estimated latent by the regression model are also displayed in Figure.4.21.

4.1.3.4 Discussion

The generated images synthesized by the previous models, EmoGANs and EmoGANs1, are noisy, and the facial expressions in the images are unclear. Additionally, model training is also unstable, as the discriminator completes its updates in a few training epochs, resulting in the inability to provide useful gradients to the generator for improving the generated distribution. Furthermore, the generator performed additional tasks, including facial expression feature extraction, to assemble the baseline features for the mixture of facial expressions and image generation.



(a) CK+

(b) JAFFE

(c) MUG

Figure 4.20: Distribution of cosine similarity scores measured among features from images synthesized by G on the initial latent sampled from $\mathcal{N}(0, 1)$ and the latent estimated by the regression model. The scores range from +1 (the most similarity) to -1 (the most dissimilarity).

Therefore, feature extraction and image generation have been separated into different models: a facial expressions recognition model and Progressive GANs. We trained a regression model by minimizing the squared loss in Equation.4.33 to transform the facial expression features into latent that can be used by the generator. Figure 4.12 is executed to synthesize the mixture of facial expressions, and the results are shown in Figure 4.22.

Compared to the generated images produced by the previous EmoGAN models, the synthesized images generated using the current methodology are not noisy. The facial expressions in these images can be easily assessed by the naked eye. It is evident that the generated images consist of a mixture of facial expressions. When comparing them to CK+, the facial expressions in the generated images from JAFFE are more subtle. Analyzing the results from Figure 4.22 for CK+ and MUG, their facial expressions are blatant.

We also included results tested with the same inputs for EmoGANs and EmoGANs1 for all datasets. An example result for the mixture of happiness and surprise is shown in Figure.4.23. The results for other mixed emotions can be seen in the Appendix section B.1. It can be seen that the EmoGANs2 does not maintain the facial attributes from the input as the identity in generated images is changed. Both the previous models and the current model do not

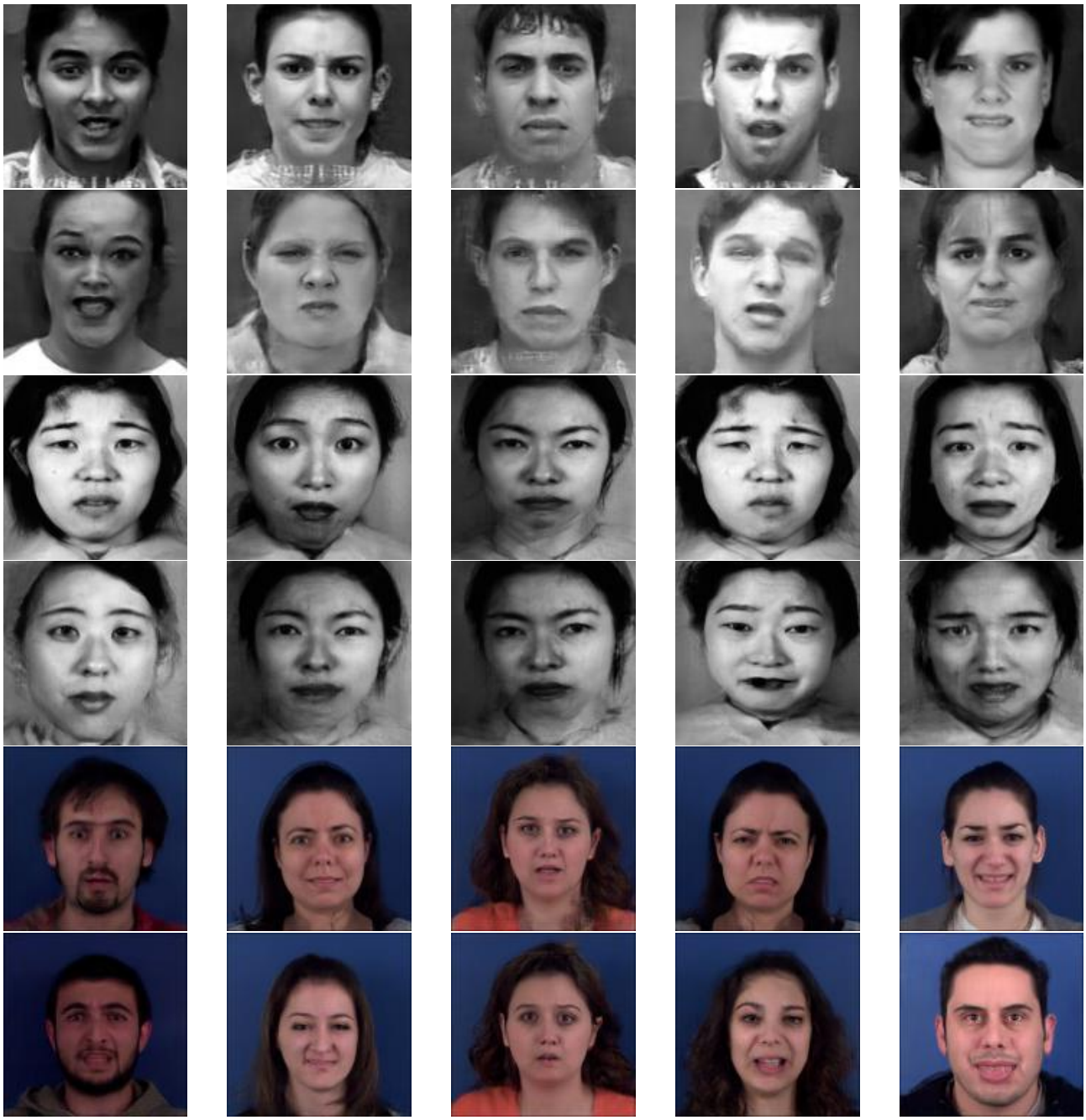


Figure 4.21: Samples of generated images by G on initial random latent and estimated latent by the regression model. 1^{st}, 3^{rd} and 5^{th} rows represent the images synthesized from the initial latent vectors sampled from the Gaussian distribution $\mathcal{N}(0, 1)$. 2^{nd}, 4^{th}, and 6^{th} rows indicate the images synthesized from the latent estimated by the regression model. The first two rows are the result of CK+, the middle two rows are the result of JAFFE, and the rest are for MUG.

preserve the facial representation from the input pair. This means that even though the images are paired based on identity in JAFFE, the identity in the generated images is altered. The alteration occurs because the previous modules before image generation in Figure 4.12 transform the combined features onto latent space. However, the organization of semantics for visual attributes in the latent space remains unknown. Additionally, the combined features are derived from facial expression features extracted by the expression model and do not include any information related to subject identity. As a result, the generated images exhibit changes

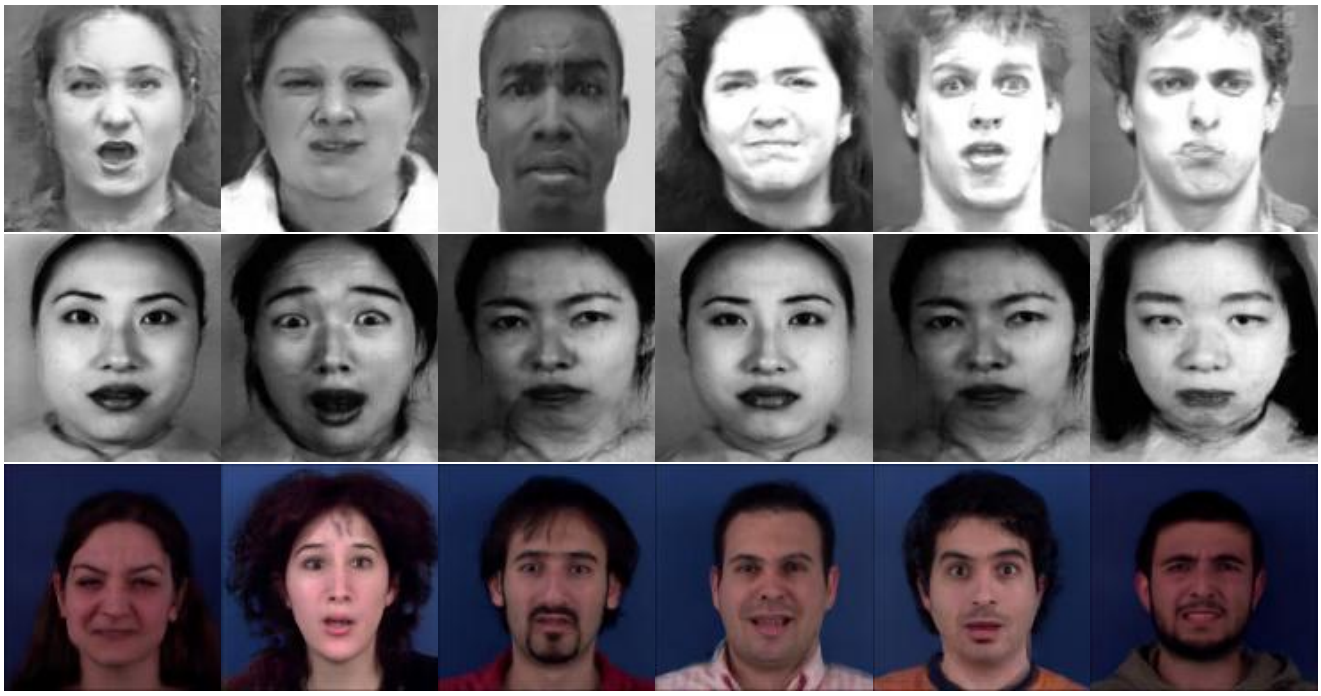


Figure 4.22: Example images generated by Progressie GANs' generator using the latent transformed from the average facial expressions features to represent the mixture of emotions. Each row represents the generated samples from CK+ and JAFFE in sequential order.

in identity. In other words, the facial representation and facial expressions are entangled in the latent space. Consequently, in the next section, we will propose a new methodology to disentangle these features and exert control over the image generation process.

4.2 Supervised Training

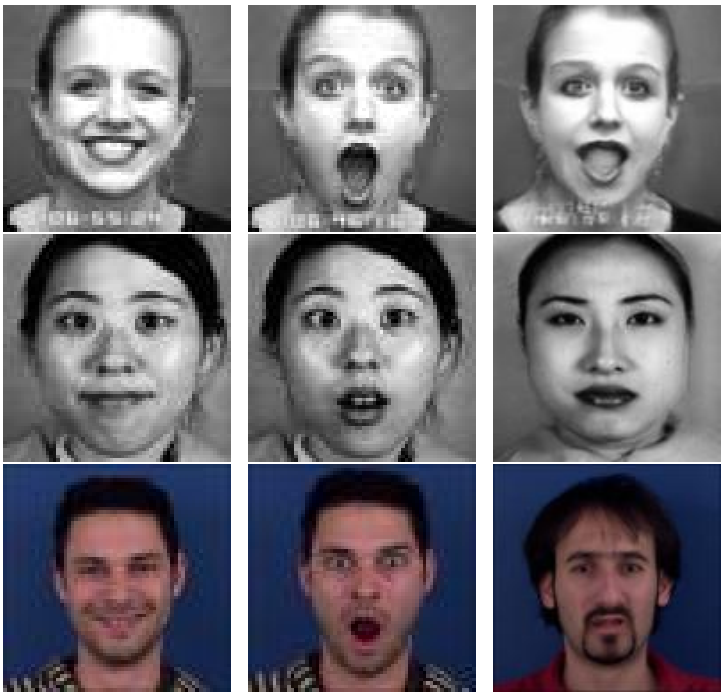
4.2.1 Conditional Emotion Generative Adversarial Networks with Identity Preservation (EmoGANs3)

In the previous EmoGANs models[91, 92, 93], the latent space of the generator G did not consider the facial expression features and subject identity information separately, therefore, those features are intertwined in the latent space. As a result, we could not control the output of image generation by G. To control image generation result on the class of complex facial expression and subject identity in the synthesized images, we proposed the new framework illustrated in Figure.4.24, where subject information and expression classes are separately encoded and given to conditional generation model . Each component will be sequentially discussed.

4.2.1.1 Encoding the label information for a mixture of emotions

In the datasets, the label for each basic emotion is defined in a categorical representation such as 'anger', 'disgust', 'fear', 'happiness', 'sadness' and 'surprise'. Categorical data might have the ordered information about which class comes first in the labels. However, in this research, the ordered relationship among classes is trivial and categorical labels are required to encode them in the numerical format before training the neural network.

A common approach to encode the categorical labels into numerical format is one hot



(a) (b) (c)

Figure 4.23: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'happiness'*** emotion, and c_j refers to the ***'surprise'*** class. Column (c) is images synthesized by the generator of the EmoGANs2 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). Results from other classes can be found in the Appendix section B.1.

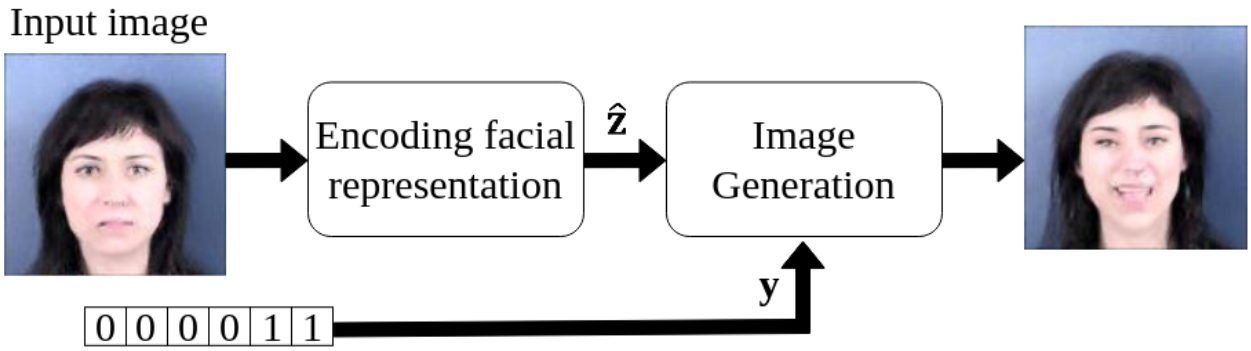


Figure 4.24: Overview of EmoGANs3 framework. $\mathbf{y}$ is the label vector for the class of mixture of emotions, which is encoded in six-dimensional space. Each dimension chronologically refers to the class of anger, disgust, fear, happiness, sadness, and surprise. $\hat{\mathbf{z}}$ is the encoded feature vector for subject identity information.

encoding which transforms them into binary vectors. For example, categorical labels for emotion classes have six different values. Therefore, they can be transformed into a vector in six-dimensional space. For the mathematical expression, it can be defined as follows.

$$\mathbf{y}_i \in \mathbb{R}^6 \tag{4.34}$$

where $\mathbf{y}$ is the standard basis in the Euclidean space, including six distinct vectors. $\mathbf{y}_i$ denotes

the value 1 in i^{th} position and 0 in elsewhere. For the categorical value of 'anger', it can be encoded as $\mathbf{y}_1 = [1, 0, 0, 0, 0, 0]$.

As morphing images are used as the training data in this research, their labels are associated with a couple of categorical data. Similar to one hot encoding in six-dimensional Euclidean space, it can be encoded as follows.

$$\mathbf{y}_{ij} \in \mathbb{R}^6, i \neq j \tag{4.35}$$

where the value of 1 is filled in i^{th} and j^{th} positions in Euclidean space to represent the emotion classes c_i and c_j respectively. For example, if the training sample has the categorical labels of 'sadness' and 'surprise', they are encoded as $\mathbf{y}_{ij} = [0, 0, 0, 0, 1, 1]$ to represent c_i as 'sadness' and c_j as 'surprise' as in Figure.4.24.

4.2.1.2 Encoding the identity information into latent space

Given a trained generator, it can be formulated as the deterministic function G as follows.

$$G : Z \rightarrow X \tag{4.36}$$

where Z is the d-dimensional latent space of the Gaussian distribution $\mathcal{N}(0, 1)$. X is the image space.

Each latent sample possesses semantic information such as gender, subject identity, and facial expression. However, the arrangement of that information is unknown as there is no inverse mapping from image space to latent space. Therefore, an external neural network is served as an inference network to do the inverse mapping. Mathematically, it can be represented as follows.

$$E : X \rightarrow Z \tag{4.37}$$

where E is the mapping function that maps the sample from image space X onto latent space Z.

Network configurations of the inference network Figure.4.25 depicts the overview of the inference network E. Once the generator model G of the conditioned Generative Adversarial Networks (cGANs)[33] had been trained, the inference network is trained with the training data containing the pair of random latent sampled from the Gaussian distribution $\mathcal{N}(0, 1)$ and generated images synthesized from those latent samples.

The network contains two convolution layers with nonlinear activation such as rectified linear units. Batch normalization is used to deescalate the magnitude of the gradients. At last, linear activation is employed to follow the latent distribution in the final fully connected layer. The loss function of the inference network E is to minimize the mean absolute error between initial random latent and latent predicted by the network in Equation.4.38. Network optimization is done by Adam stochastic gradient descent[95] for the mini-batch size 64 with a learning rate 1e-05. After training, the trained network is employed to encode the face representations in the implementation of the framework proposed in Figure.4.24.

$$\mathcal{L}_E = ||\mathbf{z} - E(\mathbf{I}_g)|| \tag{4.38}$$

where $\mathbf{z}$ is the original latent sampled from the Gaussian distribution. $\mathbf{I}_g$ is the generated images synthesized from $\mathbf{z}$. E is the inference network that maps the sample from image space onto latent space.

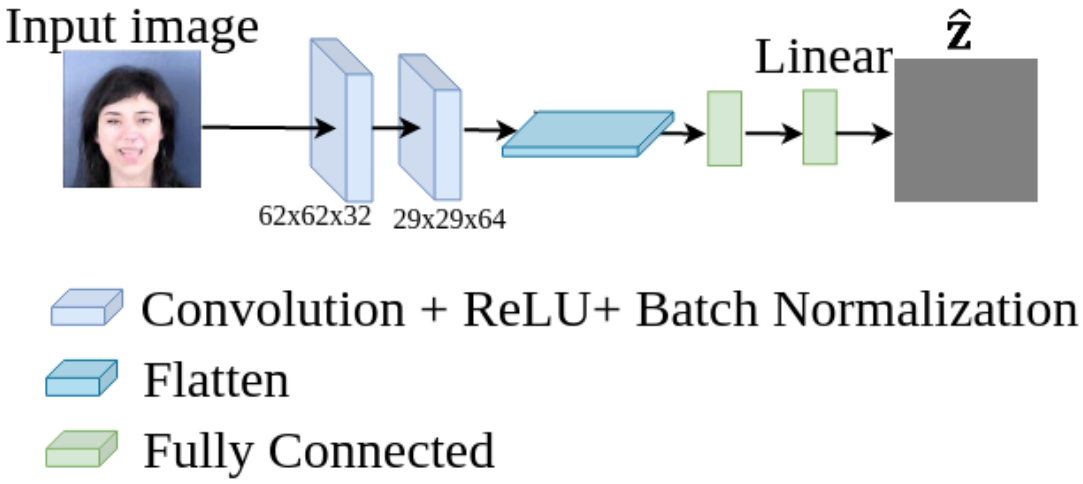


Figure 4.25: Network configuration of encoder model to encode the facial representation into latent space

4.2.1.3 Conditioned Generative Adversarial Networks

Conditioned Generative Adversarial Networks (cGANs) is proposed by Mirza et.al [33] to direct the data generation process with supplementary condition aside from prior latent. The additional condition can be class labels or different types of data, which are embodied with the respective input of each sub-model and taken by multi-layer perceptron layer in the corresponding network[33]. The loss function of cGANs can be defined as follows.

$$\min_{G}\max_{D} \mathrm{V}(D,\ G) = \mathbb{E}_{\mathbf{x}\sim P_X}[\log D(\mathbf{x}|\mathbf{y})] + \mathbb{E}_{\mathbf{z}\sim P_Z}[\log D(\ G(\mathbf{z}|\mathbf{y})] \tag{4.39}$$

where $\mathbf{z}$ is the latent sample from the Gaussian distribution with zero mean and unit variance for 100 dimensions. $\mathbf{x}$ is the sample drawn from the data distribution. $\mathbf{y}$ is the auxiliary input that is given as a condition of data generation.

Network configurations of cGANs In this research, cGANs are composed of convolution layers as in Figure.4.26. The benefits of using a convolution layer over a multi-layer perceptron or fully connected layer had been discussed in the previous section. Class labels are used as a condition, encoded in the two-hot encoding vector format as shown in the figure, and embedded in the fully connected layer of both sub-models. The corresponding input and encoded labels are then concatenated and up-sampled or down-sampled using a non-linear activation function, namely the leaky rectified linear unit.

Before training, the latent are initially sampled from the Gaussian distribution for 100 dimensions and labels for morphed images are encoded into a two hot vector in 6 dimensions. Training images are resized into 128×128. Weight updates are done by Adam optimizer [95] with mini-batch training for size 128. According to trial experiments, D learns faster than G. As a result, the slower learning rate is set for D with the value of 1e-05. For G, the learning rate is 1e-04. After training, only G is used to execute the new framework proposed in Figure.4.24 for image generation.

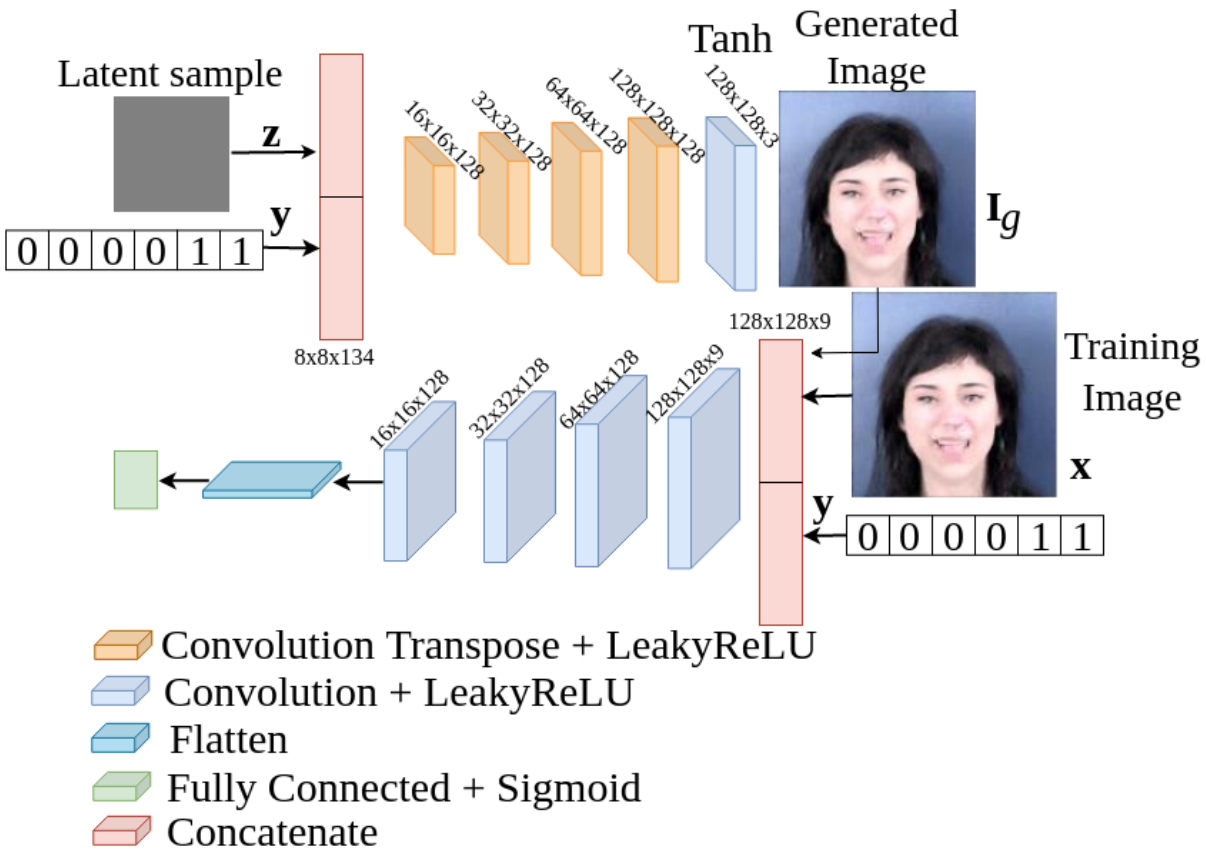


Figure 4.26: Network configuration of Conditioned Generative Adversarial Networks (cGANs)

4.2.1.4 Evaluation Methods

The new framework proposed in Figure.4.24 is evaluated on three perspective:

1. the ability to synthesize images of facial expressions that depict a mixture of emotions
2. subject identity preservation
3. disentanglement between features of facial expressions and subject identity information

Evaluation on the ability to synthesize images of facial expressions that depict a mixture of emotions In this research, we aim to synthesize the facial expressions that represent the mixture of emotions. Facial expression can be associated and coded by individual facial muscles, named Action Units (AUs), which is proposed by Psychologist Ekman[5]. For example, a standard facial expression to express happiness is a smile, which pulls up the muscles at the cheek and lips corner. In the Facial Action Coding System (FACs)[5], the associated muscles are defined as AU06 (cheek raiser), AU12(lip corner puller), and AU25(lip part) for a standard smile in expressing happiness. Similarly, the other basic emotions can be defined as in Table.1.1.

In the research [14], the presence of a mixture of facial expressions is detected by observing the activated AUs for each basic facial expression. Suppose that the set of AUs for happiness is $\mathbb{S}_{c_i}$, in which c_i refers to the 'happiness' class. Similarly, the set of AUs for surprise is $\mathbb{S}_{c_j}$, in which c_j refers to the 'surprise' class. The set of AUs for the mixture of 'happiness' and 'surprise' can be defined as follows.

$$\mathbb{S}_{c_i} \cup \mathbb{S}_{c_j} = \{y : y \in \mathbb{S}_{c_i} \text{ or } y \in \mathbb{S}_{c_j}, i \neq j\} \tag{4.40}$$

where y is the element of the action unit that belongs to the AUs set of each basic emotion c_i and c_j.

For example, the AUs for happiness emotion are AU06, AU12, and AU25. The AUs for surprise emotion are AU01, AU02, AU05, AU25, and AU26. Therefore the candidate AUs for the mixture of happiness and surprise is the combination of these two sets such as AU01, AU02, AU05, AU06, AU12, AU25, and AU26. The library with an MIT license, named Py-Feat[16] is employed to detect the candidate AUs to evaluate the presence of a mixture of facial expressions in the generated images. Py-Feat contains two pre-trained models for action unit detection that used feature representation extracted by Histogram of Oriented Gradient (HOG) and employed shallow classifiers such as linear Support Vector Machine (SVM) and ensemble learning. In this research, we use the model with SVM to detect AUs.

Evaluation on identity preservation To objectively evaluate the identity preservation in the generated images, we designed the framework illustrated in Figure.4.27. First, random latent is a sample from the distribution $\mathcal{N}(0, 1)$ and desired labels for the mixture of emotions. In the figure, the example label **y** represents the mixture of 'sadness' and 'surprise' classes.

Both prior latent and encoded labels are taken by the pre-trained generator G to synthesize the image, which is again taken by the pre-trained inference network E that does reverse mapping in Equation 4.37 to encode the identity of the included subject onto latent space. After encoding, the same generator is used to map the encoded latent into an image. The pre-trained face model [87] is employed to extract the facial representation from a couple of generated images. Similarity (Equation.3.16) among that representation is computed for the objective score.

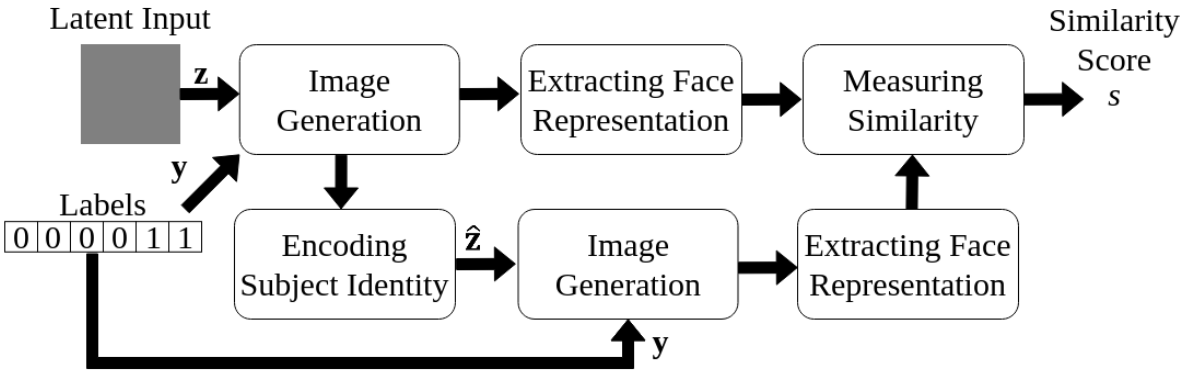


Figure 4.27: Evaluation framework to assess the property of subject identity preservation

Evaluation on feature disentanglement Feature disentanglement indicates that facial representation and emotion class representation have separate influences on the image generation process. Therefore, we illustrate the framework for evaluating the disentanglement property in Figure 4.28. In the initial setting, two instances of a latent variable and their corresponding labels are randomly sampled and mapped onto image space by the pre-trained generator G. The labels are then exchanged, and the respective images are generated using the same G and identical latent samples. Since the label representations are changed, we hypothesize that the facial expressions in the generated images will be changed while keeping the same subjects.

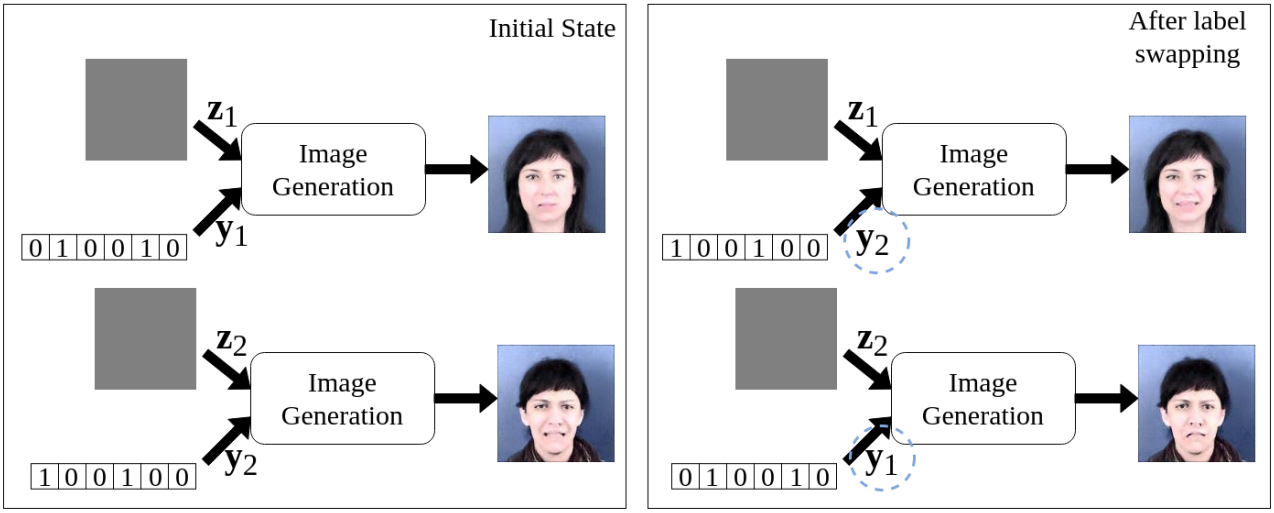


Figure 4.28: Evaluation framework to assess the property of disentanglement between facial expressions and facial representations

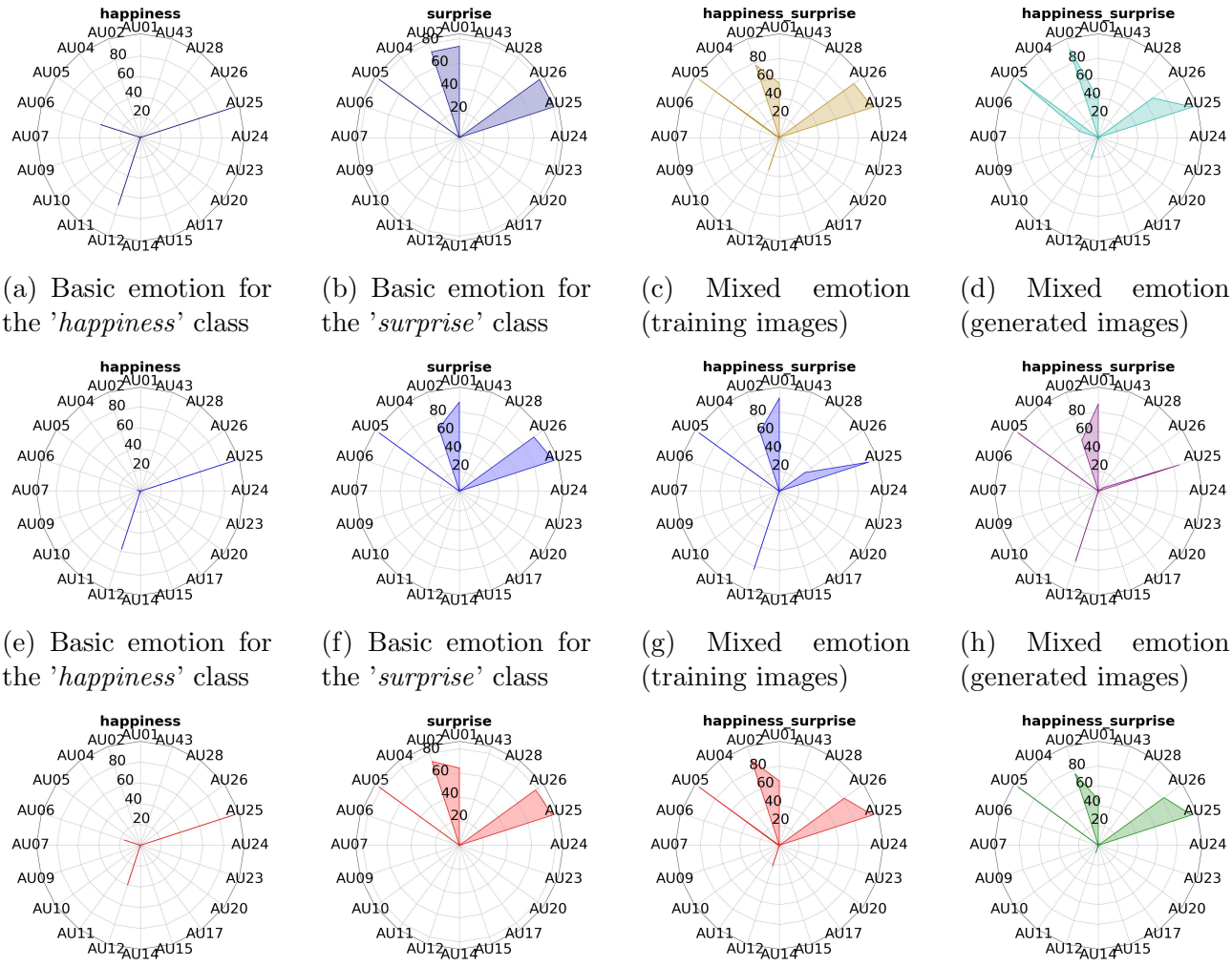
4.2.1.5 Discussion

As stated in the previous section, the EmoGANs3 model is evaluated based on three viewpoints such as the ability to construct the image representing the mixtures of facial expressions, identity preservation in the generated images and features disentanglement.

Figure.4.29 illustrates the output of the example classes, such as 'happiness' and 'surprise', for JAFFE and MUG datasets. The prototypical Action Units (AUs) are shown with the unit of the number of images in which those AUs are activated, represented as a percentage. For example, the standard AUs for happiness expressions are AU6, AU12, and AU25, while the standard AUs for surprise are AU1, AU2, AU5, AU25, and AU26. The standard AUs for the mixture of these two classes become the union combination of these two sets.

The ground truth images, as well as morphed images used during training, exhibit the least activation or inactivation for AU6 and AU26. Approximately 60% of the ground truth images have AU2, while over 90% of the images display other AUs (Refer to Figure.4.29(c)). When compared to ground truth images, the generated images show activated prototypical AUs that are similar in shape and quantity. These similarities can be observed in Figure.4.29(c and d). During the creation of morphed images, the facial expressions of the given image pair are equally blended. Therefore, the morphed images represent a mixture of facial expressions. The resemblance of the generated images to these morphed images suggests that they also contain a mixture of facial expressions corresponding to their respective emotions. This same interpretation applies to other classes. Additional graphic illustrations for other mixtures of emotions can be found in the Appendix section B.2, providing further support for our observations.

From these graphic results, we can also observe another characteristic: cultural differences in expressing facial expressions for the same emotions. Three standard datasets, CK, JAFFE, and MUG, are used to evaluate EmoGANs3, and they consist of subjects from different nationalities, such as American, Japanese, and Caucasian. As observed in Figure 4.29, the activation of AUs for the same expressions differs. For instance, a greater number of images from MUG and CK+ display AU2, which involves raising the outer eyebrows to express surprise, compared to JAFFE. Raising the eyebrows is typically associated with expressing surprise. We can interpret that Americans and Caucasians tend to exhibit more extravagant expressions, while the Japanese



(a) Basic emotion for the '*happiness*' class

(b) Basic emotion for the '*surprise*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

(e) Basic emotion for the '*happiness*' class

(f) Basic emotion for the '*surprise*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)

(i) Basic emotion for the '*happiness*' class

(j) Basic emotion for the '*surprise*' class

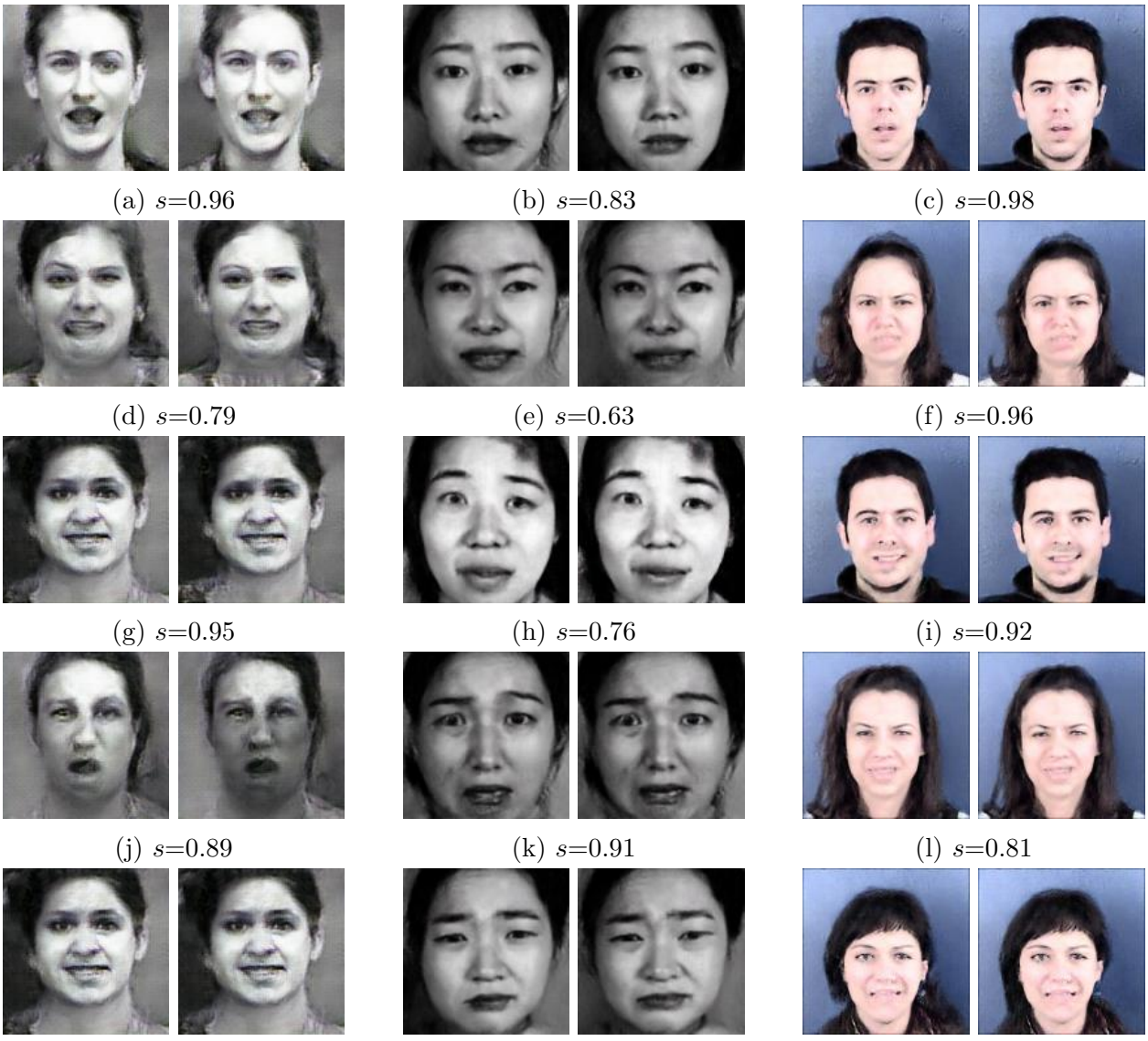
(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

Figure 4.29: Prototypical action units (AUs) for basic emotions ('***happiness***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training. Full results for the other mixed emotions can be found in the Appendix section B.2.

prefer subtler expressions when conveying emotions.

To assess the identity preservation in the generated images, the evaluation framework in Figure. 4.27 is executed. The intermediate outputs from the framework are depicted for visual interpretation, along with their respective similarity scores. The scores range from +1, indicating the highest similarity, to -1, representing the highest dissimilarity between the input images. In the Figure, the left image of each pair is generated from the initial latent sample, while the right image is synthesized using the latent vector predicted by the identity encoder. From the Figure, it can be observed that the scores are very close to +1, indicating high similarity and the visual facial structures are also similar. Observing the images generated from encoded vectors, it can be noted that the hairlines are flattened and smoothed, particularly when reconstructing curly hair (see Figure 4.30(f and l)). And some color intensity changes in



(a) s=0.96 (b) s=0.83 (c) s=0.98

(d) s=0.79 (e) s=0.63 (f) s=0.96

(g) s=0.95 (h) s=0.76 (i) s=0.92

(j) s=0.89 (k) s=0.91 (l) s=0.81

(m) s=0.95 (n) s=0.94 (o) s=0.98

Figure 4.30: Example output of evaluation framework in Figure.4.27 for identity preservation. s is the cosine similarity score between the facial representation of the image from the initial latent (left) and reconstructed image from estimated latent $\hat{\mathbf{z}}$ (right) for each pair. The score ranges from +1 (most similarity) to -1 (most dissimilarity).

CK+ results. Except this, we see high resemblances between those images.

To execute the evaluation framework from Figure 4.28, two pairs of random latent and label vectors are generated and mapped onto image space by the pre-trained generator. Later, the label vectors are exchanged, but their initial latent samples are kept for image generation. The output from this process is illustrated in Figure 4.31. Since the latent vectors encode subject identity information, using the same latent vectors results in the same subject in the image. As the labels encode emotion classes, changing them results in different facial expressions.

In Figure.4.31, initial labels $\mathbf{y}_1$ and $\mathbf{y}_2$ from JAFFE (middle row) represent the mixture of classes anger/fear and disgust/fear, respectively. For MUG, they represent disgust/sadness and anger/happiness, respectively. From the Figure, it is evident that facial expressions are changed, especially in MUG, as both mixture classes do not include the same emotion. In JAFFE, where

(a) $G(\mathbf{z}_1, \mathbf{y}_1)$ (b) $G(\mathbf{z}_2, \mathbf{y}_2)$ (c) $G(\mathbf{z}_1, \mathbf{y}_2)$ (d) $G(\mathbf{z}_2, \mathbf{y}_1)$

Figure 4.31: Example output of evaluation framework in Figure.4.28 for feature disentanglement. G is the pre-trained generator of DCGANs. $\mathbf{z}$ is the latent sample randomly drawn from prior Gaussian distribution. $\mathbf{y}$ is the random label that is two-hot encoded. $G(\mathbf{z}, \mathbf{y})$ indicates that the latent vector is mapped onto image space by the pre-trained generator. The upper row is from JAFFE. The lower row is from MUG.

both labels include fear expressions, subtle changes in facial expressions are observed near the eye regions and forehead. By observing the illustration result for both datasets, it can be seen that the subject identity in the image does not change.

Compared to previous EmoGAN models[91, 92, 93], the current EmoGANs3 has the advantage that it does not require an image pair. Instead, it takes labels to encode the class information of the image pair. This approach reduces the input dimensions significantly, as labels only require 6 dimensions compared to two additional images with a size of 64x64 each. Therefore, a significant reduction in input dimensions is achieved. Facial features and facial expressions can be controlled separately since each is represented by different vectors. This allows for much greater control in the image generation process. It is important to note that this level of controllability is not available in other EmoGAN models.

Chapter 5

Evaluation on Synthesis Images

In this chapter, we will discuss a handful of metrics to determine the goodness or quality of the synthesized images by the trained generator.

5.1 Metrics for Generated Images Quality and Data Diversity

5.1.1 Inception score

The Inception score was first proposed by Salimans et al.[101] to automatically evaluate the quality of generated images. The score was then compared with subjective evaluation to overcome the necessity of subjective assessment of generated images. Their work suggested that the Inception score is well correlated with the subjective score. As its name suggests, the calculation includes the Inception model [107], which was previously trained for image classification tasks to classify the generated images.

The Inception score simultaneously measures two properties of the generated images: image quality and diversity. For image quality, the score evaluates the generator based on whether the generated images resemble objects that the pre-trained model can classify. For diversity, the score evaluates the generator based on the variety of generated images it can produce. If both properties hold true, the Inception score will be high. Since the Inception model [107] is trained to classify 1,000 objects, the Inception score ranges from 1 to 1000. A higher score indicates better performance.

Given the generated images $\mathbf{x}$, the Inception Score metric computes the confidence probability y of being a particular class label by using the recognition model. Images that are strongly predicted as one label are considered high-quality images and have a low entropy value, indicating that their confidence probability scores are concentrated on a single value. Therefore, the conditional probability $p(y|\mathbf{x})$ is used to measure the image quality of the generated images.

To measure diversity among the images, the metric utilizes the marginal probability, which is the probability distribution of all generated images. Since we prefer a large diversity of data, entropy values should be spread out, and a high entropy value is preferred. The relative entropy between these two probabilities is measured by KL divergence. Therefore, the equation for the Inception Score metric is as follows:

$$s = exp(\mathbb{E}_{\mathbf{x}}\mathrm{KL}(p(y|\mathbf{x})||p(y))) \tag{5.1}$$

where s is the inception score.$\mathbf{x}$ is the set of generated images. y is the confidence probability

score predicted by the recognition model. $p(y|\mathbf{x})$ is the conditional probability to measure the generated image quality. $p(y)$ is the marginal probability of all generated images to measure image diversity.

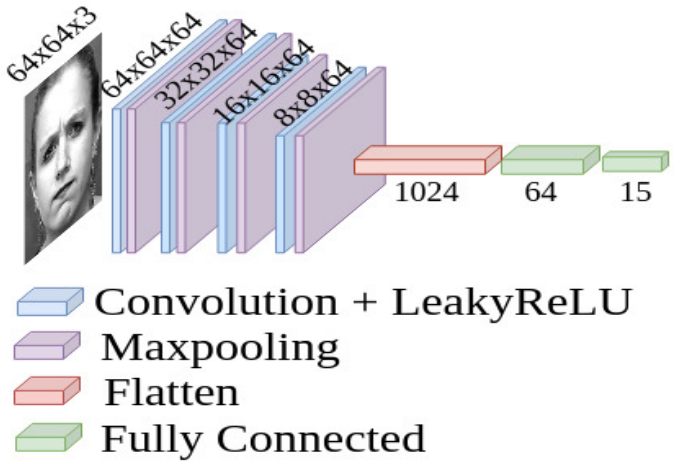


Figure 5.1: Network configuration for multi classes mixed facial expressions recognition model

However, the Inception score only works for images that include objects known to the model used in the calculation process, such as the Inception model. The Inception model is primarily trained for object classification and is not intended for facial expression classification. Therefore, we utilized a facial expression recognition model instead.

Table 5.1: Accuracy of multi classes mixed facial expressions recognition model in 10 fold cross validation set

Accuracy a (± Standard Deviation)			
Fold No.	Datasets		
	CK+	JAFFE	MUG
1	84.5%	96.4%	98.4%
2	74.6%	97.9%	97.0
3	74.3%	99.3%	97.2%
4	67.1%	99.3%	99.0%
5	74.3%	99.3%	99.1%
6	70.0%	100.0%	97.2%
7	82.9%	98.6%	97.9%
8	75.7%	97.8%	97.3%
9	57.1%	97.8%	96.4%
10	67.1%	97.8%	97.6%
Average accuracy ($\hat{a}$)	72.8% (±7.2)	98.4% (±1.0)	97.7% (±0.8)

The network configuration of the model is illustrated in Figure 5.1. It is a typical four-layered convolutional neural network trained for classifying multiple mixed emotion classes. The model is optimized by Adam Optimizer with a learning rate of 1e-03 and is updated over every 32 batches for 50 iterations. The models are evaluated on 10-fold validated sets and results are shown in Table.5.1. Based on the cross-validated results, the model achieves

approximately 73% in CK and 98% in JAFFE and MUG. This facial expressions model is used to calculate the Inception score. Similar to the inception model, the higher score for better generation results. Since this recognition model is trained for 15 classes, the maximum score is 15 and the minimum is 1.

After training the recognition model, it is used to execute Equation.5.1 for label prediction. The results are tabulated in Table.5.2. Based on the results, EmoGANs3 works best among all EmoGANs models in general. For CK+, the score of EmoGANs3 is lower than EmoGANs and EmoGANs1 but is higher than other state-of-art models. Compared to other GANs variants, EmoGANs3 ranked fourth with the JAFFE dataset. A sample from each model such as GANs, DCGANs, FastGANs, and EmoGANs3 is illustrated in Figure.5.2. Qualitatively EmoGANs results are comparable with other models.

Table 5.2: Results for Inception Score metric

Models	Datasets		
	CK+	JAFFE	MUG
GANs [27]	5.7 (± 0.4)	9.9 (± 0.4)	3.8 (± 0.1)
DCGANs [31]	8.7 (± 0.5)	**10.1 (± 0.4)**	8.4 (± 0.2)
StyleGANs [37]	8.7 (± 0.6)	5.0 (± 0.3)	11.4 (± 0.2)
WGANs [38]	5.9 (± 0.3)	3.9 (± 0.3)	7.6(±0.2)
FastGANs [120]	6.0 (± 0.6)	9.2 (± 0.5)	6.0 (± 0.1)
EmoGANs [91]	8.9 (± 0.6)	4.4 (± 0.3)	8.2 (± 0.2)
EmoGANs1 [92]	**10.1 (± 0.5)**	8.4 (± 0.4)	4.1 (± 0.1)
EmoGANs2 [93]	9.7 (± 0.9)	8.8 (± 0.4)	8.8 (± 0.3)
EmoGANs3	8.8 (±0.6)	9.0 (± 0.2)	**11.4 (± 0.1)**

(a) GANs (b) DCGANs (c) FastGANs (d) EmoGANs3

Figure 5.2: An random sample from each generator of GANs, DCGANs, FastGANs, and EmoGANs3 used in inception score metric for JAFFE dataset

5.1.2 Frechet Inception Distance (FID)

Heusel et al. [106] introduced the Frechet Inception Distance (FID) to evaluate the performance of generators. Previously, the metric for inception score did not utilize training images for computation, thus it was unable to measure the similarity between generated and training images. Therefore, the objective of FID is to evaluate generator performance by comparing the statistics of generated and training images.

Similar to the inception score metric, the FID metric also utilizes a pre-trained Inception model as a feature extractor to capture the feature representations of input images. Sets

of generated and training images are transformed into feature vectors using the pre-trained Inception model. The features of these images are then summarized as multivariate Gaussian distributions, and the distance between the two distributions is measured using either the Frechet distance or the Wasserstein-2 distance. The equation of Frechet Inception Distance can be summarized as follows.

$$d^2 = \|\mu_1 - \mu_2\|^2 + \mathrm{Tr}(C_1 + C_2 - \sqrt{2C_1C_2}) \tag{5.2}$$

where d is the FID score, μ_1 and μ_2 are the corresponding feature-wise mean of the training and generated images. C_1 and C_2 are the covariance matrices of the respective feature vectors of the training and generated images. Tr refers to the trace linear algebra operation.

Similar to the previous inception score, the same recognition model is used to get feature vectors of the training and generated images. Equation.5.2 is executed. Since it is a distance-based metric, the score ranges from zero to indicate the most similarity between training and generated images. The maximum score differ based on the statistics of the two collection images. Results are tabulated in Table.5.3.

Based on the result from Table.5.3, the EmoGANs3 model performed best in comparison with other EmoGANs models except in CK+, consistent with outputs from the inception score metric. With other GANs variants, it ranked second in both JAFFE and MUG datasets and third in CK+. For visual assessment, the generated image from EmoGANs3 is comparable with the one from the FastGANs generator in Figure.5.2 for JAFFE and in Figure.5.3 for CK+. A visual sample from WGANs and EmoGANs3 for the MUG dataset is also shown in Figure.5.3. It is evident that generated images from EMoGANs3 are visually better than the ones from the WGANs model.

Table 5.3: Results for Frechet Inception Distance metric

Models	Datasets		
	CK+	JAFFE	MUG
GANs [27]	254.2	242.0	559.4
DCGANs [31]	53.8	143.8	182.7
StyleGANs [37]	98.3	647.7	**71.0**
WGANs [38]	219.0	1102.1	98.1
FastGANs [120]	205.8	**81.4**	934.9
EmoGANs [91]	57.5	713.9	385.9
EmoGANs1 [92]	72.6	281.4	1140.3
EmoGANs2 [93]	**25.0**	309.8	467.4
EmoGANs3	56.4	107.8	136.6

5.1.3 Blind/Referenceless Image Spatial Quality Evaluator (BRISQE)

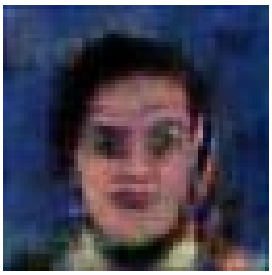
The Blind/Referenceless Image Spatial Quality Evaluator (BRISQE) was first proposed by Mittal et al. [121] to measure image quality without a reference image. It derives feature vectors using the input pixels without transforming them into another feature space, such as wavelet transformation. BRISQE includes three main stages: extracting Natural Scene Statistics (NSS), calculating feature vectors, and predicting scores using Support Vector Machine (SVM).

Generally, the distribution of normalized pixel intensities in high-quality images follows a Gaussian distribution. Therefore, the metric normalizes the given images to measure the

(a) DCGANs (CK+)

(b) EmoGANs2 (CK+)

(c) EmoGANs3(CK+)



(d) WGANs (MUG)

(e) EmoGANs3(MUG)

Figure 5.3: An random sample from each generator of WGANs, and EmoGANs3 used in Frechet Inception Distance metric for JAFFE dataset

distortion quantity based on how much the normalized pixels deviate from the ideal Gaussian distribution. In the first stage, it normalizes the pixel intensities using Mean Subtracted Contrast Normalization (MSCN). The normalization equation is as follows.

$$\hat{I}(u,v) = \frac{I(u,v) - \mu(u,v)}{\sigma(u,v) + c} \tag{5.3}$$

where u and v are the coordinates of the image pixels. μ is the local mean and σ is the local variation. c is the division constant to avoid zero division error.

The local mean is the same as applying a Gaussian filter to the input image, which is given as follows.

$$\mu = \mathbf{W} * \mathbf{I} \tag{5.4}$$

where μ is the local mean, $\mathbf{W}$ is the Gaussian kernel function and $\mathbf{I}$ is the given image.

Local variation or deviation from the mean can be calculated as follows.

$$\sigma = \sqrt{\mathbf{W} * (\mathbf{I} - \mu)^2} \tag{5.5}$$

where is the local variation.

Next, the metric finds four differently oriented images shifted from the normalized pixels image as the image quality also depends on the relationships among neighborhood pixels. In BRISQE, it uses four orientations as horizontal H, vertical V, left-diagonal D_1, and right-diagonal D_2 which can be calculated as follows.

$$H(u,v) = \hat{I}(u,v)\hat{I}(u,v+1) \tag{5.6}$$

$$V(u,v) = \hat{I}(u,v)\hat{I}(u+1,v) \tag{5.7}$$

$$D_1(u,v) = \hat{I}(u,v)\hat{I}(u+1,v+1) \tag{5.8}$$

$$D_2(u,v) = \hat{I}(u,v)\hat{I}(u+1,v-1) \tag{5.9}$$

The previous normalized image is fitted onto the Generalized Gaussian Distribution (GGD), which returns shape and variance values. Oriented images are also fitted onto the Asymmetric Generalized Gaussian Distribution (AGGD), which returns shape, mean, left variance, and right variance values. In total, 18 values are returned for five images. The original image is then downsampled by half and undergoes the same process, resulting in 36 feature vectors, which are later used by the learning algorithm such as a support vector machine to predict the score to assess the image quality. Lower scores indicate better quality. We used the implementation supported by the Python library.

We included the output from morphing to assess its quality as well. Since the previous two metrics (inception score and FID score) required the facial expression recognition model which is trained on the morphing images, those metrics will favor the morphing images for the score calculation. Unlike these two metrics, BRISQE calculates the feature vector from its given images for score prediction. Therefore, morphing images are also assessed using BRISQE. Results are shown in Table.5.4.

Based on the objective results, it was found that EmoGANs3 works best among other EmoGANs models for all datasets. When it is compared with other GANs models, it ranked first in the MUG and CK+ datasets and has better scores than morphed images. It ranks fifth in the JAFFE dataset. Similar to results from the previous metrics, the quality of generated images by EmoGANs is visually comparable as in Figure.5.4.

Table 5.4: Results for Blind/Refereenceless Image Spatial Quality Evaluator metric

Models	Datasets		
	CK+	JAFFE	MUG
GANs [27]	30.7	**20.8**	35.4
DCGANs [31]	28.4	27.9	63.3
StyleGANs [37]	**19.7**	21.8	16.5
WGANs [38]	34.8	61.2	28.8
FastGANs [120]	72.1	75.3	38.4
Morphing	27.6	25.7	18.9
EmoGANs [91]	29.4	85.0	53.8
EmoGANs1 [92]	25.4	54.8	155.3
EmoGANs2 [93]	29.1	30.8	25.0
EmoGANs3	20.1	31.8	**11.0**

(a) (b) (c) (d) (e)

Figure 5.4: An random sample from each generator of (a)GANs, (b)DCGANs, (c)Morphing, (d)EmoGANs2 and (e)EmoGANs3 used in BRISQE metric for JAFFE dataset

5.2 Metric to evaluate mixed facial expressions in the generated images

5.2.1 Cosine Distance Metric for Facial Expressions Features between generated images and morphed images

Previous metrics were mainly used for quality assessment and data diversity. Since our objective is to synthesize mixed facial expression images, we apply the methodology shown in Figure 5.5 to measure the distance between the generated and training images in the facial expression feature space. The previous multi-class facial expression model discussed in Section 5.1.1 is used to extract features from a batch of training and generated images. Once transformed into the same space, the distance among these features is measured using the cosine distance metric. The cosine distance metric is preferred as it limits the score from 0 to +1. A lower distance indicates that the features from the generated images are located close to the features from the training images.

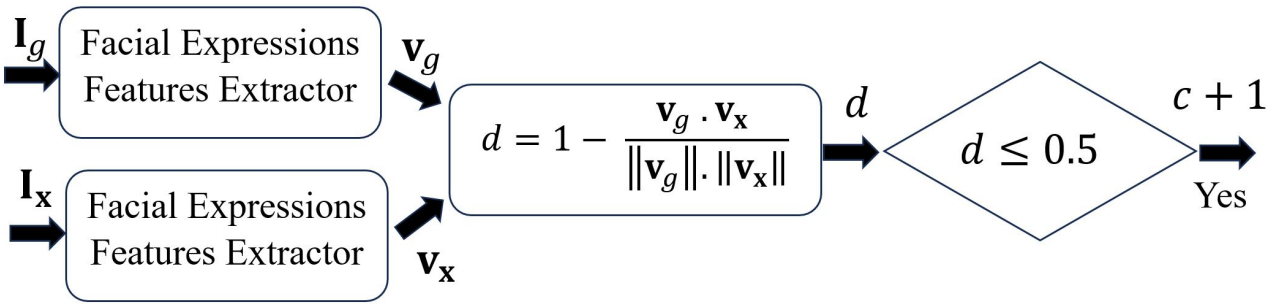


Figure 5.5: Methodology to measure the distance between generated and training images in the feature space. I_g and $I_\mathbf{x}$ are the generated and training images. $\mathbf{v}$ refers to feature vectors extracted by the facial expressions recognition model from Section.5.1.1. d refers to cosine distance. c refers to the number of samples which has a close distance than the threshold value 0.5.

Unlike the general GANs variation, for example, GANs[27], DCGANs[31], WGANs [38], FastGANs [120], generators from EmoGANs, EmoGANs1 and EmoGANs2 obtain two basic facial expressions images. Therefore, generated images from those images have a reference training image for those input images to measure the distance. For EmoGANs3, the classes of generated images are controlled by the two-hot encoded label vectors. Therefore, an example of a reference image from each mixed class is used for calculation. However, generators from

the general GANs variants produce images from a random latent sample. Therefore, they do not have reference training images for a specific generated image. Therefore, cosine distance is computed for the generated images from EmoGANs models.

Figure.5.4 gives the number of samples which has a closer distance from the training images. This value is converted into percentages divided by the number of total samples of each dataset. Results are shown in Table. 5.5. A higher percentage indicates that the majority of generated images possess similar facial expressions as training images in the feature space.

Table 5.5: Results for measuring the cosine distance between facial expressions from generated and training images.

Models	Datasets		
	CK+	JAFFE	MUG
EmoGANs [91]	95.2%	67.5%	**68.6%**
EmoGANs1 [92]	**95.6%**	**95.5%**	34.4%
EmoGANs2 [93]	36.5%	42.5%	30.1%
EmoGANs3	28.6%	67.2%	32.5%

According to the results, the majority of generated images from EmoGANs and EmoGANs1 have similar facial expressions as morphed images in CK+ and JAFFE. However, it is difficult to visually assess the generated images as the image quality is low and noisy. Although the score of EmoGANs3 is lower than EmoGANs and EmoGANs1, it is higher than EmoGANs2 in MUG and JAFFE. Besides, it had been proved that action units of generated images by EmoGANs3 is consistent with the ones from morphed images in Section.4.2.1.4. From the visualization results, image quality is better than the previous two models in Figure.5.6.

5.3 Metric to evaluate feature disentanglement property

The previous sections have demonstrated that the EmoGANs3 model generated images with superior quantitative scores compared to other EmoGANs models. The EmoGANs3 model utilizes two distinct feature vectors, namely facial expressions and subject identity information, to control the image generation process. In this section, we will evaluate the EmoGANs3 model based on the degree of disentanglement between facial expressions and subject identity information. It is important to note that, unlike the EmoGANs3 model, the previous EmoGANs models lack the capability to control and isolate these features. Consequently, we will initially propose a methodology to separate these features and subsequently compare the results with those obtained from the EmoGANs3 model, in line with the objectives of our research.

5.3.1 Methodology to manipulate latent space

As presented in the previous chapters, GANs can map the latent samples from the prior distribution onto the image space after training. However, it is still unknown how semantics representations of the generated images are arranged in the latent space. Therefore, this chapter observes the visual concepts of the generated images in the latent space by manipulation of facial expression attributes to make changes in the image space.

Figure.5.7 illustrates the proposed methodology to manipulate the facial expressions attributes in the latent space[105]. It includes four main parts: Image generation, Emotion

(a) EmoGANs (b) EmoGANs1 (c) EmoGANs3

Figure 5.6: An random sample from each generator of EmoGANs, EmoGANs1, and EmoGANs3 used in measuring cosine distance in feature space. The upper row is for CK+, the middle is for JAFFE and the lower is for MUG.

classification through facial expressions, Latent samples collection with respect to emotion labels, and finally facial expression attributes manipulation in the latent space. Each component included in the Figure is sequentially discussed.

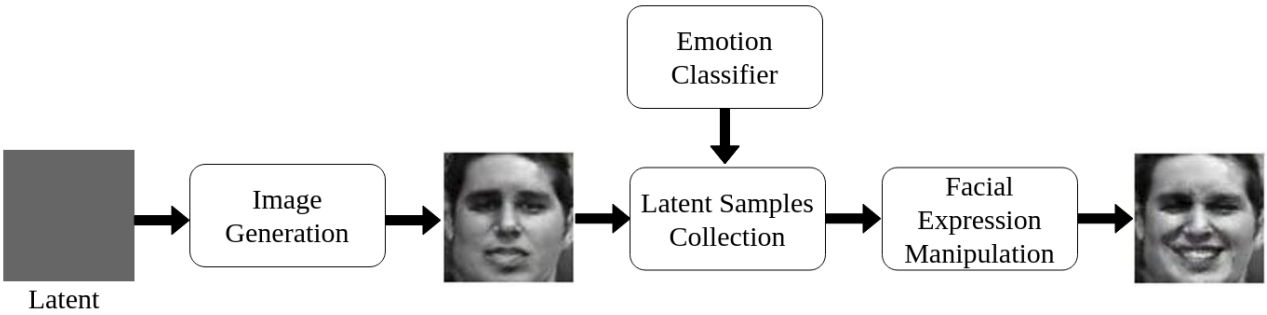


Figure 5.7: Overview of facial expression manipulation in the latent space

5.3.1.1 Image Generation

Network Configurations Since the latent space used by GANs' generator is explored, the networks are constructed based on Deep Convolution GANs (DCGANs) [31]. Detailed network configurations can be seen in Figure. 5.8. For the generator G, it up-samples a 100-dimensional

latent sample which is uniformly drawn from the prior Gaussian distribution with Convolution Transpose layers. Leaky rectified linear unit (Leaky ReLU) activation is used in G except the final layer. The latent samples are mapped onto 80 × 80 image space. For the discriminator D, five-layered of convolution filters are used for downsampling and feature extraction with non-linear activation such as ReLU. In the end, sigmoid binary cross entropy loss is used to predict the distributions of the given inputs.

Experiment settings Facial expressions images with basic emotion labels are employed to train the models. The weights are updated through mini-batch size 128 for 1000 iterations. The models are trained using Adam optimizer[95]. The learning rate is set to 1e-04 for both sub-models G and D.

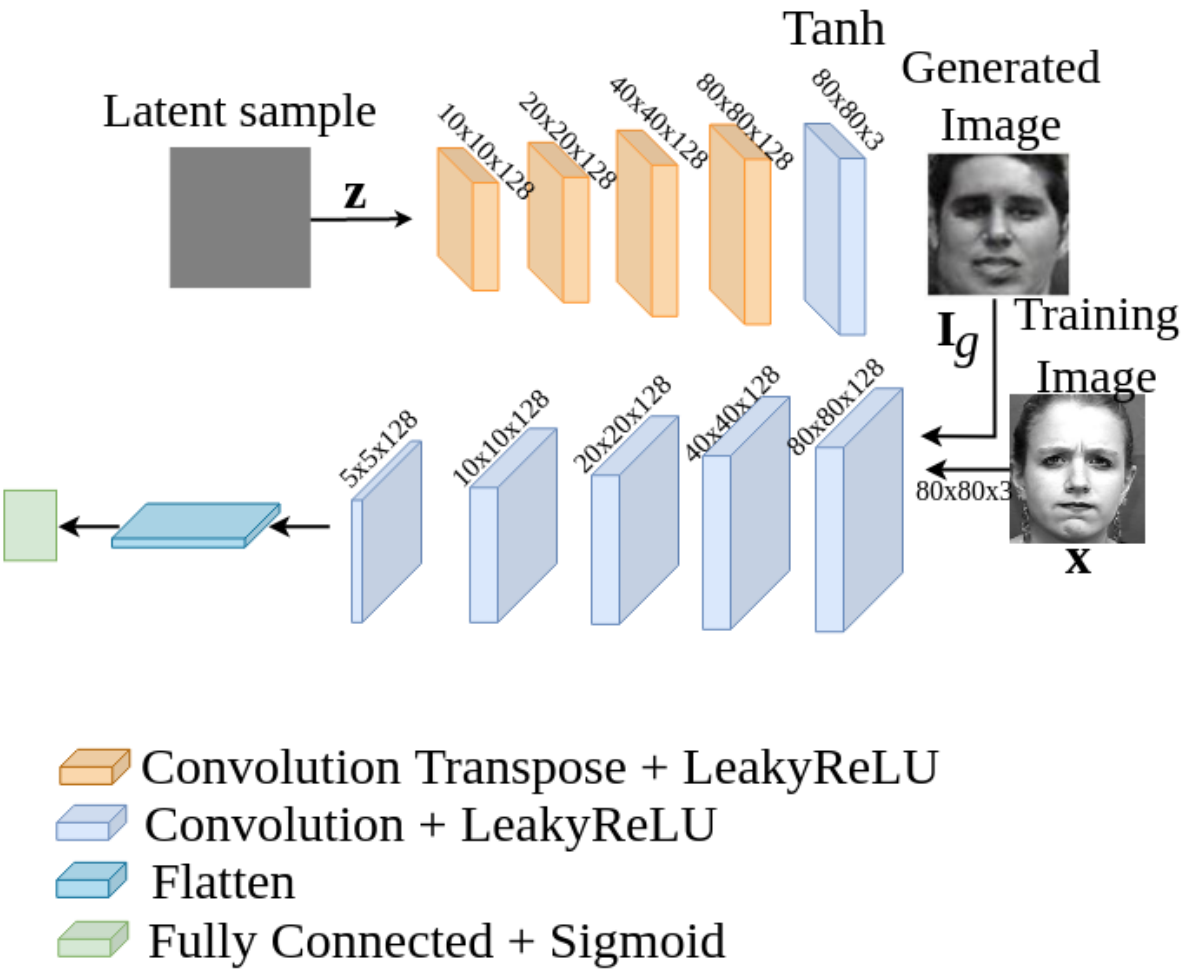


Figure 5.8: Network configurations of image generation model (Deep Convolution Generative Adversarial Networks)

Evaluation Frechet Inception Distance (FID) metric proposed in [106] is used to evaluate the quality of generated images by G of DCGANs. In the initial FID computation, the pre-trained objects recognition model such as inception v3[107] without the final classification layer was used to capture the specific features of the input images. The features are summarized as a multivariate Gaussian by finding the means and covariance of the images, which is later used

to compute the Frechet distance. A lower score indicates better image quality. FID metric can be summarized as follows.

$$d^2_{FID} = ||\mu_1 - \mu_2||^2 + \mathrm{Tr}(\mathbf{C}_1 + \mathbf{C}_2 – 2(\mathbf{C}_1\mathbf{C}_2)^{\frac{1}{2}})) \tag{5.10}$$

where d^2_{FID} is the score of Frechet distance in squared unit. μ_1 and μ_2 refer to the feature-wise mean of the real and generated images respectively. $\mathbf{C}_1$ and $\mathbf{C}_2$ are the covariance matrices for real and generated images. Tr refers to the trace linear algebra operation.

In our experiment, the inception v3 model[107] was replaced by the pre-trained emotion recognition model, as facial expressions-related features are desired. We obtained a score of 21.82 with the DCGANs model which is lower than the score of 96.09 achieved by vanilla GANs [27].

5.3.1.2 Emotion Classification

Network Configurations In this study, we developed an emotion recognition model that uses three convolutional layers to recognize seven basic emotions, including anger, disgust, fear, neutral, happiness, sadness, and surprise. The convolutional feature maps were reshaped and fully connected to recognize emotions. Softmax activation is used in the last dense layer for multi-classifications. The complete architecture is shown in Figure 5.9. Prior to training, the facial images in the input were detected and standardized.

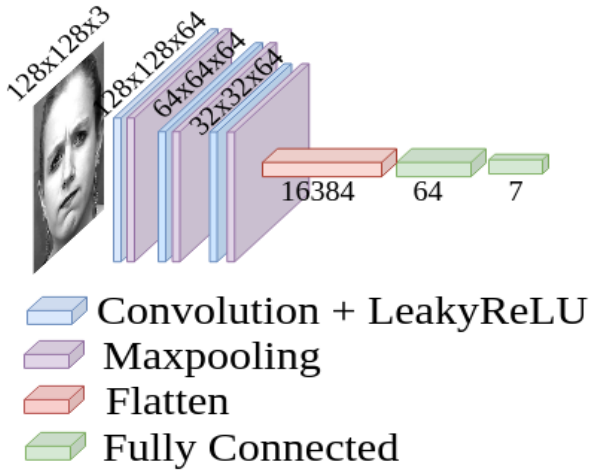


Figure 5.9: Network configurations of emotion recognition model (Convolution Neural Network(CNN))

Experiment settings The model is trained using Adam optimizer [95] with a learning rate 1e-03 for 50 iterations. Weights are updated for each mini-batch size of 128.

Evaluation The model is evaluated through 10 fold cross-validation where one subset is used to test the model performance for each iteration. Due to the simplicity of the frontal facial expressions used in the input data, the current configuration of the CNN model was effective in achieving an accuracy of approximately 91% in the CK+ dataset.

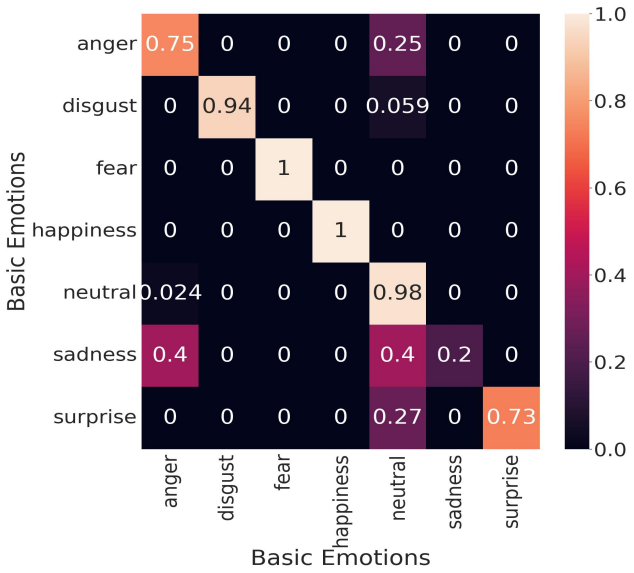


Figure 5.10: Confusion matrix of the emotion classification model for each emotion

Figure 5.10 shows the confusion matrix used to evaluate the model's performance for each class. We can observe that the model confuses sadness images with anger and neutral expressions. This result is reasonable since the training data for the sadness class is the fewest among the emotional classes, with only 28 images in the CK+ dataset.

5.3.1.3 Latent Sample Collection With Respect To Emotion Labels

To collect the latent samples with respect to emotion labels, previously trained generator G and emotion classifier are employed. The complete system flow can be seen in Figure.5.11. First, the random latent is drawn from the Gaussian distribution and mapped onto image space by G. An emotion classifier was used to predict the labels for the generated images. If the predicted labels and desired emotion label is matched, the initial latent is recorded with emotion labels. Otherwise, the latent are re-sampled from the same distribution. Two thousand latent samples are collected for each emotion class.

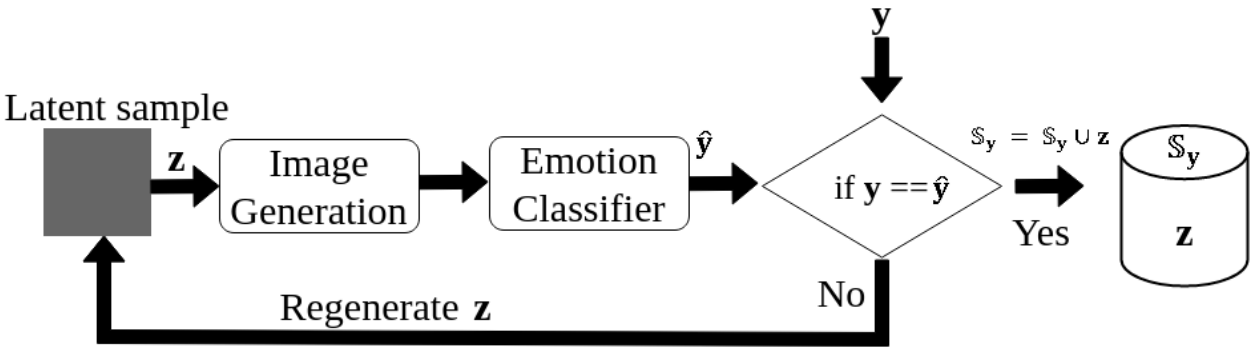


Figure 5.11: System flow to collect latent samples with respect to emotion labels. $\mathbf{z}$ is the latent sampled from the prior distribution $\mathcal{N}(0,1)$. $\mathbf{y}$ is the desired emotion label. $\hat{\mathbf{y}}$ is the label predicted by the emotion classifier. $\mathbb{S}_\mathbf{y}$ is a set that includes latent samples with emotion label $\mathbf{y}$ and initially an empty set ϕ.

5.3.1.4 Facial Expression Manipulation

The concept behind manipulating facial expression attributes in the latent space is that if generated images have different facial expressions, the underlying semantics or facial expression attributes in the latent space must also differ. By locating the boundary that separates these two expressions, the semantics of a specific latent sample can be altered by pushing it towards the opposite side of the boundary. Figure.5.12 shows the manipulation process.

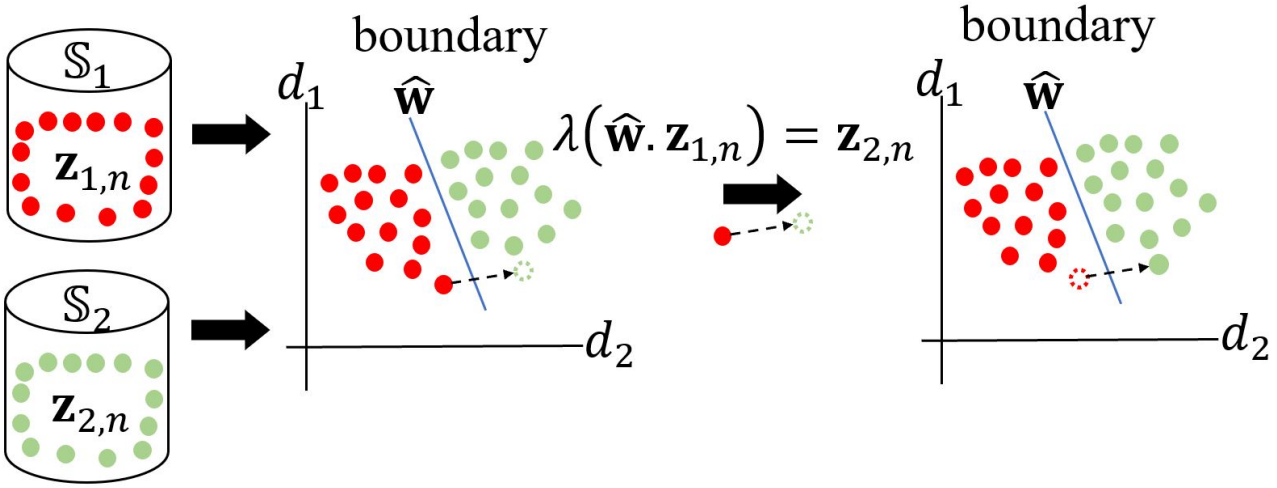


Figure 5.12: Diagram to manipulate the facial expressions attribute in the latent space. $\mathbb{S}_1$ belongs to the set containing the latent samples $\mathbf{z}$ with emotion label 1, for example, '*happiness*' emotion. $\mathbb{S}_2$ refers to the set for emotion label 2, for example, '*sadness*' emotion. $\hat{\mathbf{a}}$ refers to the decision boundary that separates the latent samples from two different classes. λ is the control parameter for direction. n is the number of latent samples. In this experiment, two thousand samples with respect to emotion labels are collected. d is the dimensions of the latent samples. Only 2 dimensions are used for illustration purpose.

To manipulate the facial expressions, first the boundary line that classifies two different emotion classes is found using binary Support Vector Machine (SVM) algorithm. SVM is the supervised learning algorithm that finds the hyperplane with a maximum margin to the closet points or support vectors in classification. Suppose, we have the feature points $\mathbf{z}$ in d dimensional space such as $\mathbf{z} \in \mathbb{R}^d, d \leq 100$. SVM defines the initial hyperplane as follows.

$$\mathbf{W}^{\mathrm{T}}.\mathbf{z} + b = 0 \tag{5.11}$$

where $\mathbf{W}$ denotes the weight vector of SVM. b is the intercept or bias of the hyperplane equation. $\mathbf{z}$ is the 100 dimensional latent samples with corresponding emotion label.

Next, the hyperplane is required to best separate the points to reduce mis-classification and fit by minimizing the distance function as follows.

$$d_H(\mathbf{z}) = \frac{\mathbf{W}^T.\mathbf{z} + b}{||\mathbf{W}||_2} \tag{5.12}$$

where H stands for hyperplane. d is the distance between a given point $\mathbf{z}$ and the hyperplane. b is the bias term. $||\mathbf{W}||_2$ is the Euclidean norm of the weight vector.

The objective of SVM is to find the hyperplane with the maximum margin from the closest points. It can be written as

$$\mathbf{W}^* = \arg\max_{\mathbf{W}}(\min d_H(\mathbf{z})) \tag{5.13}$$

where $\mathbf{W}^*$ is the optimal weights that gives the best hyperplane H for binary facial expressions classification.

After training SVM, the facial expressions of a particular latent is altered by using the boundary line of the desired emotion class as follows.

$$\hat{\mathbf{z}} = \lambda(\hat{\mathbf{w}}.\mathbf{z}) \tag{5.14}$$

where $\mathbf{z}$ is the initial latent with emotion label $\mathbf{y}$. $\hat{\mathbf{w}}$ is the boundary norm which is defined as $\frac{\mathbf{W}}{|\mathbf{W}|}$. λ is the control parameter. $\hat{\mathbf{z}}$ is the modified latent with different emotion label.

5.3.1.5 Discussion

Theoretically, λ can be any real number. However, based on our trial experiments, we found that values between -2.0 and 2.0 provide the best results for disentangling facial expressions from subject identity features. Therefore, we set the default λ value within this range. As shown in Figure 5.12, when the λ value moves in the same direction as the input expression, i.e., $\lambda > 0$, the generated facial expressions are similar to the input expressions. Conversely, when $\lambda < 0$, the expressions are opposite to the input expressions. At $\lambda = 0$, the expressions are similar to those located near the boundary line. An example output of changing the facial expressions from happiness class to sadness is illustrated in Figure.5.13. Complete experiment results for other transformations can be seen in the Appendix section B.3.

Based on the experimental results, changing facial expression can also alter the other facial attributes such as face shape. For an example, when facial expressions are changed to a surprise expressions, it results in changing face shape vertically by widely opening the mouth and raising eyebrows upwards (See the illustration results of the surprise class in the Appendix section B.3). If similar facial expressions are altered, for example sadness and fear, the facial expression attribute disentangles with other facial attributes while maintaining the same subject identity.

We compare the results obtained by the current study with other methodologies proposed in [31, 108]. Figure.5.14 shows the result using the arithmetic operations on mean latent vectors proposed in [31]. From the Figure, we can observe that the facial expressions attribute and identity attribute are tangled in the latent space. Figure.5.15 illustrates the result using the deep feature interpolation proposed in [108]. Since the model in this work is trained with facial attributes instead of facial expressions, the attributes that make changes in facial expressions are altered such as 'smiling', 'frowning', 'mouth slightly open', and 'mouth wide open'. From the result, we can perceive that image quality is degraded as a better result depends on accurate face alignment. Unlike the compared work[108], the methodology in the current study[105] semantically controls the facial expression attributes in the latent space by pushing the sample towards the opposite side of the decision boundary supported by SVM. Besides, it does not require careful face alignment.

Manipulating the facial expression attributes in the latent space can improve the performance of emotion recognition by providing augmented data when the training data is insufficient and unbalanced. As presented earlier in this section, the emotion recognition model misrecognizes the samples from the sadness class the most, as the training samples for that class are the fewest. Therefore, we augment the sadness samples using the proposed methodology, altering facial expressions from the happiness class to sadness expressions. These augmented samples are used to train the previous emotion recognition model. Figure 5.16 shows the performance of the emotion recognition model after augmentation. Compared to the previous confusion matrix in Figure 5.10, the performance in the sadness class has improved from 20% to 80%.

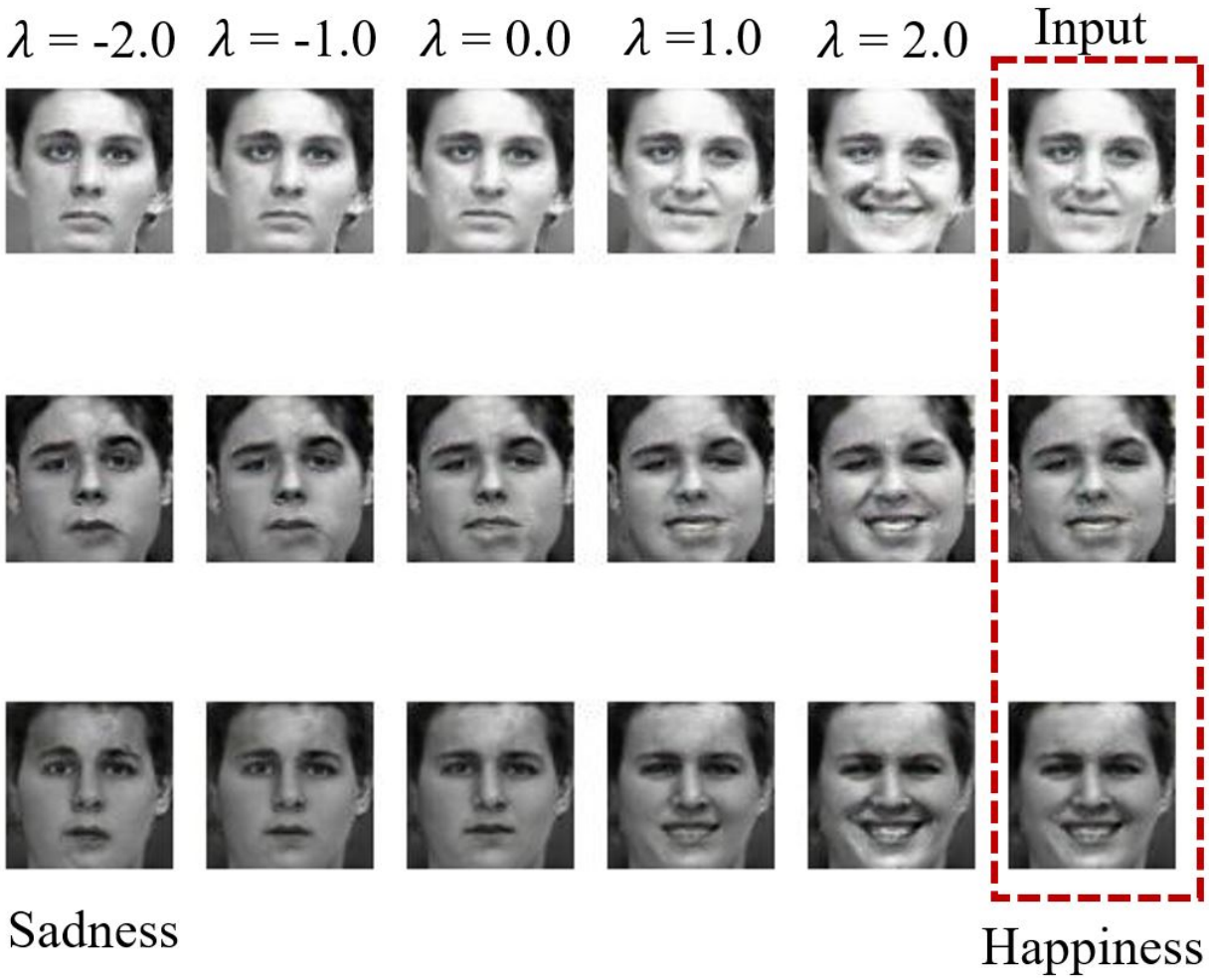


Figure 5.13: An example output of facial expressions manipulation from *happiness* class to *sadness* class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment. More experimental results for different classes can be found in Appendix section B.3.

5.3.2 Cosine similarity metric for subject identity information between an input image and transformed image

Our hypothesis is that if the models or methodology have disentanglement properties between facial expressions and subject identity information, the transformed images by those models should maintain similar faces. Therefore, to objectively assess the facial similarity between input and transformed images, the evaluation methodology is displayed in Figure 5.17. We use the pre-trained VGG Face model [87] to extract facial features and measure the cosine similarity metric among these features.

In the proposed methodology in Section 5.3.1, a single input image is transformed by multiplying it with the boundary norm and control parameter λ. Subsequently, we measure the cosine similarity metric among the facial features of the given image and the transformed image to analyze the extent to which these images include similar faces. We expect that the transformed image at the decision boundary (λ = 0) will have mixed facial expressions while still maintaining the subject identity, as the boundary line exists between two different classes.

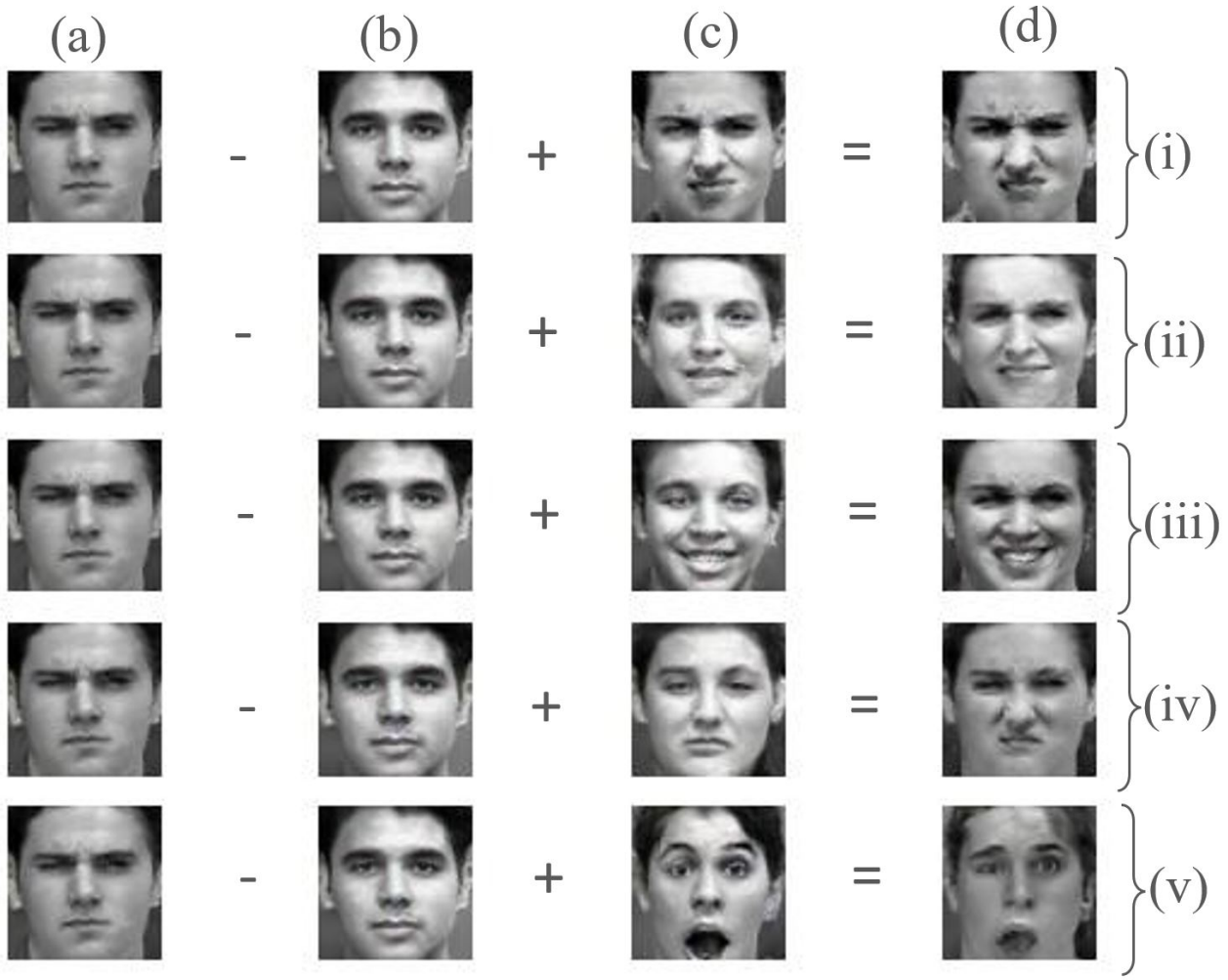


Figure 5.14: Output of manipulating facial expressions attribute in the latent space using arithmetic property proposed in DCGANs by Radford et.al [31]. Columns (a) and (b) represent the generated images synthesized from the mean latent of anger and neutral emotions. Column (c) refers to generated images synthesized from the mean latent of the corresponding emotions such as (i) disgust, (ii) fear, (iii) happiness, (iv) sadness and (v) surprise. Column (d) is the output of arithmetic operation performed on those mean latent vectors.

For comparison, we also include the StyleGANs model as a state-of-the-art, as it has the capability to mix the style between two images. StyleGANs are trained on the CK+ dataset, which exhibits less variation in styles but different facial expressions. We anticipate that by mixing style in the latent space, we can generate images with mixed facial expressions in the image space. Consequently, we measure the facial similarity between the new images obtained from mixing style in the latent space and the two input images. Since we have two input images, we calculate the cosine similarity between each input image and the corresponding mixed image separately, and then find the average similarity. The training process is implemented using the TensorFlow library for the implementation support.

Figure.5.18 (upper row) displays the input image and the transformed image at the decision boundary by the methodology[105], along with their corresponding facial similarity scores. The quantitative results indicate that the input image and the transformed image have a similar identity, as evidenced by a cosine similarity score closer to +1. This observation is also supported by a visual assessment of the displayed images. We followed similar procedures for

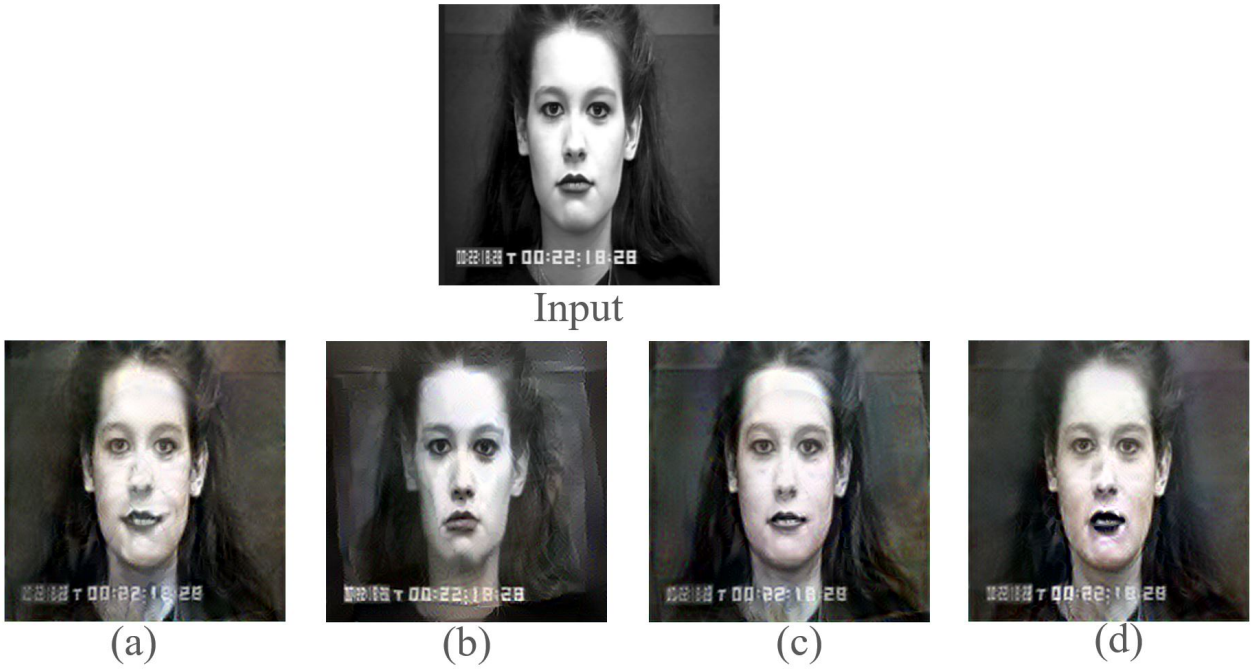
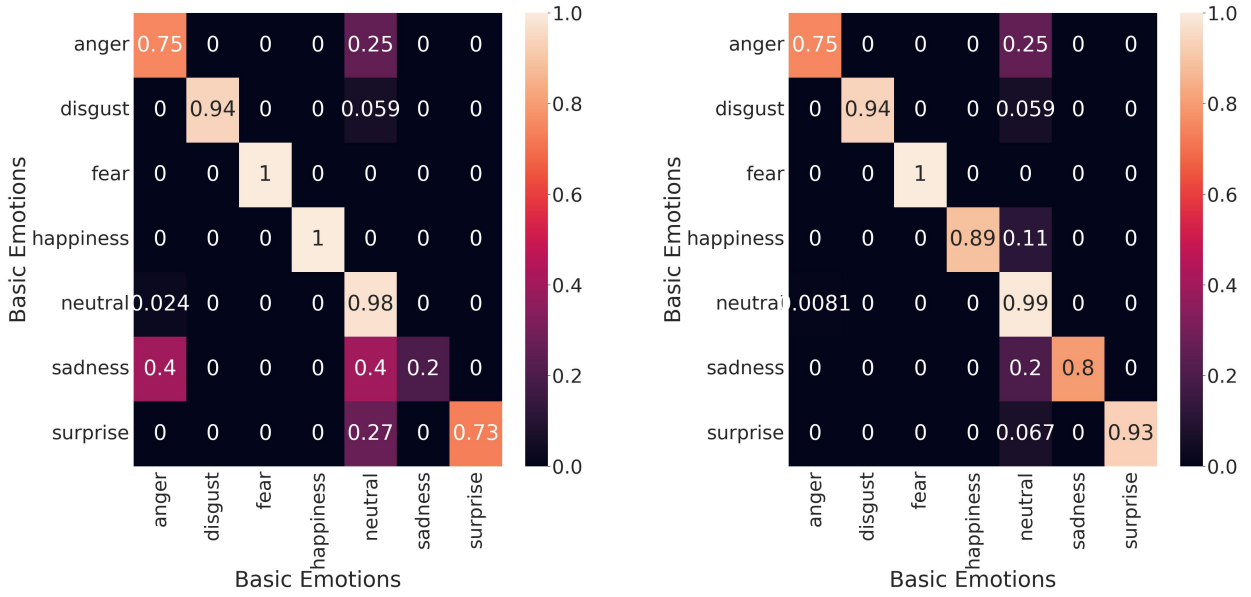


Figure 5.15: Output of editing facial attributes that results to change facial expressions in the image such as (a) '*smiling*', (b) '*frowning*', (c) '*mouth slightly open*', (d) '*mouth wide open*' using deep feature interpolation proposed by Upchurch et.al [108].



(a) before (b) after

Figure 5.16: Confusion matrix of the emotion classification model for each emotion after adding the samples created using the proposed methodology

the images in the test set. In 99% of the transformed images by the work[105], we observed similar faces compared to their respective input images. Similarly, EmoGANs3 model also maintain the facial information by the encoder model in the image generation by observing the

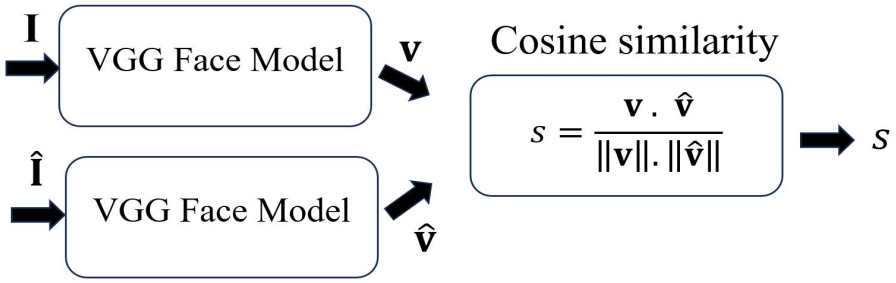


Figure 5.17: Methodology to measure the similarity between an input image and output image transformed in the latent space. **I** is an input image and $\hat{I}$ is the manipulated or transformed image. **v** represents the feature vectors containing the subject identity information extracted by the VGG Face model [87]. s is a score given by the cosine similarity metric, which ranges from -1 for the most dissimilarity to +1 for the most similarity.

(a) $s = 0.90$ (b) $s = 0.92$

(c) $s = 0.86$ (d) $s = 0.92$

Figure 5.18: Output of evaluation methodology to measure facial similarity between a given input image (left) and transformed output image (right) in each pair. The output image at the upper row is transformed by the proposed methodology [105] and the ones at the lower row are generated from the predicted latent by the facial encoder of EmoGANs3. s is the cosine similarity score calculated from facial embedding vectors. The value ranges from +1 for the most similarity and -1 for the most dissimilarity.

high similarity scores in Figure.5.18 (lower row).

Figure 5.19 presents the results for StyleGANs, including the facial similarity scores and illustrated images. In each row, the first two images represent randomly generated images by StyleGANs, while the last image is the result of mixing the style between the first two. According to the quantitative scores, the mixed image exhibits a certain degree of similarity, although it is lower than the transformed image obtained using the methodology described in Section 5.3.1. We computed the similarity metric for the test set as well. It was found that

(a) $s = 0.45$

(b) $s = 0.40$

Figure 5.19: Output of evaluation methodology to measure facial similarity between input images and transformed output image. Images from 1^{st} and 2^{nd} columns are random generated images from StyleGANs[37]. Images at 3^{rd} are the results of mixing style from input images. s is the cosine similarity score calculated from facial embedding vectors. The value ranges from +1 for the most similarity and -1 for the most dissimilarity.

45% of the mixed images have a similarity score greater than the threshold value of 0.5.

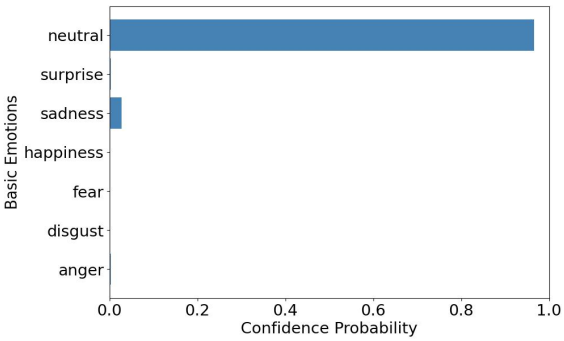
5.3.3 Facial expressions recognition on images transformed in the latent space

In the previous section, we measured the extent to which subject information is retained in the newly transformed images in the latent space. To further analyze the disentanglement property between identity and facial expressions, it is necessary to assess the facial expressions present in these transformed images. As mentioned earlier, we anticipate that the transformed or mixed images will exhibit mixed facial expressions. Therefore, we will utilize the Py-Feat facial expression recognition model [16] to recognize the facial expressions in the transformed images.

The output of emotion recognition by Py-Feat [16] are depicted in Figures. 5.20, 5.21 and 5.22. Analyzing the confidence probability scores for each emotion, we observed that the transformed images at the boundary line, obtained using the proposed methodology [105], predominantly exhibit neutral expressions rather than mixed facial expressions. In contrast, the mixed images generated by StyleGANs and EmoGANs3 displayed mixed facial expressions.

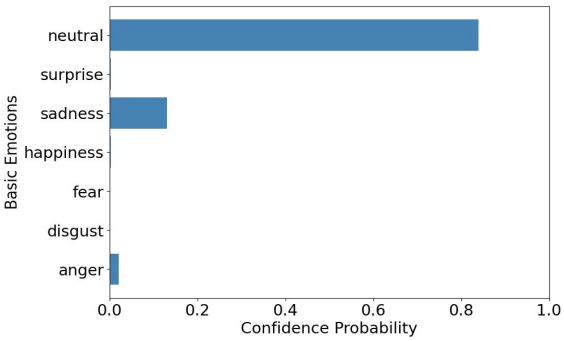
To sum up the evaluation on disentangle property provided by StyleGANs [37], the methodology [105], and EmoGANs3, the last two provides the best maintenance of facial identity in generation process. However, the transformed image by [105] expresses the neutral expressions rather than mixed facial expressions. In contrast, the generated images by EmoGANs3 shows the mixed facial expressions based on the recognition results by Py-feat.

(a) Transformed Image by methodology[105]



(b) Recognized emotion in (a)

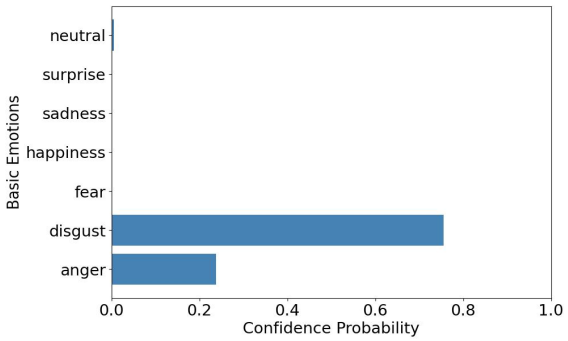
(c) Transformed Image by methodology[105]



(d) Recognized emotion in (c)

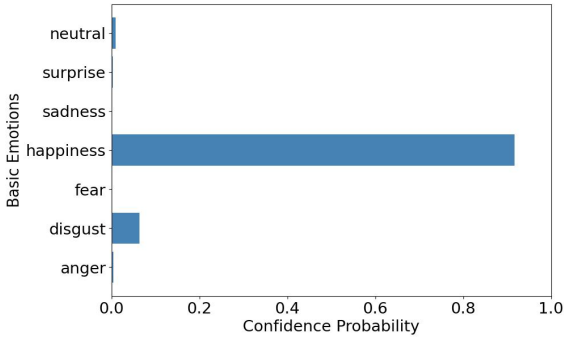
Figure 5.20: Confidence probability score of synthesized images transformed in the latent space recognized by PyFeat Library[16].

(a) Transformed Image by StyleGANs[37]



(b) Recognized emotion in (a)

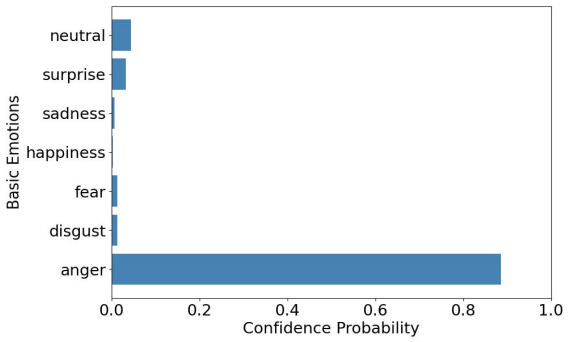
(c) Transformed Image by StyleGANs[37]



(d) Recognized emotion in (c)

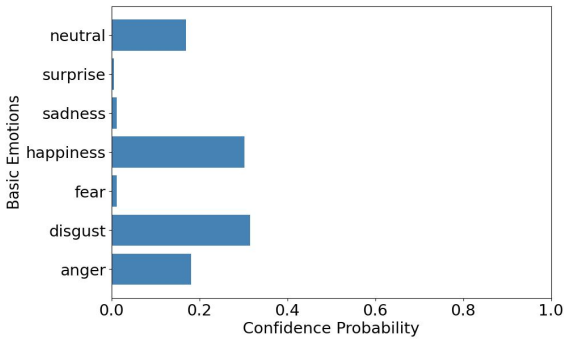
Figure 5.21: Confidence probability score of synthesized images transformed in the latent space recognized by PyFeat Library[16].

(a) Generated Image by EmoGANs3



(b) Recognized emotion in (a)

(c) Generated Image by EmoGANs3



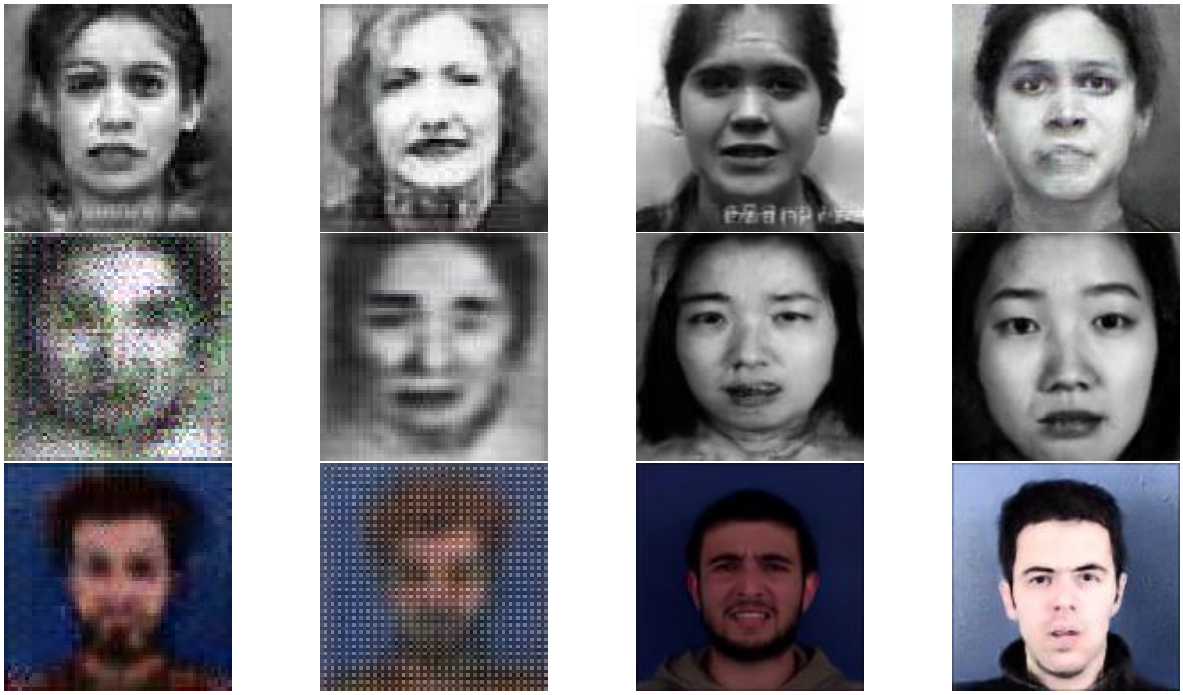
(d) Recognized emotion in (c)

Figure 5.22: Confidence probability score of synthesized images transformed in the latent space recognized by PyFeat Library[16].

5.4 Summary on synthesized images by all EmoGANs models

We summarized the properties of proposed EmoGANs models [91, 92, 93].

- The first two models produce noisy images, while the second model has less noisy data but it is still infeasible to be assessed by the naked eye. The last two models can generate higher output resolution and facial expressions are clear in the generated image (See in Figure.5.23).
- As discussed in the previous sections to quantitatively assess the image quality, inception score, and FID score by EmoGANs3 model provides the best score in MUG and JAFFE. Based on the BRISQE score, its score is best in CK+ compared with other EmoGANs models. From those results, we can conclude that including the facial identity information in the image generation process can improve the image quality as only EmoGANs3 considers keeping facial information. Besides, the image generation process can be controlled by label vectors and encoded facial representation.
- As discussed in the previous chapter, generators of the first two models perform both feature extraction from image pair and image generation and require longer training time for convergence. However, the discriminator convergences after a few training epochs, resulting the unstable training. In EmoGANs2, these two tasks are separated by the different models and are able to construct high-resolution images.
- All properties are summarized in Table.5.6.



(a) EmoGANs (64×64) (b) EmoGANs1 (64×64) (c) EmoGANs2 (64×64) (d) EmoGANs3 (128×128)

Figure 5.23: Example of generated images by each EmoGANs for CK+ (upper row), JAFFE (middle row), MUG(lower row).

Table 5.6: Summary of EmoGANs models proposed in this dissertation

Property	EmoGANs[91]	EmoGANs1[92]	EmoGANs2[93]	EmoGANs3
Image quality	Noisy	Noisy	Clean	Clean
Output resolution	64×64	64×64	128×128	128×128
Training Stability	No	No	Yes	Yes
Control on generation	No	No	No	Yes
Non-linear transformation	Yes	Yes	Yes	Yes

Chapter 6

Emotion Estimation

6.1 Estimation of Basic Emotions through Basic Facial Expressions Features

The basic framework for an automatic emotion recognition model includes two main parts: feature extraction and classification. The goodness of the features extracted by conventional feature extraction methods, such as the Gabor filter and local binary pattern, relies on meticulous data processing. Following feature extraction, several classification methods, such as support vector machines, k-nearest neighbors, and decision trees, are employed to categorize the features into different groups. The performance of the classifiers also depends on the goodness of the extracted features or the degree of their discriminative power. However, conventional feature extraction methods are sensitive to noise and various factors, such as lighting conditions, head pose, and occlusion. In addition, the study [109] reports that manual detection of facial attributes or landmarks registration is required during the extraction process. Furthermore, since feature extraction and classification are independent phases, it can be challenging to improve the recognition performance.

While the challenges of the conventional framework can be addressed by using deep learning-based models, it should be noted that their generalization power is often dependent on the quantity of available training data. Unfortunately, such data is not always readily available. The new training technique, named transfer learning had widely discussed to tackle this problem in [85, 110, 111]. As discussed in the previous chapter, transfer learning is a technique in which previously acquired knowledge is applied to new tasks. This involves adjusting previously learned knowledge to suit the new domain and often fine-tuning the final layers of the model for the new task. By sharing low-level features learned from the previous task, transfer learning can reduce the amount of training data required for the new task, thus reducing training time and computational resources. It often leads to improved performance, especially when the new dataset is small or similar to the original dataset. Since we used benchmark facial expressions datasets that have a limited number of images, transfer learning is well-suited to our problem.

Figure.6.1 summarize the training strategies employed in fine-tuning the pre-trained models for transfer learning in the emotion recognition task[112]. Additionally, it includes an analysis of training a completely new model with appropriate layers, as well as training pre-trained models with deep structural layers that have millions of parameters starting from random weights. Since we prefer the models to have attention solely on the face region, faces are detected, extracted, and masked before the fine-tuning process. The output after the pre-processing phase is shown in Figure.6.2. In the next section, we discuss the choice of pre-trained models, their unique structures, and training configurations.

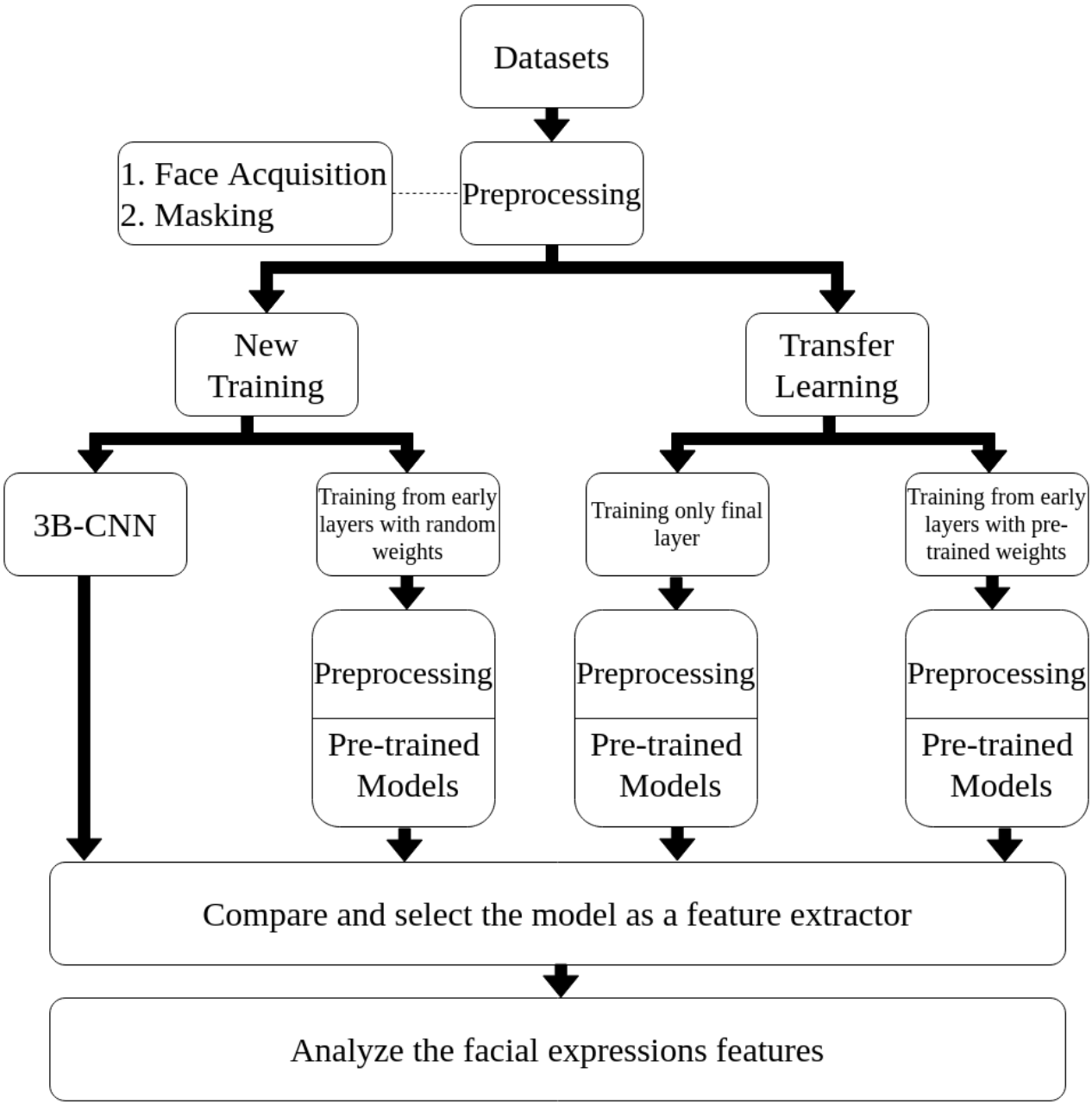


Figure 6.1: Overview of emotion recognition model with transfer learning using various Training Strategies for performance comparison

6.1.1 Extraction of Facial Expression Features

Transfer learning technique has been employed in many studies, as reported in [84, 113, 114]. The choice of pre-trained deep models also differs and varies. The current study selects the common networks often used for emotion recognition tasks in the transfer learning literature. They are

1. **VGG16 Face** [87]: It is the deep convolutional network consisting of five convolution blocks, each containing convolution and pooling layers, proposed by the Visual Geometry Group (VGG) of Oxford University in 2015. The model was trained on 2.6 million face images to recognize 2,622 identities.

Figure 6.2: An example output of data pre-processing phase

2. **ResNet 50**[115]: ResNet 50 is a type of deep convolutional neural network with 175 layers that employs skip connections to overcome the vanishing gradient problem that can obstruct back-propagation and learning. The model was trained on a dataset of 1.28 million images from 1000 classes.

3. **MobileNet**[116]: MobileNet is also a convolutional network with a lightweight architecture suitable for mobile devices, incorporating inverted residuals to connect the input and output of residual blocks. The model was trained on the ImageNet dataset.

4. **Inception V3**[107]: It composes of 48 convolution layers and has a small convolution before a large convolution for dimension reduction. It was also trained with ImageNet data.

5. **Inception ResNet V2**[117]: It combines the structure of inception and residual networks to prevent the degrading problem. It was trained with the ImageNet dataset.

6. **EfficientNet B0**[118]: It is also a family of convolution networks and efficiently expands the structure using the compound scaling method. It was trained with the ImageNet dataset.

6.1.1.1 Networks Training With Different Strategies

Compared to facial expressions classification, the face recognition domain had advanced and a variety of large-scale face datasets is available. The structure of pre-trained models of the face domain is either a type of widely spread or deeper layers with millions of parameters. Therefore, in transfer learning, mostly the final classification layer of pre-trained models is often fine-tuned to the new tasks using weights from the previous tasks. When such deep models are trained with a small dataset without the previously learned weights, they might not generalize well on the unseen data and lead to memorizing the training data. Therefore, in this current study, we consider three training strategies. They are

1. Training from early layers with *random weights*: The structures of the pre-trained models are loaded and trained for emotion recognition with the randomly assigned weights as the initial weights. The weights from the previous face recognition are not loaded and discarded.

2. Training from early layers with *pre-trained weights* from face recognition task: The initial weights of pre-trained models are loaded from the previous learning. The models are re-trained through the earlier layers.

3. Training only the last classification layer for emotion recognition: It is the common tactic used in fine-tuning the pre-trained model. Only the final layer is trained to recognize the new recognition task.

Training Configurations During training, early stopping of the learning process and data augmentation are used to prevent the deep models from over-fitting, where the models cannot generalize well on unseen data. Complete configurations can be seen in Table. 6.1.

- Early Stopping: 'Yes' indicates 'early stopping' is used in the training process to prevent the model from over-fitting. 'No' means the opposite.
- Data Augmentation: 'Yes' indicates training data are augmented by flipping from left to right and vice versa. 'No' means the opposite.

Table 6.1: Training configurations for transfer learning and new training for emotion recognition

Transfer Learning			
Setting Number	Early Stopping	Data Augmentation	Training only the last classification layer (or) Training from the earliest layers with *pre-trained weights*
1-1 1-2	Yes No	No Yes	Training only the last classification layer
1-3 1-4	Yes No	No Yes	Training from the earliest layer
New Training			
Setting Number	Early Stopping	Data Augmentation	Training from the earliest layers with *initial random weights*
2-1 2-2	Yes No	No Yes	Yes Yes

Recognition Performance The accuracy metric, defined in Equation 3.6, is used to evaluate the recognition performance. All models are trained using the Adam optimizer with a learning rate of 1e-03. Except for early stopping, all models are trained for 500 iterations. The models are assessed through five cross-validation sets. Tables 6.2 and 6.3 summarize the results for transfer learning and new training, respectively.

Analyzing the cross-validated results for transfer learning in Table 6.2, we observe that training only the last classification layer yields comparable recognition performance and operates efficiently on the small dataset. In comparison to the entire model, the last layer contains fewer weights, resulting in saved training time.

For the new training results in Table 6.3, the CNN model with three convolution layers achieves the highest performance among the others. This is because the weights from the CNN are relatively small, and the structure of the other models is only loaded and trained from initial random weights.

The models are evaluated on different test segmentations and run for 5 trials. The results are presented in Table 6.4. From the table, we can observe that the 75-25 data partition provides the comparative results compared to the other partitions. Among the four settings in transfer learning, setting 4 achieves the highest accuracy score of approximately 93% using

the EfficientNet model. The ResNet V2 model also yields similar results across 5 trials under the 75-25 data split. Therefore, these two models, EfficientNet and ResNet V2, from setting 1-4 are selected for further analysis in the transfer learning technique. Based on the cross-validated results for new training, the 3B-CNN model achieves the highest score. Additionally, EfficientNet performs well with augmented data when given sufficient training. Consequently, these two models, 3B-CNN and EfficientNet, from setting 2-2, are selected for further study as a result of the new training.

Table 6.2: Average accuracy of each model evaluated over 5-fold cross-validation sets for transfer learning. Accuracy is shown in percentage (±standard deviation). Maximum score is highlighted in bold font.

Models	Settings Number (**Transfer Learning**)			
	Setting 1-1	Setting 1-2	Setting 1-3	Setting 1-4
VGG16 Face	82.4 ± 5.7	**90.6± 2.5**	23.2±2.8	22.6± 4.7
ResNet 50	82.7± 2.1	85.0±3.4	31.3±18.7	85.3± 5.1
MobileNet	87.3±2.7	79.8± 2.8	58.1±17.9	67.1± 15.9
Inception ResNet V2	**87.9 ± 2.5**	86.3± 2.1	32.3±28.8	81.2± 12.3
Inception V3	84.4±2.6	82.4± 4.0	22.8± 6.3	61.6± 14.5
EfficientNet B0	86.0± 3.6	89.9± 2.7	**76.0± 13.5**	**92.8±6.0**

Table 6.3: Average accuracy of each model evaluated over 5-fold cross-validation sets for new training. Accuracy is shown in percentage (± standard deviation). Maximum score is highlighted in bold font.

Models	Settings Number (**New Training**)	
	Setting 2-1	Setting 2-2
3B-CNN	**74.0 ± 6.6**	**78.5 ± 4.1**
VGG16 Face	26.4± 5.1	22.5± 5.0
ResNet 50	30.0± 8.8	39.1 ± 5.1
MobileNet	17.6± 5.1	46.0± 12.8
Inception ResNet V2	31.2± 17.1	77.1 ± 10.2
Inception V3	20.6 ± 8.7	72.3 ± 5.3
EfficientNet B0	33.6±20.3	73.0± 12.9

Table 6.4: Average accuracy of each model tested across different data segmentation for 5 trials in each settings. Accuracy is shown in percentage (± standard deviation). Maximum score is highlighted in bold font.

Models	Train Test Split (train set, test set)				
	95%, 5%	90%, 10%	75%, 25%	50%, 50%	25%, 75%
Setting 1-1					
VGG16 Face	90.0 ± 5.0	83.9 ± 2.9	87.0 ± 3.7	80.7 ± 3.0	75.3 ± 3.8
ResNet 50	97.5 ± 5.0	85.8 ± 3.9	87.8 ± 1.9	78.1 ± 3.1	76.5 ± 1.0
MobileNet	88.8 ± 2.5	74.8 ± 3.8	79.0 ± 1.7	78.6 ± 1.2	73.2 ± 2.4
Inception ResNet V2	**97.5 ± 3.1**	**94.2 ± 4.3**	**91.4 ± 2.3**	**84.6 ± 0.7**	**79.2 ± 2.3**
Inception V3	83.8 ± 3.1	77.4 ± 4.6	84.4 ± 0.8	77.7 ± 2.6	71.5 ± 2.7
EfficientNet B0	95.0 ± 2.5	83.9 ± 4.1	88.1 ± 2.1	82.0 ± 2.2	73.6 ± 4.5
Setting 1-2					
VGG16 Face	92.5 ± 6.1	88.4 ± 3.3	88.3 ± 1.2	83.6 ± 5.3	76.0 ± 4.2
ResNet 50	**98.8 ± 2.5**	84.5 ± 3.2	84.4 ± 3.0	77.0 ± 2.7	73.2 ± 2.1
MobileNet	86.3 ± 2.5	74.2 ± 5.8	80.8 ± 3.8	76.1 ± 2.7	73.0 ± 1.1
Inception ResNet V2	96.6 ± 5.0	**90.3 ± 5.4**	88.1± 2.9	83.8 ± 2.6	**80.6 ± 2.0**
Inception V3	81.3 ± 0.0	79.4 ± 6.0	81.6 ± 5.6	78.6 ± 2.6	71.5 ± 2.1
EfficientNet B0	95.0 ± 2.5	87.7 ± 2.4	**92.5 ± 1.9**	**84.8 ± 1.1**	77.9 ± 1.2
Setting 1-3					
VGG16 Face	37.5 ± 31.4	38.7 ± 23.7	34.6 ± 22.3	24.8 ± 1.0	25.0 ± 5.6
ResNet 50	30.0 ± 17.0	24.5 ± 12.7	39.0 ± 21.5	32.7 ± 19.2	29.9 ± 21.7
MobileNet	31.3 ± 31.1	44.5 ± 11.8	45.2 ± 18.4	43.4 ± 15.0	43.5 ± 18.5
Inception ResNet V2	65.0 ± 28.9	35.5 ± 15.5	39.7 ± 27.5	39.6 ± 24.8	47.1 ±17.3
Inception V3	18.8 ± 11.2	23.2 ± 9.4	40.3 ± 15.5	29.4 ± 17.3	35.4 ± 9.5
EfficientNet B0	**83.8 ± 19.2**	**81.3 ± 11.2**	**86.0 ± 5.5**	**84.0 ± 2.9**	**70.4 ± 8.5**
Setting 1-4					
VGG16 Face	6.3 ± 0.0	19.4 ± 0.0	23.4 ± 0.0	25.3 ± 0.0	25.5 ± 0.0
ResNet 50	82.5 ± 32.0	**91.0 ± 2.4**	**91.2 ± 3.0**	71.4 ± 14.8	73.1 ± 6.2)
MobileNet	63.8 ± 20.7	66.5 ± 18.8	79.0 ± 14.9	54.9 ± 27.1	42.3 ± 19.4
Inception ResNet V2	78.8 ± 17.5	66.5 ± 20.4	77.7 ± 18.1	73.9 ± 9.9	71.4 ± 5.1
Inception V3	80.0 ± 4.7	36.1 ± 15.3	75.6 ± 11.2	49.4 ± 5.6	53.7 ± 16.9
EfficientNet B0	**91.3 ± 8.5**	90.3 ± 4.1	90.1 ± 7.5	**85.6 ± 7.7**	**85.0 ± 4.5**
Setting 2-1					
3B-CNN	**77.5 ± 5.0**	**77.4 ± 6.5**	**77.9 ± 7.9**	**69.7± 3.3**	**55.8± 4.5**
VGG16 Face	6.3± 0.0	19.4± 0.0	23.4±0.0	25.3± 0.0	25.5± 0.0
ResNet 50	31.3±20.5	30.3± 9.7	42.3 ± 3.2	32.9± 9.2	21.7± 6.0

Continued on next page

Table 6.4 – continued from previous page

Models	Train Test Split (train set, test set)				
	95%, 5%	90%, 10%	75%, 25%	50%, 50%	25%, 75%
MobileNet	18.8±0.0	15.5± 1.3	19.7 ± 3.4	14.7 ±2.1	16.1 ± 1.0
Inception ResNet V2	26.3±14.5	26.5± 5.5	24.4± 6.4	19.4± 4.4	18.2±0.0
Inception V3	12.5± 7.9	19.4 ± 6.5	17.1± 5.2	19.4± 5.2	20.7 ± 4.2
EfficientNet B0	21.3 ± 12.2	22.6±9.8	25.2±9.8	15.7± 2.5	19.7± 3.1
Setting 2-2					
3B-CNN	78.8 ± 8.5	74.2 ± 4.1	71.4± 3.2	69.7± 2.0	**63.5± 3.8**
VGG16 Face	6.3± 0.0	19.4± 0.0	23.4± 0.0	25.3± 0.0	25.5± 0.0
ResNet 50	31.3± 22.0	47.1± 7.5	50.7± 15.6	55.8± 10.0	42.9± 5.2
MobileNet	33.8± 11.6	38.7± 8.7	33.3± 9.3	24.8± 9.3	23.4± 0.0
Inception ResNet V2	65.0± 35.9	72.3± 13.0	74.6± 7.1	60.0± 15.5	45.6± 14.3
Inception V3	47.5± 17.0	63.2± 20.2	68.1± 21.2	49.4± 17.9	34.9± 15.3
EfficientNet B0	**86.3± 7.3**	**82.6± 7.5**	**83.6± 3.6**	**73.5± 8.2**	58.4± 4.1

6.1.2 Analysis of Facial Expression Features

The previous section selects the best models to work on emotion recognition. These selected models are used as functions to extract features related to facial expressions in a multi-dimensional feature space. Among these features, some information for the emotion class might be redundant, and analyzing the hidden patterns can be tedious in high-dimensional features. Not all attributes in the features are useful for output emotion class. For example, if a particular attribute of features has little variance or no variance (i.e., constant), this attribute does not have impact on output variable. Therefore, feature attributes with maximum variance is desired.

Principal Component Analysis (PCA) is employed for feature analysis. PCA discovers the features that have maximum variance across among emotion classes as much as feasible using eigenvectors, eigenvalues and covariance. Suppose that v_n^j is the n^{th} feature sample with attribute j where $n \leq N$ and $j \leq M$. N is the total number of feature samples and M is the dimension of the feature space. It can be represented in matrix form as follows.

$$\mathbf{V} = \begin{bmatrix} v_1^{(1)} & v_1^{(2)} & \dots & v_1^{(j)} \\ v_2^{(1)} & v_2^{(2)} & \dots & v_2^{(j)} \\ \vdots & \vdots & \ddots & \vdots \\ v_n^{(1)} & v_n^{(2)} & \dots & v_n^{(j)} \end{bmatrix} \tag{6.1}$$

where $\mathbf{V}$ is the feature matrix with size $M \times N$.

Then, covariance is constructed to measures how much each attribute is spread out from the center mass or mean with respect to each other. The covariance matrix can be defined as

follows.

$$\mathbf{C} = \begin{bmatrix} cov(v^{(1)}, v^{(1)}) & cov(v^{(1)}, v^{(2)}) & \dots & cov(v^{(1)}, v^{(j)}) \\ cov(v^{(2)}, v^{(1)}) & cov(v^{(2)}, v^{(2)}) & \dots & cov(v^{(2)}, v^{(j)}) \\ \vdots & \vdots & \ddots & \vdots \\ cov(v^{(j)}, v^{(1)}) & cov(v^{(j)}, v^{(2)}) & \dots & cov(v^{(j)}, v^{(j)}) \end{bmatrix} \tag{6.2}$$

where $\mathbf{C}$ is the symmetrical diagonal covariance matrix with size $M \times M$. Diagonal is the variance of each attribute.

Covariance among two attributes can be calculated as follows. Suppose that two attributes are defined as x and y. The covariance between x and y becomes

$$cov(x, y) = \frac{\sum(x_n - \bar{x})(y_n - \bar{y})}{N - 1} \tag{6.3}$$

where $\bar{x}$ and $\bar{y}$ are the center masses or means over all samples for the corresponding dimensions x and y. N is the total number of feature samples. n is the sample index which is $n \leq N$.

The square covariance matrix is decomposed into eigenvectors and eigenvalues to create the principal component space, where eigenvectors serve as the axes of the new space and eigenvalues carry the amount of variance that the eigenvectors have. The eigen decomposition can be written as follows.

$$\begin{bmatrix} cov(v^{(1)}, v^{(1)}) & cov(v^{(1)}, v^{(2)}) & \dots & cov(v^{(1)}, v^{(j)}) \\ cov(v^{(2)}, v^{(1)}) & cov(v^{(2)}, v^{(2)}) & \dots & cov(v^{(2)}, v^{(j)}) \\ \vdots & \vdots & \ddots & \vdots \\ cov(v^{(j)}, v^{(1)}) & cov(v^{(j)}, v^{(2)}) & \dots & cov(v^{(j)}, v^{(j)}) \end{bmatrix} \begin{bmatrix} u^{(1)} \\ u^{(2)} \\ \vdots \\ u^{(j)} \end{bmatrix} = \lambda \begin{bmatrix} u^{(1)} \\ u^{(2)} \\ \vdots \\ u^{(j)} \end{bmatrix} \tag{6.4}$$

where $\mathbf{U}$ is the eigenvector that is $[u^{(1)}u^{(2)} \dots u^{(j)}]^{\mathrm{T}}$ and λ is the eigenvalue.

It can be solved as follows.

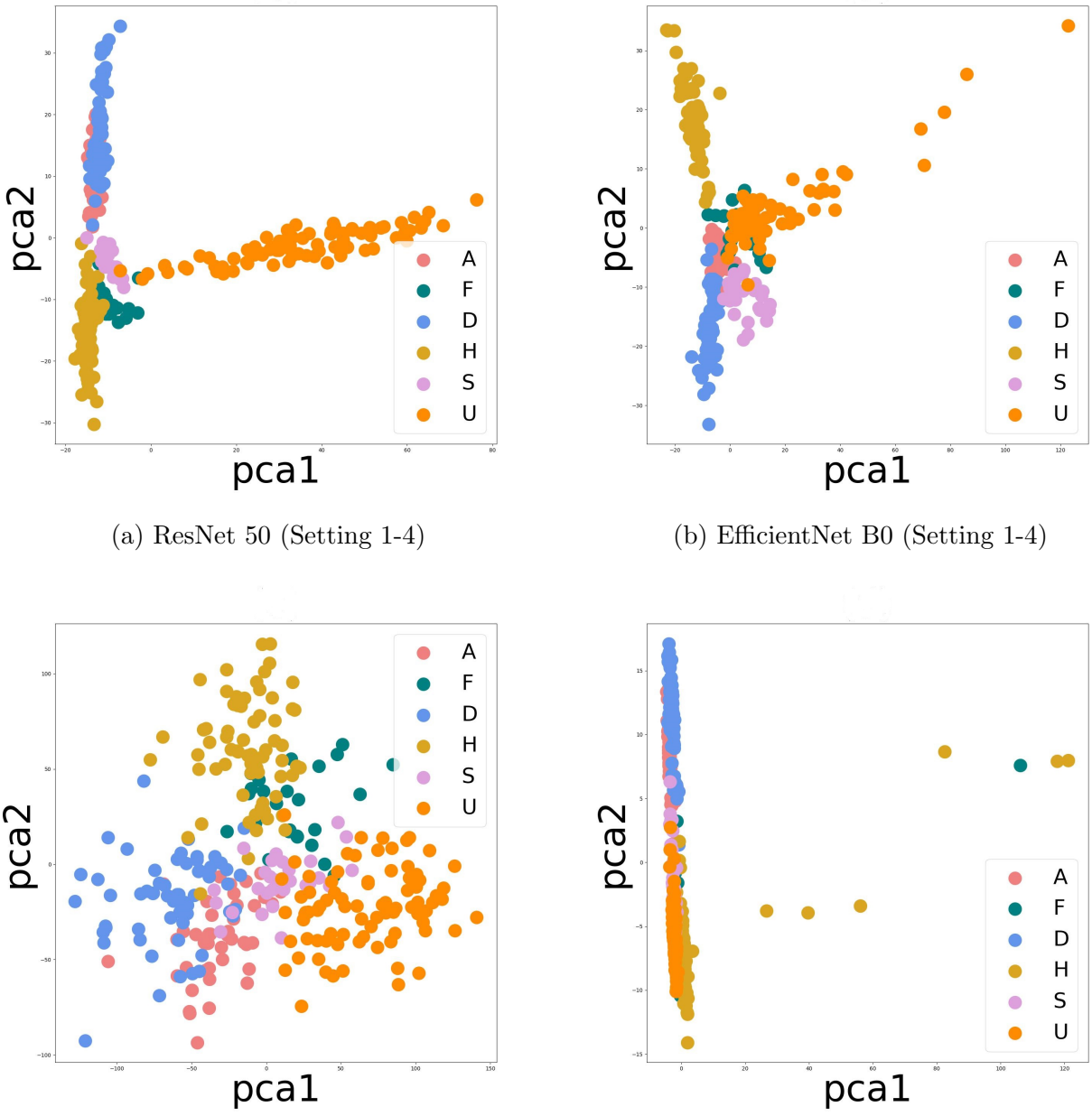
$$(\mathbf{C} - \lambda\mathbf{I})\mathbf{U} = 0 \tag{6.5}$$

where $\mathbf{I}$ is the identity matrix. $\mathbf{C}$ is the covariance matrix. $\mathbf{U}$ is the eigenvectors with corresponding eigenvalues λ.

After obtaining eigenvectors and eigenvalues, PCA creates the projection space where eigenvectors become the axes with maximum variances, and their corresponding eigenvalues maximize the variance. In PCA, the principal components represent the directions of the axes. The first principal component carries the highest amount of variance. The subsequent components are orthogonal to the previous ones and account for the subsequent maximum variances. The number of components equals to the number of the feature dimensions, and they are ordered according to the maximum variance. Consequently, the lower components carry less variance, and the dimensions related to those components can be removed as they have little impact on the output class.

6.1.3 Discussion

In the previous section, the candidate models are chosen depending on their recognition performance in terms of accuracy over the validated set and test set. The candidate models are EfficientNet B0 and ResNet 50 from setting 1-2 for transfer learning and 3B-CNN and EfficientNet B0 from setting 2-2 for new training. Suppose that those models are mapping functions M which projects the input $\mathbf{x}$ from high-dimensional image space onto features space with fewer dimensions for emotion recognition that is $\mathbf{v} = M(\mathbf{x})$. The expressions features $\mathbf{v}$ is analyzed



(a) ResNet 50 (Setting 1-4)

(b) EfficientNet B0 (Setting 1-4)

(c) 3B-CNN (Setting 2-2)

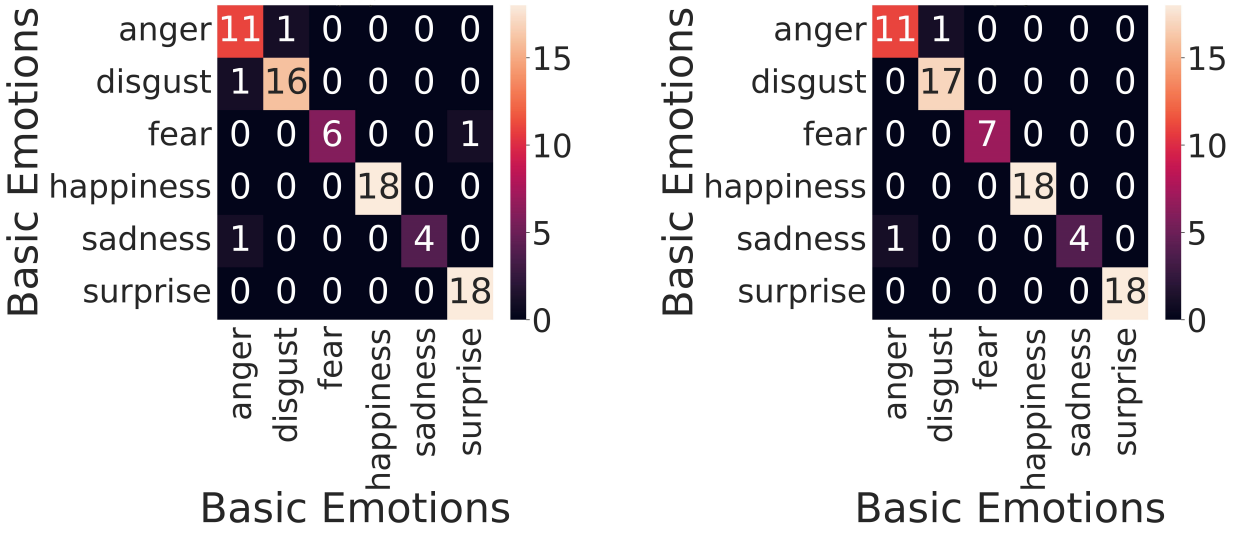
(d) EfficientNet B0 (Setting 2-2)

Figure 6.3: Visualization of projected feature points in the principal component space with the first two components (pca1, pca2). ● (A), ● (F), ● (D), ● (H), ● (S), ● (U) represent 'anger', 'fear', 'happiness', 'sadness', and 'surprise', respectively.

using PCA to determine the goodness of the features. We hypothesized that if facial expressions features possess strong discrimination across classes, they should be located far from each other to make the easier classification.

The extracted features are analyzed using PCA and projected onto two-dimensional space. Figure.6.3 visualize the features sample of the principal component space for all classes. Features extracted by the models from new training exist close to each other across different classes, indicating they do not have strong discrimination in the facial expressions feature space made by the models. In contrast, feature samples extracted by the models from transfer learning are

located close to each other among intra-classes.



(a) ResNet 50 (Setting 1-4)

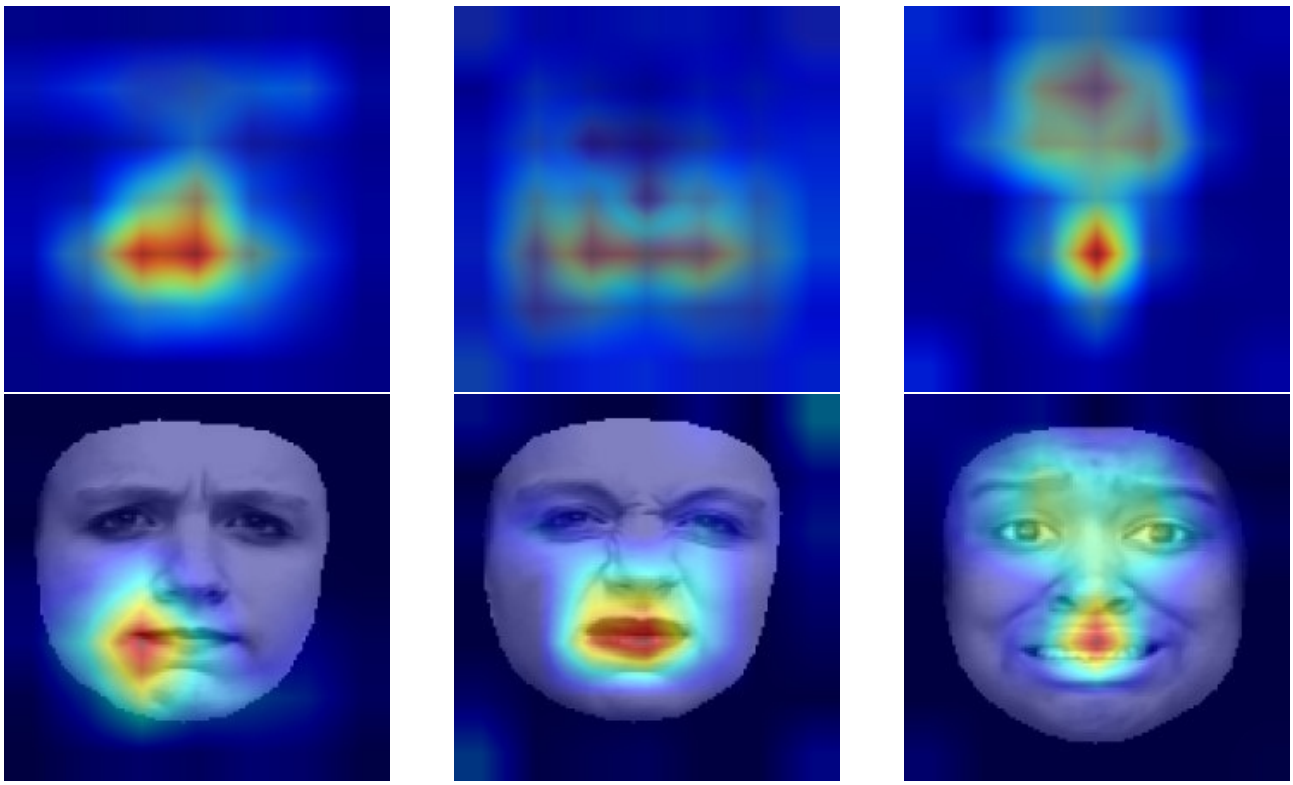
(b) EfficientNet B0 (Setting 1-4)

Figure 6.4: Performance of emotion recognition models with transfer learning technique (confusion matrix over a test set)

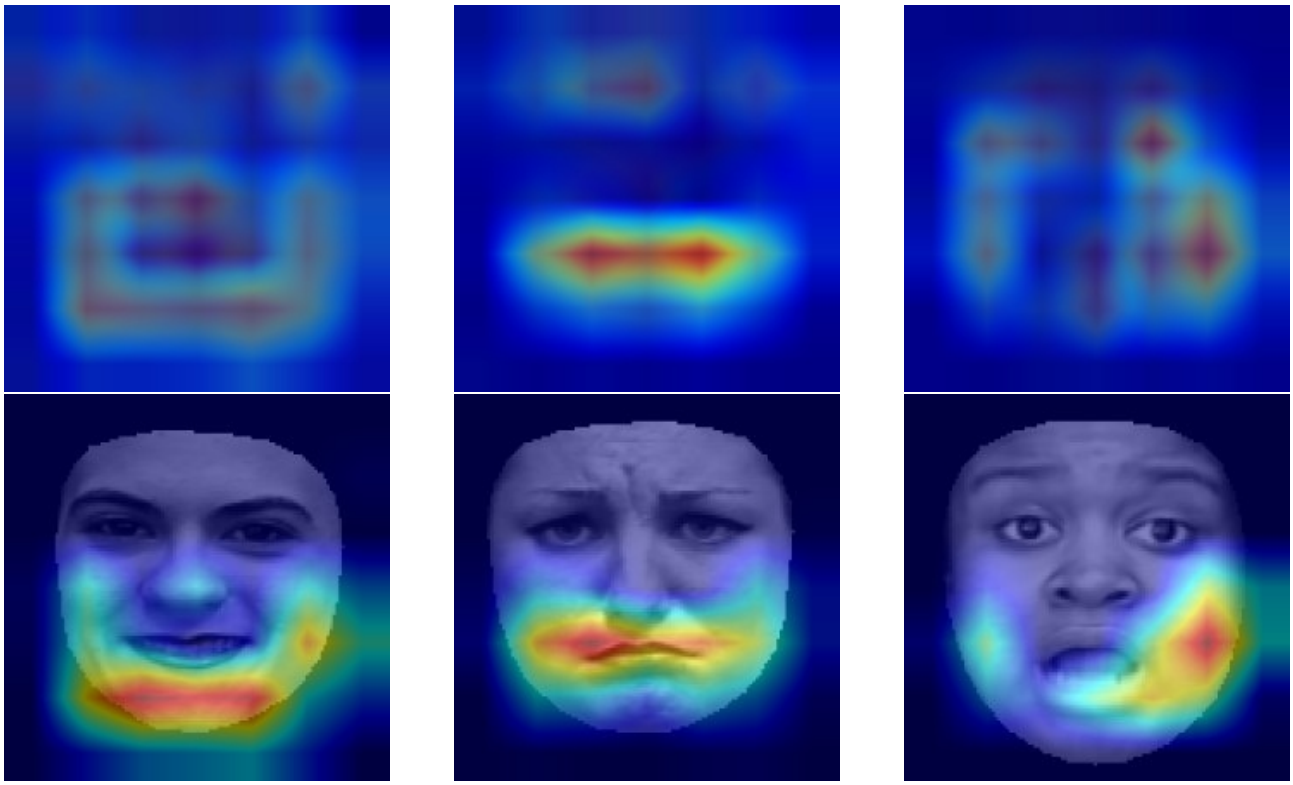
Based on the result from ResNet 50, features representing the surprise class have strong discriminative power among inter-classes as they are solely located far from others. In two-dimensional space, features representing disgust and anger are overlapped in the space, indicating that the model often misclassifies the anger samples with disgust and vice versa. Similarly, fear features are also close to the border of surprise and happiness classes. Therefore, they can be misclassified into those classes by the model. A similar interpretation goes to sadness features. We can observe that the analysis result matches with the model performance in Figure.6.4a.

Analyzing the results of the EfficientNet B0 model from transfer learning in the principal component space, it is evident that the features extracted by this model possess stronger discriminative power in comparison to those obtained by ResNet 50 under the same setting for surprise, happiness, and disgust. Due to their positioning in the projection space, there is a possibility of misrecognition between anger features and both disgust and sadness and vice versa. The interpretation matches with the model performance across classes for the test set in Figure.6.4b.

Since the features extracted from the EfficientNet B0 demonstrate stronger discrimination against other classes, further investigation is needed to understand what the model captures from the input and determine which facial expressions are crucial for classification. We adopt the method proposed by the study [119] to visualize the feature maps filtered by the model. Results are shown in Figure.6.5. From the result, we can observe that the model focuses on the jaw, chin, and philtrum (the area between the nostrils and the upper lip) for happy facial expressions. It focuses on the jaw and eyes for surprise and the perioral region (central part of the lower third of the face) for disgust. It emphasizes on glabella (skin area between eyebrows and above the nose) and the mouth for sadness. From those highlighted illustrations, we can perceive that the model observes the important and useful facial region for emotion recognition.



(a) anger (b) disgust (c) fear

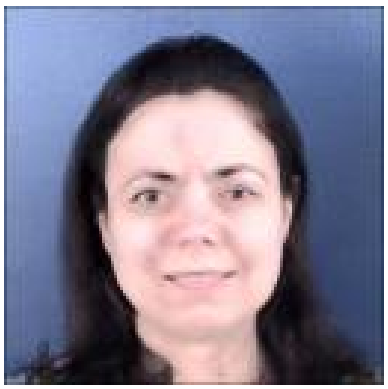


(d) happiness (e) sadness (f) surprise

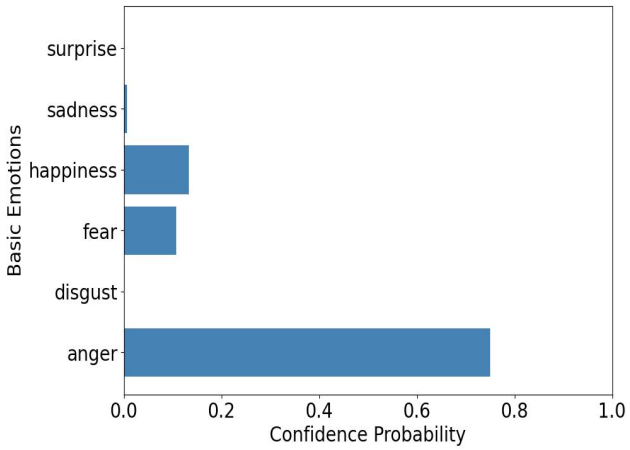
Figure 6.5: Visualization of activated facial components while recognition the emotion by EfficientNet B0 model from setting 1-4. Upper row in each class represents average activation over all samples of the respective class. Lower row represents the output of a particular sample with activated faical components.

6.2 Estimation of Mixed Emotions through Synthesized Images by EmoGANs3

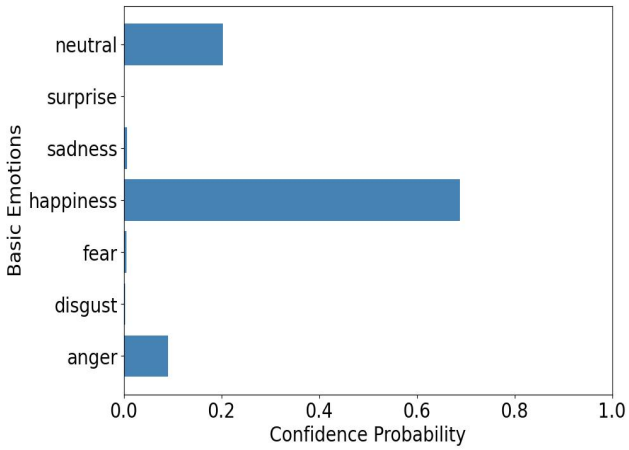
Based on the discussion on evaluation results in the previous chapter, EmoGANs3 can synthesize qualitative images compared to other EmoGANs models. An example-generated image is given in Figure 6.6(a). From the discussion of basic emotion estimation in the previous section, we concluded that Efficientnet B0 provides the most discriminative features related to facial expressions. Therefore, we apply this model to estimate the emotions in the example-generated image. According to the result in Figure 6.6(b), the model returns 'anger' as the most probable emotion class for the example image. We also apply the Py-Feat Python library to recognize the emotion in the example image, which returns 'happiness' as the most detected emotion for the image. However, the original labels used to synthesize the images are 'happiness' and 'sadness'. Based on this result, we can conclude that a specialized facial expression recognition system is required to estimate the generated mixed facial expression images.



(a) An example generated image



(b) Efficient Net B0



(c) Py-feat

Figure 6.6: (a) An example image synthesized by the proposed EmoGANs3 model for the MUG dataset. The given mixed labels for example are 'happiness' and 'sadness' such as [0,0,0,1,1,0] in vector form. (b) and (c) are confidence probability scores predicted by Efficientnet B0 [112] and Py-feat[16].

6.2.1 Network configurations of multi-labels facial expressions recognition model

The multi-label formulation is suitable to estimate the mixed emotions estimation system, as each generated image contains two emotion labels out of six basic classes such as 'happiness' and 'sadness' in the example case. The network configuration for multi-emotion label estimation is given in Figure.6.7. Convolutional layers are applied to extract the facial expression's specific features and the max pooling layer is used to reduce spatial dimension size after convolution operation. Since the model is expected to output multiple one's values, the sigmoid activation is applied instead of softmax activation to obtain the multiple emotion labels.

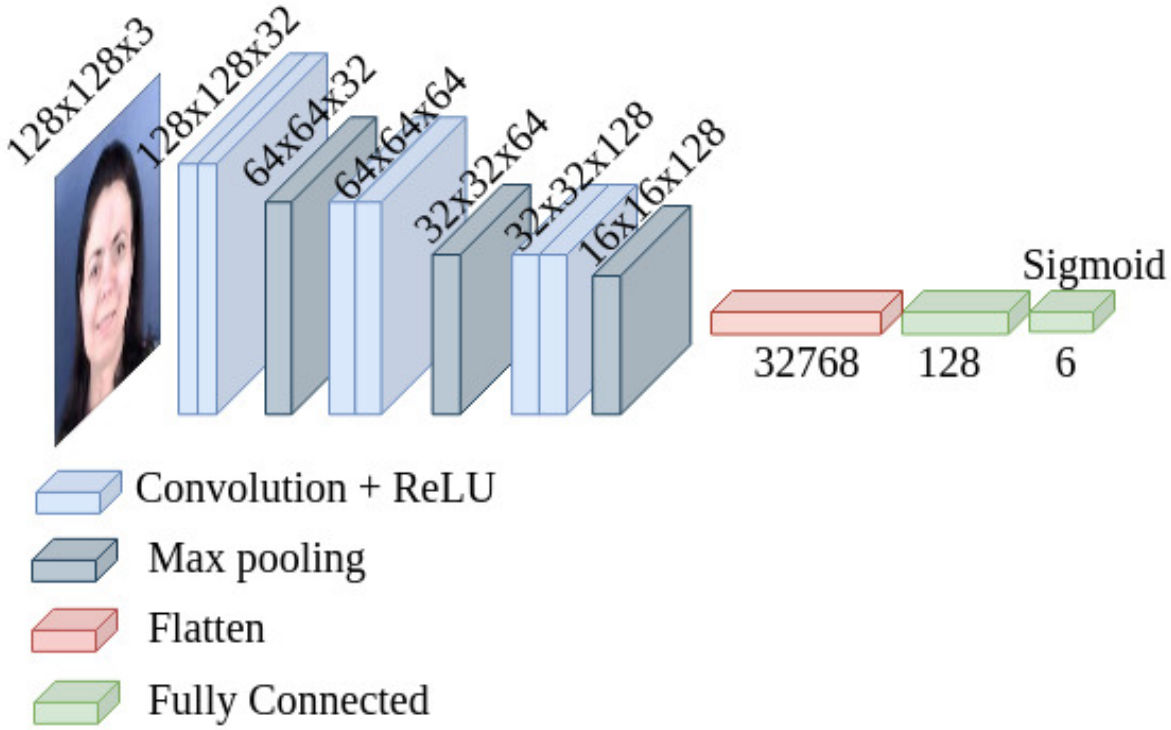


Figure 6.7: Network configurations to estimate multi-emotions labels of generated images (multi-labels CNN model)

Since the model produces multiple emotion labels, accuracy is not a good metric to evaluate multiple-label classification. For example, when the true label for a particular generated image is 'happiness' and 'sadness' as in the example image, the vector form of the true label becomes [0, 0, 0, 1, 1, 0], where each dimension represents each basic emotion such as anger, disgust, fear, happiness, sadness, and surprise, respectively. Suppose that the model predicts 'happiness' and 'anger' as its emotions, resulting in [1, 0, 0, 1, 0, 0]. From these two vectors, we can see that the model can predict 'happiness' correctly but misrecognizes 'sadness' as 'anger'. If the accuracy metric is used in the evaluation, it will not consider one of the correct classes and strictly reject the entire prediction.

Therefore, f-beta (f_β) metric is used to evaluate the model that uses the mean of precision and recall weighted by parameter β. Precision evaluates the model based on the degree of goodness for predicting the positive class, while recall measures the model's ability to identify the positive classes. Since it is a multiple-label classification problem, a specific emotion class

from the true label is defined as the positive class, while the rest are defined as negative classes using a one-versus-the-rest (OVS) approach. The f_β score ranges from 1 for the optimal value and 0 for the minimum. The equation for the f-beta (f_β) metric can be defined as follows.

$$f_\beta = (1 + \beta^2)\frac{precision.recall}{(\beta^2.precision + recall)} \tag{6.6}$$

where f_β is the f-beta metric or f-measure which uses weighted parameter β.

6.2.2 Recognition Performance

The model shown in Figure 6.7 is trained to estimate the multiple labels of generated images produced by the EmoGANs3 model using mini-batch stochastic gradient descent with a size of 128. Before training, the images are normalized to the range [0, 1]. A total of 15,000 generated images are randomly produced by EmoGANs3 and then split into training and test sets using a 70-30 ratio.

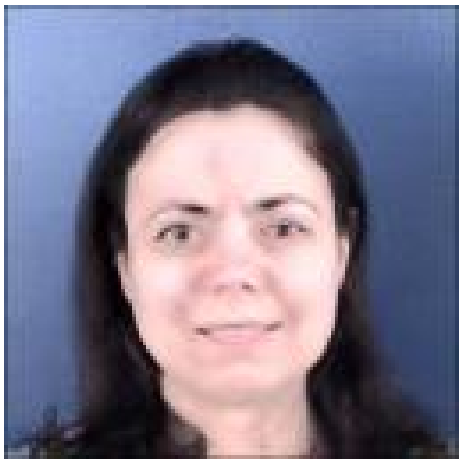
We compared the performance of our multi-label emotion estimation model with two baseline models. Previously, the EfficientNet B0 model was the best to estimate the basic emotions based on the accuracy metric. The model was trained with the generated images by EmoGANs3 and evaluated its performance over the test set. Additionally, we applied the same network configuration but used the generated images from StyleGANs. The model is again evaluated over the same test set.

Table 6.5: Evaluation for multiple labels estimation model over test set. f_β is the f-beta metric and EMR is the exact match ratio that calculates the ratio of the predicted labels as identical as true labels

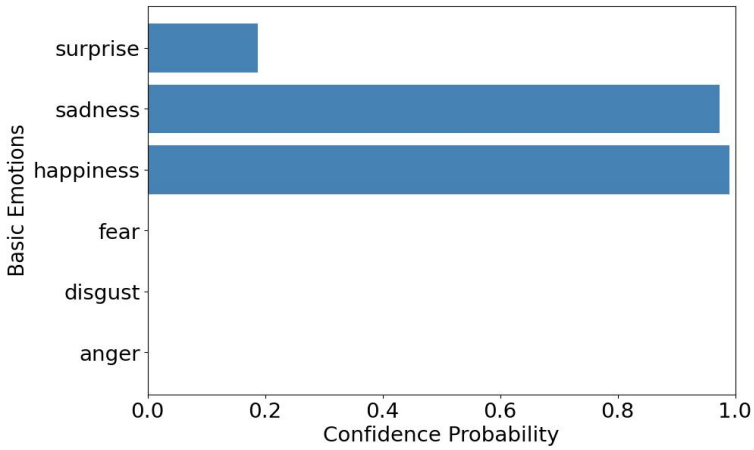
Models	Metrics					
	f_β			EMR		
	CK+	JAFFE	MUG	CK+	JAFFE	MUG
EfficientNet B0[112]	0.45	0.60	0.51	6.2%	5.0%	5.9%
Multi-labels CNN model (StyleGANs)	0.42	0.54	0.53	14.2%	25.6%	26.0%
Multi-labels CNN model (EmoGANs3)	**0.99**	**0.99**	**0.80**	**100%**	**99%**	**68.7%**

The performance of all models is measured by two metrics: Exact Match Ratio (EMR) and f_β. The EMR is the metric where the predicted emotion labels are the same as the true labels used in the image generation process. The output of the Exact Match Ratio (EMR) is similarly interpreted as a true positive rate. Evaluation results are tabulated in Table.6.5. Based on the results, the multi-label model works well with generated images by EmoGANs3 in CK+ and JAFFE compared to MUG. Approximately 68% of the predicted multiple labels are the same as the true labels of generated images in MUG. This is because the characteristics of MUG are different from the other two. Moreover, the multi-label CNN model achieves higher scores compared to the model that had the same network configurations but was trained based on generated images by StyleGANs.

Previously, the example-generated images were incorrectly recognized by a fine-tuned Efficient Net B0 model that was trained on basic images and Py-feat in Figure.6.6. The emotion of the example image will be estimated by the multi-label CNN model in Figure.6.7. Figure.6.8 shows the estimation results. From the result, we can see that the model can estimate the correct label, which is used to generate the image by EmoGANs3. We also test the model on a



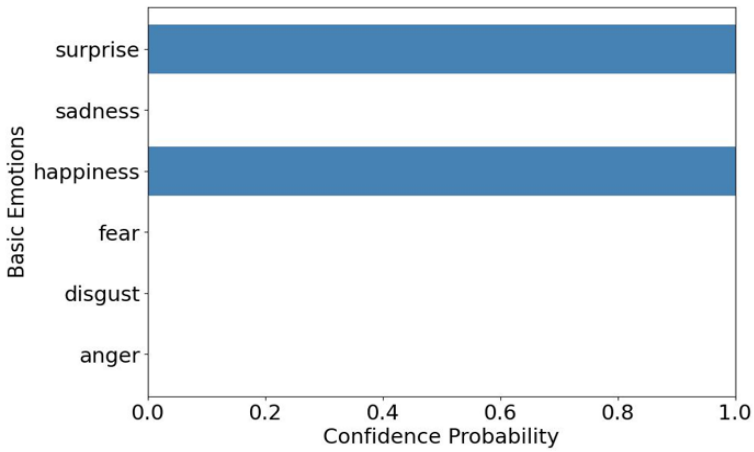
(a) Generated image (MUG)



(b) Confidence probability score

Figure 6.8: Estimation of mixed emotions of an example generated image by multi-labels CNN model in MUG.

(a) Gold Medalist Risako Kawai during Rio Olympic

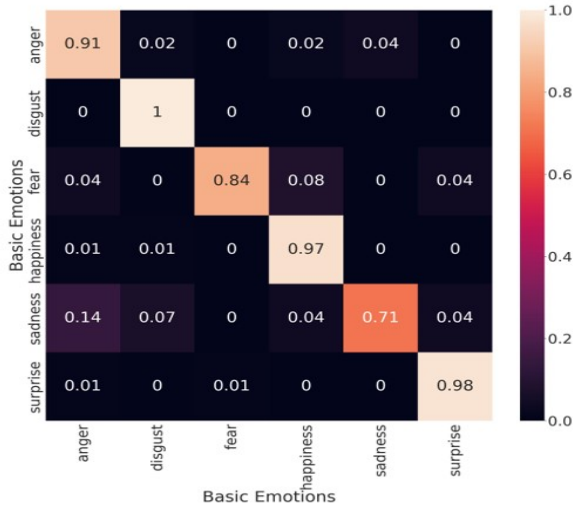


(b) Confidence probability score

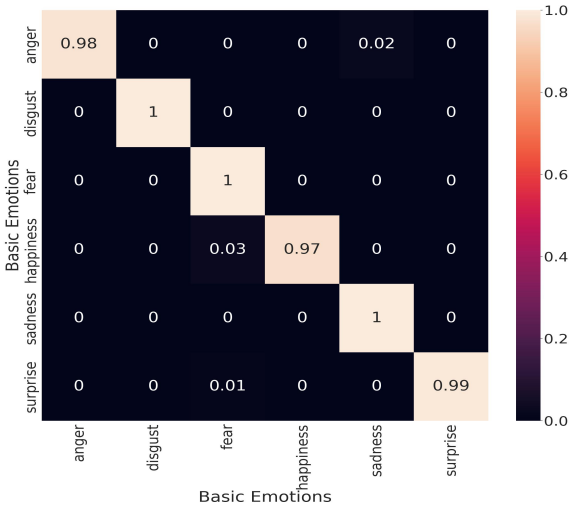
Figure 6.9: Estimation of mixed emotions of a real image by multi-labels CNN model.

given real sample and the result is shown in Figure.6.9. This photo was captured at the winning moment during the Olympic games. Therefore, the subject in the photo has happy emotions, Since Olympic game results are unpredictable, the subject also has surprise emotions as well. Those emotions can be predicted by the multi-label CNN model.

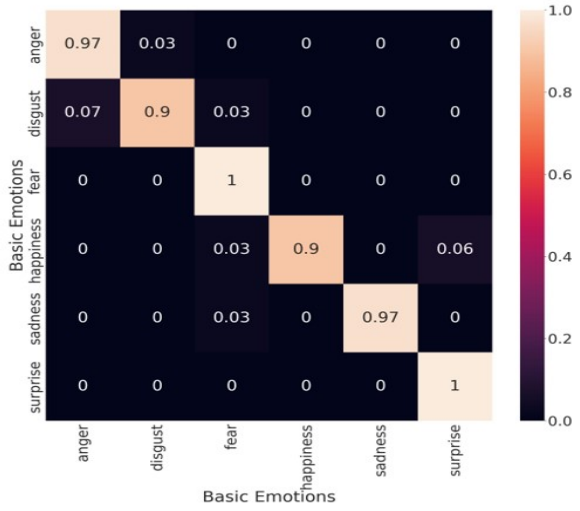
One can argue that the multi-label mixed emotions estimation model works well because it was directly trained based on the generated images. We also evaluate the model based on the basic facial expressions images as well. Confusion matrices for each basic class by multi-labels estimation model and Efficient Net B0 model[112] is shown in Figure.6.10. Both models are trained on the same data which are basic facial expressions images. Based on the confusion matrix, the multi-label CNN model performed better than the Efficient Net B0 in JAFFE. Their performances are similar in the other two datasets.



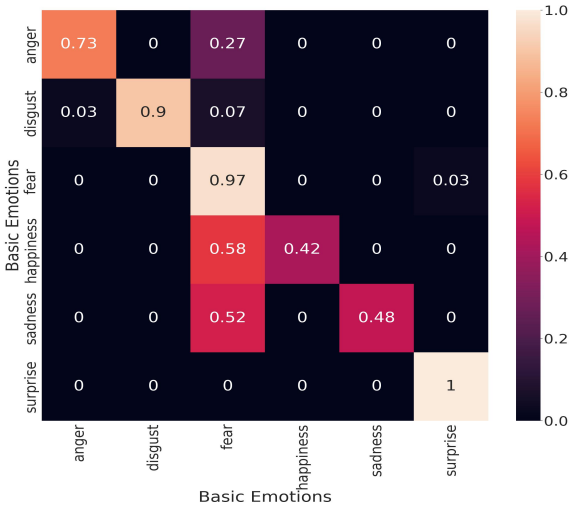
(a) Mixed emotions estimation model (CK+)



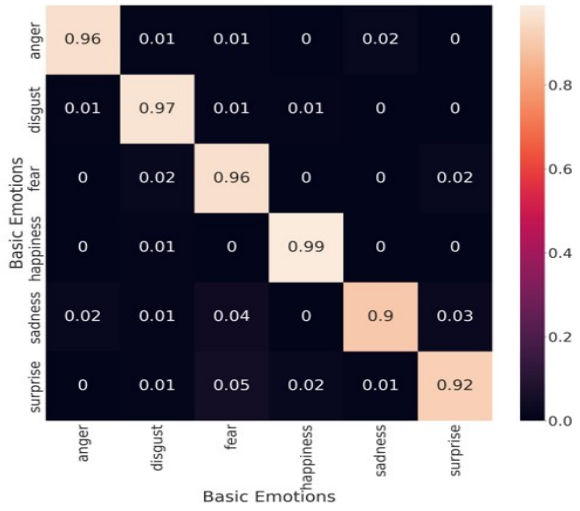
(b) Efficient Net B0 (CK+)



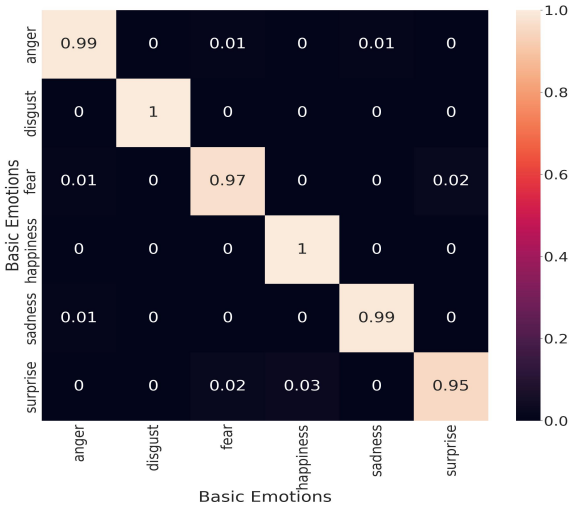
(c) Mixed emotions estimation model (JAFFE)



(d) Efficient Net B0 (JAFFE)



(e) Mixed emotions estimation model (MUG)



(f) Efficient Net B0 (MUG)

Figure 6.10: Confusion matrices evaluated over **basic facial expressions images** by multi-label CNN model and Efficient Net B0 [112].

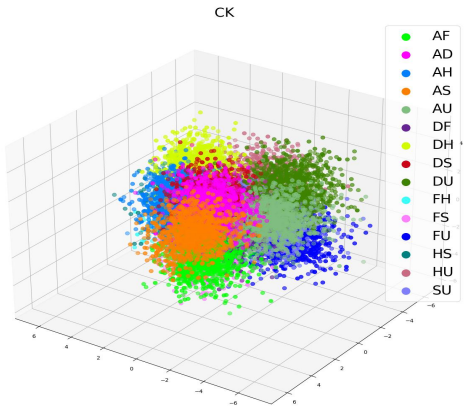
To summarize the results from Table 6.5 and the confusion matrices, it can be concluded that the EfficientNet B0 model, which we identified as the best model for extracting discriminable facial expression features, is not suitable for estimating mixed facial expressions in images generated by EmoGANs3. However, in terms of generated images representing mixed facial expressions, it can be observed that images generated by EmoGANs3 supported the model in achieving better recognition performance. Additionally, the multi-label CNN model demonstrates comparable performance in recognizing basic facial expressions in all datasets.

We now further analyze the mixed facial expression features learned by the multi-label CNN model using Linear Discriminant Analysis (LDA), which summarizes the statistics of the given features for the best separation based on their labels. From the multi-label CNN model, 128-dimensional feature vectors are obtained before the final classification (Figure 6.7). These features are then learned by LDA to find the best linear combinations of the given variables to achieve maximum separation among inter-classes and minimum separation among intra-classes. It can also be used to reduce the feature dimensions. Therefore, 128-dimensional features by multi-labels CNN model are projected by LDA into three components which are plotted in Figure.6.11. Front and back sides of 3D plots are provided to see all mixed emotions classes for each dataset. Name of the each mixed emotion represented in the plot is coded in Table.6.6.

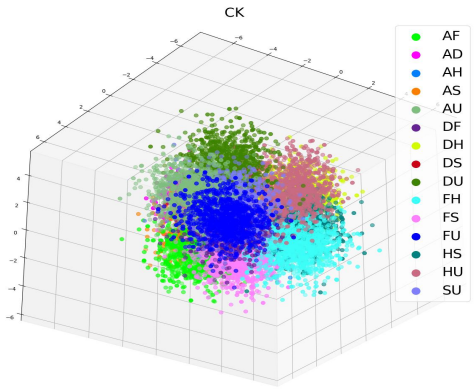
Table 6.6: Name code for mixed emotions classes

No.	Mixed Emotion Classes	Name Code
1	Anger, Fear	AF
2	Anger, Disgust	AD
3	Anger, Happiness	AH
4	Anger, Sadness	AS
5	Anger, Surprise	AU
6	Disgust, Fear	DF
7	Disgust, Happiness	DH
8	Disgust, Sadness	DS
9	Disgust, Surprise	DU
10	Fear, Happiness	FH
11	Fear, Sadness	FS
12	Fear, Surprise	FU
13	Happiness, Sadness	HS
14	Happiness, Surprise	HU
15	Sadness, Surprise	SU

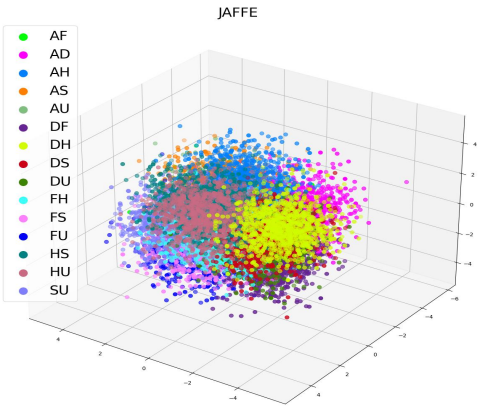
From the visualization results in Figure 6.11, it is evident that samples belonging to the same classes have a close distance in the projected space. Although a maximum distance between samples among inter-classes is preferred, samples belonging to different classes are relatively close in the feature space. This is because the six basic emotions are intertwined with each other within each mixed emotion class. For example, in the plot for CK+, the mixed emotion classes associated with anger, such as AF, AD, AH, AS, and AU, are close to each other as they all contain the common class of anger. The same interpretation can be applied to the other plots. Therefore, from this visualization results, we can conclude that the mixed emotions estimation model provides relatively discriminable features useful to recognize mixed emotions.



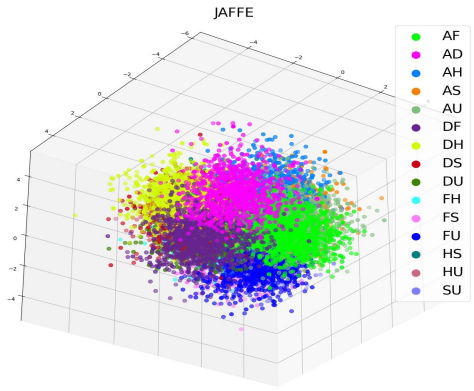
(a) LDA of CK+ (front side)



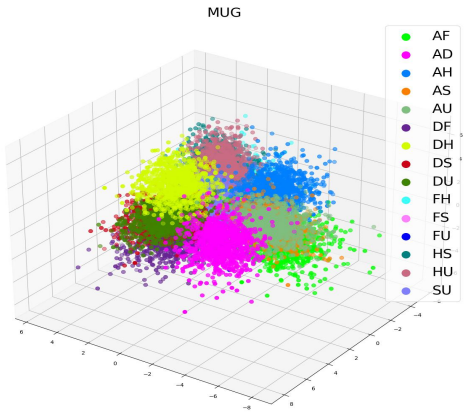
(b) LDA of CK+ (back side)



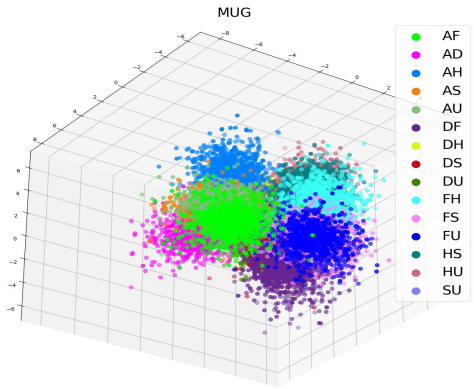
(c) LDA of JAFFE (front side)



(d) LDA of JAFFE (back side)



(e) LDA of MUG (front side)



(f) LDA of MUG (back side)

Figure 6.11: 3D plot of Linear Discirminant Analysis (LDA) for features related to mixed facial expressions extracted by multi-labels CNN model. Each dimension represents each LDA component. Each color represents each mixed emotions class, which can be referred in Table.6.6.

Chapter 7

Conclusion and Future Work

7.1 Chapters Summary

With advanced technology, emotion recognition is no longer limited to human interaction but is also performed by computers using social signals, especially facial expressions, as faces are the first engaged area in face-to-face communication. However, state-of-the-art research on emotion recognition typically focuses on six basic emotions, but human emotions are more sophisticated than those emotions. In Chapter 1, possible life events are shown as an example of mixed emotions. Currently, the literature on mixed emotions describes mixed facial expressions as a linear combination of action units. However, it might not be a linear transformation and highlights the gap between spontaneous and non-spontaneous mixed facial expressions images. This motivates us to consider non-linear transformations for the formation of mixed facial expressions. Chapter 1 describes our definition of complex facial expressions for mixed emotions and introduces the overview of the approach used in this dissertation by emphasizing its significance.

Chapter 2 divides the literature into two parts. The first part reviews the literature on generation models, typically focusing on Generative Adversarial Networks and their variant models that are commonly used in the image generation process. The second part includes reviews of the literature on automatic emotion recognition through facial expressions. Chapter 3 describes the data preparation process used in this dissertation to train the proposed models. Since the current research focuses on working with images, the chapter introduces the process of selecting the peak frame from image sequences as the pre-processing step. Additionally, morphed images were used as training data, as morphing can create a mixture of images. However, it should be noted that morphing is a linear transformation process and cannot perform interpolation outside of the given pixels.

The beginning of Chapter 4 briefly introduces the proposed model for mixed emotions estimation using an analysis-by-synthesis approach. Consequently, it discusses the proposed Emotion Generative Adversarial Networks (EmoGANs) and its evolution. As a summary, it can be divided into two categories: Unsupervised and Supervised. During the adversarial training, the GANs model does not use the original label from the data, making GANs training known as unsupervised training.

In the first part, we introduce the first proposed model named Emotion Generation Adversarial Networks (EmoGANs). We discuss their network configurations, training settings, and output while pointing out the challenges associated with this model. However, due to its unstable training and noisy output, a second model named EmoGANs1 is introduced. EmoGANs1 utilizes a Wasserstein distance-based objective function, resulting in less noisy images compared

to the first model. Nevertheless, it still faces training instability due to its generator design. Therefore, the model is further restructured into two different models, EmoGANs2, by separating the feature extraction and image generation processes. To control the image generation process, we employ the label vector and inference network to deduce the facial representation in EmoGANs3.

In Chapter 5, the generated images are evaluated based on the perspective of image quality, image diversity, and feature disentanglement property and compared with state-of-art GAN variants. The generation model, such as GANs, utilizes samples from a prior existing distribution to estimate real samples from the training distribution. However, the knowledge of how visual attributes of the generated images are in the latent space is unknown. Therefore, Chapter 5 also explores the latent space used by the DCGANs model and introduces a methodology to edit the facial expression attributes in the latent space. Additionally, the chapter demonstrates how facial expression manipulation benefits the emotion recognition model.

Based on the evaluation results of generated images by the proposed EmoGANs models, we concluded that EmoGANs3 provides the best performance and the generated images by EmoGANs3 are used for further analysis. Chapter 6 includes two sections. The first section investigates the best model among the commonly used models for facial expression recognition, which provides the most discriminative features useful for basic emotion estimation. The second section discusses the multiple emotions estimation model used to estimate the mixed emotion labels and analyzes them in the feature space.

7.2 Conclusion

In this research, we aim to estimate mixed emotions using synthesized images through the Analysis-by-Synthesis (AbS) approach. The research milestones include obtaining mixed facial expression images and estimating the corresponding emotion labels.

Based on the experimental results obtained from the proposed EmoGANs models in Chapters 4 and 5, we can conclude that the first milestone has been accomplished. Among the four models introduced in this research, the final model (EmoGANs3) provides the highest-quality images. The evaluation of these images was conducted considering the spatial resolution, image quality, and data diversity using three different metrics: Inception Score, Frechet Inception Score, and Blind/Referenceless Image Spatial Quality Evaluator (BRISQE). The scores of these metrics for the generated images by EmoGANs3 were found to be the highest when tested on three different datasets, namely CK+, JAFFE, and MUG. Since only the EmoGANs3 model considers identity information, we can conclude that maintaining facial embedding helps the model in producing high-quality images.

During the adversarial training for EmoGANs models, morphed images are used to support the training process. Morphing involves blending the same local regions from two given images, thereby producing an intermediate image that represents the mixed facial expressions of the image pair. However, morphing is limited to performing linear interpolation and cannot go beyond the given pixel values, which means it cannot generate a new subject identity.

In contrast, EmoGANs employ non-linear activation in their network configurations. The projection mapping from the latent space onto the image space is a non-linear transformation. This enables each sample from the latent space to be transformed into a new image, thereby increasing the data diversity and introducing large variance. According to the inception score metric, EmoGANs3 achieves a score of 11.4 on the MUG dataset, outperforming state-of-the-art models, including other EmoGANs models.

As the name suggests, the inception score metric employs the inception model to measure

the scores and limits the generated images to be recognized by the inception model. Therefore, we utilized the multi-class facial expressions recognition model to calculate the score. Since that recognition model is trained using morphed images, it might favor the morphed images. Therefore, both morphed images and generated images by all EmoGAN models are evaluated based on BRISQE, which derives the feature vector only from the input without involving another model. The BRISQE score of generated images by the EmoGANs3 model is superior to that of the morphed images.

As for the second milestone of this research, the first section (Chapter 6) discusses several recognition models commonly used in the literature to recognize basic facial expression images. Based on the results, we concluded that EfficientNet B0 has the best network configurations, supporting the most discriminant features for estimating basic emotions.

However, estimating mixed emotions from generated images using EfficientNet B0 is not appropriate. Instead, multi-label classification is more suitable for estimating multiple emotion labels from the generated images. Therefore, the second section of Chapter 6 focuses on the multiple-label CNN model used to estimate mixed emotion labels, which is evaluated based on f_β and the exact match ratio (EMR).

According to the scores supported by these metrics, the multi-label CNN model outperforms the previously claimed best model, EfficientNet B0. Since both models have different training data (the multi-label CNN model is trained using generated images by EmoGANs3, whereas EfficientNet B0 is trained on basic images), we evaluated the multi-label model on those basic emotion images. Analyzing the results, the multi-label CNN model shows comparable performance in all datasets. Besides, we tested the model on the real sample for spontaneous mixed facial expressions. Its results showed that the multi-label CNN model can effectively estimate the mixed emotions.

To gain insights into the features extracted by the multi-label CNN model, we employed linear discriminant analysis (LDA) to reduce the feature dimensions and identify the optimal linear combinations of input variables. This approach aims to maximize the distance between different classes (inter-class) while minimizing the distance within the same class (intra-class).

Based on the visualization results of LDA features on 3D plots, we can conclude that the mixed facial expressions are relatively closer to other classes, as they share the six basic emotions among the mixed classes. This suggests that there is some overlap and similarity between the mixed facial expressions.

7.3 Future Work

For the short term, we consider the following work.

- Mixed emotions between 'happiness' and 'sadness' can be defined by the single word 'bittersweet'. Defining the vocabulary for each mixed emotion will be taken into consideration.

In the long term, we expect the following work.

- The current research introduces 15 mixed emotions, which represent possible non-repetitive combinations of Ekman's six basic emotions. At the moment, the degree of each underlying emotion in mixed emotions is not considered for the sake of simplicity. However, it will be addressed in future studies.

- The current research simplifies the scope of mixed emotions by focusing on images instead of image sequences and using standard datasets that have a minimal occlusion in faces and

maximally expressed facial expressions. However, these conditions do not occur frequently in real life. Therefore, for long-term studies, we will consider more robust models and recognition systems that can be applicable to real-world conditions.

Publications

(I) Win Shwe Sin Khine, Siritanawan Prarinya and Kotani Kazunori. 'Generation of compound emotions expressions with emotion generative adversarial networks (emogans)', 59th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), pp.748–755, Chiang Mai, Thailand (2020)

(II) Win Shwe Sin Khine, Shinobu Hasegawa and Kotani Kazunori. 'Engagement Estimation for an E-Learning Environment Application', The Thirteenth International Conference on Advances in Computer-Human Interactions, pp.1–6, Valencia, Spain (2021)

(III) Win Shwe Sin Khine, Siritanawan Prarinya and Kotani Kazunori. 'Wasserstein Based EmoGANs+', Joint 10th International Conference on Informatics, Electronics & Vision (ICIEV) and 2021 5th International Conference on Imaging, Vision & Pattern Recognition (icIVPR), pp.1–8, Fukuoka, Japan (2021)

(IV) Win Shwe Sin Khine, Siritanawan Prarinya and Kotani Kazunori. 'Automatic Peak Frame Selection from Dynamic Facial Expressions', 60th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), pp.1088–1093, Tokyo, Japan (2021)

(V) Win Shwe Sin Khine, Siritanawan Prarinya and Kotani Kazunori. 'Facial Expression Features Analysis With Transfer Learning' , 14th International Conference on Knowledge and Systems Engineering (KSE), pp.1–6, Nha Trang, Vietnam (2022)

(VI) Win Shwe Sin Khine, Siritanawan Prarinya and Kotani Kazunori. 'Disentangled Facial Expressions Editing in Trained Latent Space', IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp.1700–1706, Prague, Czech Republic (2022)

(VII) Win Shwe Sin Khine, Siritanawan Prarinya and Kotani Kazunori. 'Compound facial expressions image generation for complex emotions', Multimedia Tools and Applications, vol.82, no.8, pp.11549–11588 (2023)

Bibliography

[1] Mehrabian, Albert and Ferris, Susan R. Inference of attitudes from nonverbal communication in two channels., *Journal of consulting psychology*, vol.31, no.3, pp.248(1967)

[2] Kleinginna, Paul R and Kleinginna, Anne M. A categorized list of emotion definitions, with suggestions for a consensual definition, *Motivation and emotion*, vol.5, no.4, pp.345–379 (1981)

[3] Hamilton, David R. *I heart me: The science of self-love*, Hay House Pub, (2015)

[4] Matsumoto, David and Willingham, Bob. Spontaneous facial expressions of emotion of congenitally and noncongenitally blind individuals, *Journal of personality and social psychology*, vol.96, no.1, pp.1 (2009)

[5] Ekman, Paul and Friesen, Wallace V. Facial action coding system, *Environmental Psychology & Nonverbal Behavior*, (1978)

[6] Prarinya Siritanawan (2015), Estimation of human emotion by analyzing facial expression transition of image sequences, Doctoral Dissertation, Japan Advanced Institute of Science and Technology

[7] Russell, James A. A circumplex model of affect, *Journal of personality and social psychology*, vol.39, no.6, pp.1161 (1980)

[8] Zhang, Hanzhong and Yin, Jibin and Zhang, Xiangliang. The Study of a Five-Dimensional Emotional Model for Facial Emotion Recognition, *Mobile Information Systems*, vol.2020,(2020)

[9] Jack, Rachael E and Garrod, Oliver GB and Schyns, Philippe G. Dynamic facial expressions of emotion transmit an evolving hierarchy of signals over time, *Current biology*, vol.24, no.2, pp.187–192 (2014)

[10] Plutchik, Robert and Kellerman, Henry. *Biological foundations of emotion*, Academic Press, (2013)

[11] Mehrabian, Albert. Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in temperament, *Current Psychology*, vol.14, no.4, pp.261–292(1996)

[12] Larsen, Jeff T and McGraw, A Peter. The case for mixed emotions, *Social and Personality Psychology Compass*, vol.8, no.6, pp.263–274 (2014)

[13] Du, Shichuan and Tao, Yong and Martinez, Aleix M. Compound facial expressions of emotion, *Proceedings of the National Academy of Sciences*, vol.111, no.15, pp.E1454–E1462 (2014)

[14] Du, Shichuan and Martinez, Aleix M. Compound facial expressions of emotion: from basic research to clinical applications, *Dialogues in clinical neuroscience*, (2015)

[15] Ekman, Paul and Keltner, Dacher. Universal facial expressions of emotion, *California mental health research digest*, vol.8, no.4, pp.151–158 (1970)

[16] Jin Hyun Cheong and Eshin Jolly and Tiankang Xie and Sophie Byrne and Matthew Kenney and Luke J. Chang. Py-Feat: Python Facial Expression Analysis Toolbox, *arXiv*(2023)

[17] Halle, Morris and Stevens, Kenneth. Speech recognition: A model and a program for research, *IRE transactions on information theory*, vol.8, no.2, pp.155–159 (1962)

[18] Parent, André. Duchenne De Boulogne: a pioneer in neurology and medical photography, *Canadian journal of neurological sciences*, vol.32, no.3, pp.369–377 (2005)

[19] Hupka, Ralph B. Jealousy: Compound emotion or label for a particular situation?, *Motivation and Emotion*, vol.8, pp.141–155 (1984)

[20] LaPlante, Debi and Ambady, Nalini. Multiple messages: Facial recognition advantage for compound expressions, *Journal of Nonverbal Behavior*, vol.24, pp.211–224 (2000)

[21] Martinez, Aleix and Du, Shichuan. A model of the perception of facial expressions of emotion by humans: research overview and perspectives., *Journal of Machine Learning Research*, vol.13, no.5 (2012)

[22] Lucey, Patrick and Cohn, Jeffrey F and Kanade, Takeo and Saragih, Jason and Ambadar, Zara and Matthews, Iain. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression, *2010 ieee computer society conference on computer vision and pattern recognition-workshops*, pp.94–101 (2010)

[23] Lyons, Michael and Akamatsu, Shigeru and Kamachi, Miyuki and Gyoba, Jiro. Coding facial expressions with gabor wavelets, *Proceedings Third IEEE international conference on automatic face and gesture recognition*, pp.200–205 (1998)

[24] Aifanti, Niki and Papachristou, Christos and Delopoulos, Anastasios. The MUG facial expression database, *11th International Workshop on Image Analysis for Multimedia Interactive Services WIAMIS 10*, pp.1–4 (2010)

[25] Goodfellow, Ian J and Erhan, Dumitru and Carrier, Pierre Luc and Courville, Aaron and Mirza, Mehdi and Hamner, Ben and Cukierski, Will and Tang, Yichuan and Thaler, David and Lee, Dong-Hyun and others. Challenges in representation learning: A report on three machine learning contests, *Neural Information Processing: 20th International Conference, ICONIP 2013, Daegu, Korea, November 3-7, 2013. Proceedings, Part III 20*, pp.117–124 (2013)

[26] D. Aneja and B. Chaudhuri and A. Colburn and G. Faigin and L. Shapiro and B. Mones. Learning to Generate 3D Stylized Character Expressions from Humans, *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp.160-169 (2018)

[27] Goodfellow, Ian and Pouget-Abadie, Jean and Mirza, Mehdi and Xu, Bing and Warde-Farley, David and Ozair, Sherjil and Courville, Aaron and Bengio, Yoshua. Generative adversarial nets, *Advances in neural information processing systems*, Publisher, pp.2672–2680(2014)

[28] Goodfellow, Ian and Bengio, Yoshua and Courville, Aaron. *Deep learning*, MIT press, (2016)

[29] Springenberg, Jost Tobias. Unsupervised and semi-supervised learning with categorical generative adversarial networks, *arXiv*(2015)

[30] Odena, Augustus. Semi-supervised learning with generative adversarial networks , *arXiv* (2016)

[31] Radford, Alec and Metz, Luke and Chintala, Soumith. Unsupervised representation learning with deep convolutional generative adversarial networks, *CoRR* (2015)

[32] Shuang Wu and Guoqi Li and Lei Deng and Liu Liu and Dong Wu and Yuan Xie and Luping Shi. $L1$ -Norm Batch Normalization for Efficient Training of Deep Neural Networks, *IEEE Transactions on Neural Networks and Learning Systems*, vol.30, no.7, pp.2043–2051 (2019)

[33] Mirza, Mehdi and Osindero, Simon. Conditional generative adversarial nets, *CoRR*, vol.abs/1411.1784(2014)

[34] Odena, Augustus and Olah, Christopher and Shlens, Jonathon. Conditional image synthesis with auxiliary classifier gans, *International conference on machine learning*, pp.2642–2651 (2017)

[35] Chen, Xi and Duan, Yan and Houthooft, Rein and Schulman, John and Sutskever, Ilya and Abbeel, Pieter. Infogan: Interpretable representation learning by information maximizing generative adversarial nets, *Advances in neural information processing systems*, vol.29(2016)

[36] Karras, Tero and Aila, Timo and Laine, Samuli and Lehtinen, Jaakko. Progressive growing of gans for improved quality, stability, and variation, *International Conference on Learning Representations*(2018)

[37] Karras, Tero and Laine, Samuli and Aila, Timo. A style-based generator architecture for generative adversarial networks, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp.4401–4410(2019)

[38] Martin Arjovsky and Soumith Chintala and Léon Bottou Wasserstein GAN, *arXiv* (2017)

[39] Gulrajani, Ishaan and Ahmed, Faruk and Arjovsky, Martin and Dumoulin, Vincent and Courville, Aaron C. Improved training of wasserstein gans, *Advances in neural information processing systems*, vol.30 (2017)

[40] Mao, Xudong and Li, Qing and Xie, Haoran and Lau, Raymond YK and Wang, Zhen and Paul Smolley, Stephen. Least squares generative adversarial networks, *Proceedings of the IEEE international conference on computer vision*, pp.2794–2802 (2017)

[41] Viola, Paul and Jones, Michael. Rapid object detection using a boosted cascade of simple features, *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, vol.1, pp.I–I (2001)

[42] Rowley, Henry A and Baluja, Shumeet and Kanade, Takeo. Neural network-based face detection, *IEEE Transactions on pattern analysis and machine intelligence*, vol.20, no.1, pp.23–38 (1998)

[43] King, Davis E. Max-margin object detection, *arXiv preprint arXiv:1502.00046*(2015)

[44] Jiang, Huaizu and Learned-Miller, Erik. Face detection with the faster R-CNN, *2017 12th IEEE international conference on automatic face & gesture recognition (FG 2017)*, pp.650–657(2017)

[45] Cootes, Timothy F and Taylor, Christopher J and Cooper, David H and Graham, Jim. Active shape models-their training and application, *Computer vision and image understanding*, vol.61, no.1, pp.38–59 (1995)

[46] Cootes, Timothy F. and Edwards, Gareth J. and Taylor, Christopher J. Active appearance models, *IEEE Transactions on pattern analysis and machine intelligence*, vol.23, no.6, pp.681–685 (2001)

[47] Cristinacce, David and Cootes, Timothy F and others. Feature detection and tracking with constrained local models., *Bmvc*, vol.1, no.2, pp.3 (2006)

[48] Happy, SL and Routray, Aurobinda. Robust facial expression classification using shape and appearance features, *2015 eighth international conference on advances in pattern recognition (ICAPR)*, pp.1–5(2015)

[49] Liu, Yu and Wang, Jing-dong and Li, Peng. A feature point tracking method based on the combination of SIFT algorithm and KLT matching algorithm, *Journal of Astronautics*, vol.32, no.7, pp.1618(2011)

[50] Islam, Rahul and Ahuja, Karan and Karmakar, Sandip and Barbhuiya, Ferdous. SenTion: A framework for sensing facial expressions, *arXiv preprint arXiv:1608.04489* (2016)

[51] Liew, Chun Fui and Yairi, Takehisa. Facial expression recognition and analysis: a comparison study of feature descriptors, *IPSJ transactions on computer vision and applications*, vol.7, pp.104–120 (2015)

[52] Gabor, Dennis. Theory of communication. Part 1: The analysis of information, *Journal of the Institution of Electrical Engineers-part III: radio and communication engineering*, vol.93, no.26, pp.429–441 (1946)

[53] Al-Sumaidaee, S. A. M. and Dlay, S. S. and Woo, W. L. and Chambers, J. A. Facial expression recognition using local Gabor gradient code-horizontal diagonal descriptor, *2nd IET International Conference on Intelligent Signal Processing 2015 (ISP)*, pp.1-6, IET, (2015)

[54] Tsai, Hung-Hsu and Chang, Yi-Cheng. Facial expression recognition using a combination of multiple facial features and support vector machine, *Soft Computing*, vol.22, pp.4389–4405 (2018)

[55] Huang, Xiaohua and Zhao, Guoying and Pietikäinen, Matti and Zheng, Wenming. Dynamic facial expression recognition using boosted component-based spatiotemporal features and multi-classifier fusion, *Advanced Concepts for Intelligent Vision Systems: 12th International Conference, ACIVS 2010, Sydney, Australia, December 13-16, 2010, Proceedings, Part II 12*, pp.312–322 (2010)

[56] Sirovich, Lawrence and Kirby, Michael. Low-dimensional procedure for the characterization of human faces, *Josa a*, vol.4, no.3, pp.519–524 (1987)

[57] Cortes, Corinna and Vapnik, Vladimir. Support-vector networks, *Machine learning*, vol.20, no.3, pp.273–297 (1995)

[58] Ho, Tin Kam. Random decision forests, *Proceedings of 3rd international conference on document analysis and recognition*, vol.1, pp.278–282(1995)

[59] Schapire, Robert E. Explaining adaboost, *Empirical inference*, pp.37–52, Springer (2013)

[60] Wu, Xindong and Kumar, Vipin and Quinlan, J Ross and Ghosh, Joydeep and Yang, Qiang and Motoda, Hiroshi and McLachlan, Geoffrey J and Ng, Angus and Liu, Bing and Philip, S Yu and others. Top 10 algorithms in data mining, *Knowledge and information systems*, vol.14, no.1, pp.1–37 (2008)

[61] Khan, Sajid Ali and Hussain, Ayyaz and Usman, Muhammad. Reliable facial expression recognition for multi-scale images using weber local binary image based cosine transform features, *Multimedia Tools and Applications*, vol.77, pp.1133–1165 (2018)

[62] LeCun, Yann and Bottou, Léon and Bengio, Yoshua and Haffner, Patrick. Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, vol.86, no.11, pp.2278–2324 (1998)

[63] Dang, Vu-Tuan and Do, Hong-Quan and Vu, Viet-Vu and Yoon, Byeongnam. Facial expression recognition: A survey and its applications, *2021 23rd International Conference on Advanced Communication Technology (ICACT)*, pp.359–367 (2021)

[64] Hamester, Dennis and Barros, Pablo and Wermter, Stefan. Face expression recognition with a 2-channel convolutional neural network, *2015 international joint conference on neural networks (IJCNN)*, pp.1–8(2015)

[65] Nwosu, Lucy and Wang, Hui and Lu, Jiang and Unwala, Ishaq and Yang, Xiaokun and Zhang, Ting. Deep convolutional neural network for facial expression recognition using facial parts, *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*, pp.1318–1321 (2017)

[66] Hinton, Geoffrey E and Osindero, Simon and Teh, Yee-Whye. A fast learning algorithm for deep belief nets, *Neural computation*, vol.18, no.7, pp.1527–1554 (2006)

[67] Ranzato, Marc'Aurelio and Susskind, Joshua and Mnih, Volodymyr and Hinton, Geoffrey. On deep generative models with applications to recognition, *CVPR 2011*, pp.2857–2864 (2011)

[68] Rifai, Salah and Bengio, Yoshua and Courville, Aaron and Vincent, Pascal and Mirza, Mehdi. Disentangling factors of variation for facial expression recognition, *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI 12*, pp.808–822 (2012)

[69] Liu, Ping and Han, Shizhong and Meng, Zibo and Tong, Yan. Facial expression recognition via a boosted deep belief network, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.1805–1812 (2014)

[70] Usman, Muhammad and Latif, Siddique and Qadir, Junaid. Using deep autoencoders for facial expression recognition, *2017 13th International Conference on Emerging Technologies (ICET)*, pp.1–6 (2017)

[71] Zeng, Nianyin and Zhang, Hong and Song, Baoye and Liu, Weibo and Li, Yurong and Dobaie, Abdullah M. Facial expression recognition via learning deep sparse autoencoders, *Neurocomputing*, vol.273, pp.643–649(2018)

[72] Larsen, Anders Boesen Lindbo and Sønderby, Søren Kaae and Larochelle, Hugo and Winther, Ole. Autoencoding beyond pixels using a learned similarity metric, *International conference on machine learning*, pp.1558–1566(2016)

[73] Werbos, Paul J. Backpropagation through time: what it does and how to do it, *Proceedings of the IEEE*, vol.78, no.10, pp.1550–1560 (1990)

[74] Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory, *Neural computation*, vol.9, no.8, pp.1735–1780(1997)

[75] Dalal, Navneet and Triggs, Bill. Histograms of oriented gradients for human detection, *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol.1, pp.886–893(2005)

[76] Jiang, Bin and Ren, Qiang and Dai, Fei and Xiong, Jian and Yang, Jie and Gui, Guan. Multi-task cascaded convolutional neural networks for real-time dynamic face recognition method, *Communications, Signal Processing, and Systems: Proceedings of the 2018 CSPS Volume III: Systems 7th*, pp.59–66 (2020)

[77] Liu, Wei and Anguelov, Dragomir and Erhan, Dumitru and Szegedy, Christian and Reed, Scott and Fu, Cheng-Yang and Berg, Alexander C. Ssd: Single shot multibox detector, *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pp.21–37 (2016)

[78] Papageorgiou, Constantine P and Oren, Michael and Poggio, Tomaso. A general framework for object detection, *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pp.555–562 (1998)

[79] Bradski, Gary and Kaehler, Adrian and others. OpenCV, *Dr. Dobb's journal of software tools*, vol.3, no.2(2000)

[80] Zhalehpour, Sara and Akhtar, Zahid and Eroglu Erdem, Cigdem. Multimodal emotion recognition based on peak frame selection from video, *Signal, Image and Video Processing*, vol.10, pp. 827–834(2016)

[81] Liong, Sze-Teng and See, John and Wong, KokSheik and Le Ngo, Anh Cat and Oh, Yee-Hui and Phan, Raphael. Automatic apex frame spotting in micro-expression database, *2015 3rd IAPR Asian conference on pattern recognition (ACPR)*,pp. 665–669(2015)

[82] Khine, Win Shwe Sin and Siritanawan, Prarinya and Kotani, Kazunori. Automatic Peak Frame Selection from Dynamic Facial Expressions, *2021 60th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pp. 1088–1093 (2021)

[83] Chen, Jingying and Xu, Ruyi and Liu, Leyuan. Deep peak-neutral difference feature for facial expression recognition, *Multimedia Tools and Applications*, vol.77, pp.29871–29887 (2018)

[84] Akhand, MAH and Roy, Shuvendu and Siddique, Nazmul and Kamal, Md Abdus Samad and Shimamura, Tetsuya. Facial emotion recognition using transfer learning in the deep CNN *Electronics*, vol.10, no.9, pp.1036 (2021)

[85] Xu, Mao and Cheng, Wei and Zhao, Qian and Ma, Li and Xu, Fang. Facial expression recognition based on transfer learning from deep convolutional networks, *2015 11th International Conference on Natural Computation (ICNC)*, pp.702–708 (2015)

[86] Dubey, Arun Kumar and Jain, Vanita. Automatic facial recognition using VGG16 based transfer learning model, *Journal of Information and Optimization Sciences*, vol.41, no.7, pp.1589–1596(2020)

[87] Parkhi, Omkar M and Vedaldi, Andrea and Zisserman, Andrew. *Deep face recognition*, British Machine Vision Association, (2015)

[88] Arthur, David and Vassilvitskii, Sergei. K-means++ the advantages of careful seeding, *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp.1027–1035 (2007)

[89] Geitgey, Adam. Face recognition documentation, *Release*, vol.1, no.3, pp.3–37 (2019)

[90] Davis E. King. Dlib-ml: A Machine Learning Toolkit, *Journal of Machine Learning Research*,vol.10, pp. 1755-1758(2009)

[91] Khine, Win Shwe Sin and Siritanawan, Prarinya and Kotani, Kazunori. Generation of compound emotions expressions with emotion generative adversarial networks (emogans), *2020 59th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pp. 748–755(2020)

[92] Khine, Win Shwe Sin and Siritanawan, Prarinya and Kotani, Kazunori. Wasserstein Based EmoGANs+, *2021 Joint 10th International Conference on Informatics, Electronics & Vision (ICIEV) and 2021 5th International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pp. 1–8(2021)

[93] Win, Shwe Sin Khine and Siritanawan, Prarinya and Kotani, Kazunori. Compound facial expressions image generation for complex emotions, *Multimedia Tools and Applications*, vol.82, no.8, pp. 11549–11588(2023)

[94] Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift, *International conference on machine learning*, pp.448–456 (2015)

[95] Kingma, Diederik P and Ba, Jimmy. Adam: A method for stochastic optimization, *Proceedings of the 3rd International Conference on Learning Representations (ICLR)* (2015)

[96] Narayanan, Hariharan and Mitter, Sanjoy. Sample complexity of testing the manifold hypothesis, *Advances in neural information processing systems*, vol.23 (2010)

[97] Arjovsky, Martin and Bottou, Léon. Towards principled methods for training generative adversarial networks, *arXiv preprint arXiv:1701.04862*(2017)

[98] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, *Software available from tensorflow.org*(2015)

[99] LeCun, Yann and Bengio, Yoshua and others. Convolutional networks for images, speech, and time series, *The handbook of brain theory and neural networks*, vol.3361, no.10, pp.1995 (1995)

[100] Yi-Tong Zhou and Rama Chellappa. Computation of optical flow using a neural network, *IEEE 1988 International Conference on Neural Networks*, vol.2, pp.71-78 (1988)

[101] Salimans, Tim and Goodfellow, Ian and Zaremba, Wojciech and Cheung, Vicki and Radford, Alec and Chen, Xi. Improved techniques for training gans, *Advances in neural information processing systems* vol.29 (2016)

[102] Salimans, Tim and Kingma, Durk P. Weight normalization: A simple reparameterization to accelerate training of deep neural networks, *Advances in neural information processing systems*, vol.29 (2016)

[103] Ba, Lei Jimmy and Kiros, Jamie Ryan and Hinton, Geoffrey E. Layer normalization. CoRR abs/1607.06450 (2016), *CoRR, abs/1607.06450*, vol.178(2016)

[104] He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, *Proceedings of the IEEE international conference on computer vision*, pp.1026–1034 (2015)

[105] Khine, Win Shwe Sin and Siritanawan, Prarinya and Kotani, Kazunori. Disentangled Facial Expressions Editing in Trained Latent Space, *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp.1700–1706 (2022)

[106] Heusel, Martin and Ramsauer, Hubert and Unterthiner, Thomas and Nessler, Bernhard and Hochreiter, Sepp. Gans trained by a two time-scale update rule converge to a local nash equilibrium, *Advances in neural information processing systems*, vol.30 (2017)

[107] Szegedy, Christian and Vanhoucke, Vincent and Ioffe, Sergey and Shlens, Jon and Wojna, Zbigniew. Rethinking the inception architecture for computer vision, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.2818–2826 (2016)

[108] Upchurch, Paul and Gardner, Jacob and Pleiss, Geoff and Pless, Robert and Snavely, Noah and Bala, Kavita and Weinberger, Kilian. Deep feature interpolation for image content changes, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.7064–7073 (2017)

[109] Nonis, Francesca and Dagnes, Nicole and Marcolin, Federica and Vezzetti, Enrico. 3D approaches and challenges in facial expression recognition algorithms—a literature review, *Applied Sciences*, vol.9, no.18, pp.3904 (2019)

[110] Olivas, Emilio Soria and Guerrero, Jos David Mart and Martinez-Sober, Marcelino and Magdalena-Benedito, Jose Rafael and Serrano, L and others. *Handbook of research on machine learning applications and trends: Algorithms, methods, and techniques: Algorithms, methods, and techniques*, IGI global, (2009)

[111] Ng, Hong-Wei and Nguyen, Viet Dung and Vonikakis, Vassilios and Winkler, Stefan. Deep learning for emotion recognition on small datasets using transfer learning, *Proceedings of the 2015 ACM on international conference on multimodal interaction*, pp.443–449 (2015)

[112] Khine, Win Shwe Sin and Siritanawan, Prarinya and Kotani, Kazunori. Facial Expression Features Analysis With Transfer Learning, *2022 14th International Conference on Knowledge and Systems Engineering (KSE)*, pp.1–6 (2022)

[113] Chowdary, M Kalpana and Nguyen, Tu N and Hemanth, D Jude. Deep learning-based facial emotion recognition for human–computer interaction applications, *Neural Computing and Applications*, pp.1–18 (2021)

[114] Houshmand, Bita and Khan, Naimul Mefraz. Facial expression recognition under partial occlusion from virtual reality headsets based on transfer learning, *2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM)*, vol.xx, no.yy, pp.70–75 (2020)

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

[116] Sandler, Mark and Howard, Andrew and Zhu, Menglong and Zhmoginov, Andrey and Chen, Liang-Chieh. Mobilenetv2: Inverted residuals and linear bottlenecks, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.4510–4520(2018)

[117] Szegedy, Christian and Ioffe, Sergey and Vanhoucke, Vincent and Alemi, Alexander. Inception-v4, inception-resnet and the impact of residual connections on learning, *Proceedings of the AAAI conference on artificial intelligence*, vol.31, no.1 (2017)

[118] Tan, Mingxing and Le, Quoc. Efficientnet: Rethinking model scaling for convolutional neural networks, *International conference on machine learning*, pp.6105–6114 (2019)

[119] Selvaraju, Ramprasaath R and Cogswell, Michael and Das, Abhishek and Vedantam, Ramakrishna and Parikh, Devi and Batra, Dhruv. Grad-cam: Visual explanations from deep networks via gradient-based localization, *Proceedings of the IEEE international conference on computer vision*, pp.618–626 (2017)

[120] Liu, Bingchen and Zhu, Yizhe and Song, Kunpeng and Elgammal, Ahmed. Towards faster and stabilized gan training for high-fidelity few-shot image synthesis, *International Conference on Learning Representations*(2021)

[121] Mittal, Anish and Moorthy, Anush Krishna and Bovik, Alan Conrad. No-reference image quality assessment in the spatial domain, *IEEE Transactions on image processing*, vol.21, no.12, pp.4695–4708 (2012)

Appendix A

Notation

Table A.1: Notation of symbols, variables and functions with their descriptions used in the dissertation.

Notation	Descriptions
k	number of cluster in K-means clustering
i	index of an feature instance, observation or sample
c_i	i^{th} emotion class where $i \leq 6$
y	class label which has scalar value
$\mathbf{y}$	class label vector
$\hat{y}$	predicted label
u, v	coordinates of image pixels
p	confidence probability
$p(y)$	confidence probability score of being label y
j	attributes of a particular feature or feature dimension
d	distance score
a	accuracy score
$\hat{a}$	average accuracy
$s_{\mathbf{c}}$	summation the attribute values of cluster center
s	cosine similarity score
t	time step
μ	mean
ω	frequency of correct prediction
α	blending parameter in morphing
γ	momentum
$\mathbf{e}$	embedded facial representation
$\triangle$	difference
$\mathbf{f}$	facial expressions features
T	list of delaunay triangles
$\mathbf{C}$	covariance matrix
$\mathbf{V}$	feature matrix
$\mathbf{U}$	eigenvectors
$\mathbf{c}$	cluster center
$\mathbf{I}$	image
$\mathbf{I}_{c_i}$	image belong to i^{th} emotion class

Continued on next page

Table A.1 – continued from previous page

Notation	Descriptions
$\mathbf{z}$	latent sample
$\mathbf{x}$	input or training sample
$\mathbf{W}$	weighted matrix
$\mathbf{m}$	the first moment of the gradient descent during Adam optimization
$\mathbf{v}$	velocity vector or second moment of the gradient descent used in Adam optimization
$\boldsymbol{\theta}$	learning weights
β_1, β_2	hyper-parameters of Adam optimizer
ε	error during optimization by regression model
$\mathbb{S}$	set of Action Units (AUs)
$\mathbb{X}$	training set
$\mathbb{E}$	expectation or average
P_X	the input or real data distribution
P_Z	the latent distribution or Gaussian distribution
P_g	the generated distribution
$\hat{\mathbf{x}}$	the sample from the distribution $P_{\hat{\mathbf{x}}}$ uniformly sampling along the line between P_X and P_Z
$\mathcal{N}(0, 1)$	Gaussian distribution with zero mean and unit variance
ψ	regression coefficient
G	the generator
D	the discriminator
$\mathcal{L}$	objective function or loss function
$\mathcal{W}$	Wasserstein distance function
$A(\mathbf{x}, \boldsymbol{\phi})$	fine-tuned VGG16 face model that takes $\mathbf{x}$ as input and parameterized by the weights $\boldsymbol{\phi}$
e	standard exponential function
R	randomization or random function
B	facial feature detector provided by Dlib open source library
$W(.)$	warping function after affine transformation
$\mathcal{W}$	Wasserstein distance function
$E(.)$	function to find delaunay triangles
$V(D, G)$	objective value function of G and D in Generative Adversarial Networks (GANs)
J'	first order derivative of an objective function
g^2	element-wise multiplication of partial gradient g
$E(\mathbf{I})$	inference network to encode facial representation from input image $\mathbf{I}$

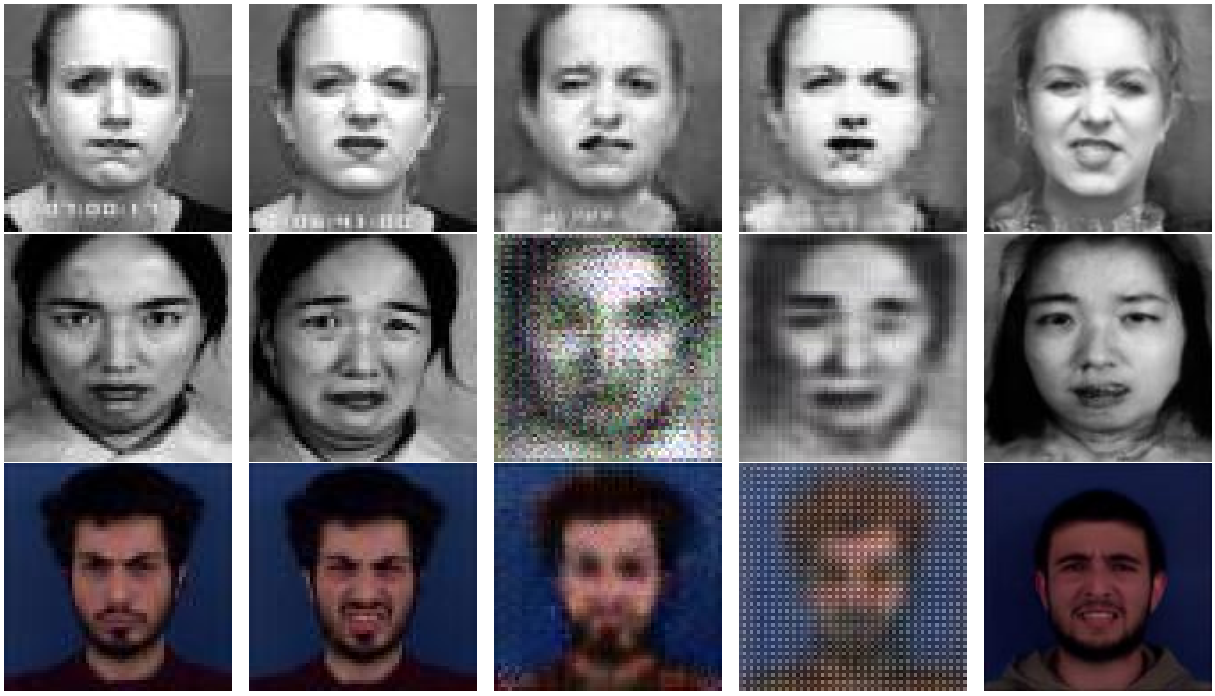
Appendix B

Additional Information

Table B.1: Relationship between Action Units (AUs) and Facial Muscle in Facial Action Coding Systems (FACS) [5]

AU Number	FACS Name	Muscular Basis
1	Inner Brow Raiser	Frontalis (Pars Medialis)
2	Outer Brow Raiser	Frontalis (Pars Lateralis)
4	Brow Lowerer	Depressor Glabellae; Depressor Supercilii; Corrugator Supercilii
5	Upper Lip Raiser	Levator Palpebrae Superioris; Superior Tarsal Muscle
6	Cheek Raiser	Orbicularis Oculi (Pars Orbitalis)
7	Lip Tightener	Orbicularis Oculi (Pars Palpebralis)
8	Lips Toward Each Other	Orbicularis Oris
9	Nose Wrinkler	Levator Labii Superioris Alaeque Nasi
10	Upper Lip Raiser	Levator Labii superioris; Caput Infraorbitalis
11	Nasolabial Deepener	Zygomaticus Minor
12	Lip Corner Puller	Zygomaticus Major
13	Sharp Lip Puller	Levator Anguli Oris (also known as Caninus)
14	Dimpler	Buccinator
15	Lip Corner Depressor	Depressor Anguli Oris (also known as Triangularis)
16	Lower Lip Depressor	Depressor Labii Inferioris
17	Chin Raiser	Mentalis
18	Lip Pucker	Incisivii Labii Superioris; Incisivii Labii Inferioris
20	Lip Stretcher	Risorius
21	Neck Tightener	Platysma
22	Lip Funneler	Orbicularis Oris
23	Lip Tightener	Orbicularis Oris
24	Lip Pressor	Orbicularis Oris
25	Lips Part	Depressor Labii Inferioris; or Relaxation of Mentalis; or Orbicularis Oris
26	Jaw Drop	Masseter; Relaxed Temporalis; Internal Pterygoid
27	Mouth Stretch	Pterygoids; Digastric
28	Lip Suck	Orbicularis Oris

B.1 Image Generation by EmoGANs, EmoGANs1 and EmoGANs2 From Given Inputs For Each Mixture of Emotions

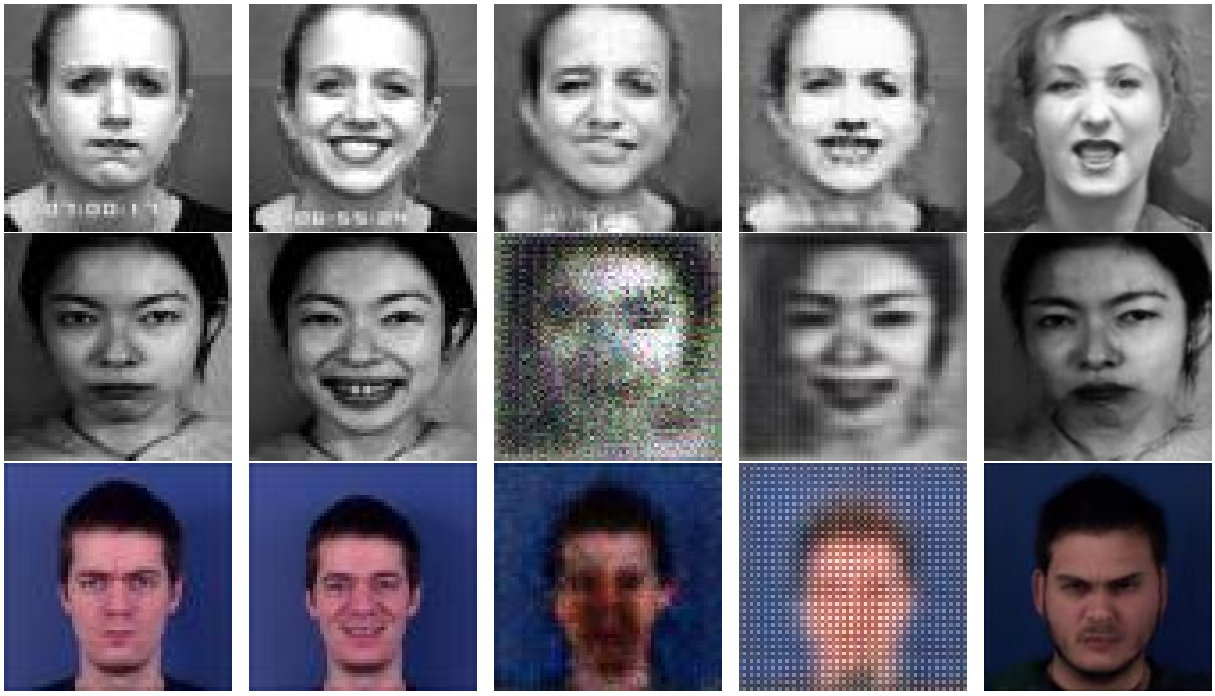


(a) (b) (c) (d) (e)

Figure B.1: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ $(c_i \neq c_j)$. In this figure, c_i indicates the ***'anger'*** emotion, and c_j refers to the ***'disgust'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$). Results from other classes can be found in the Appendix section.

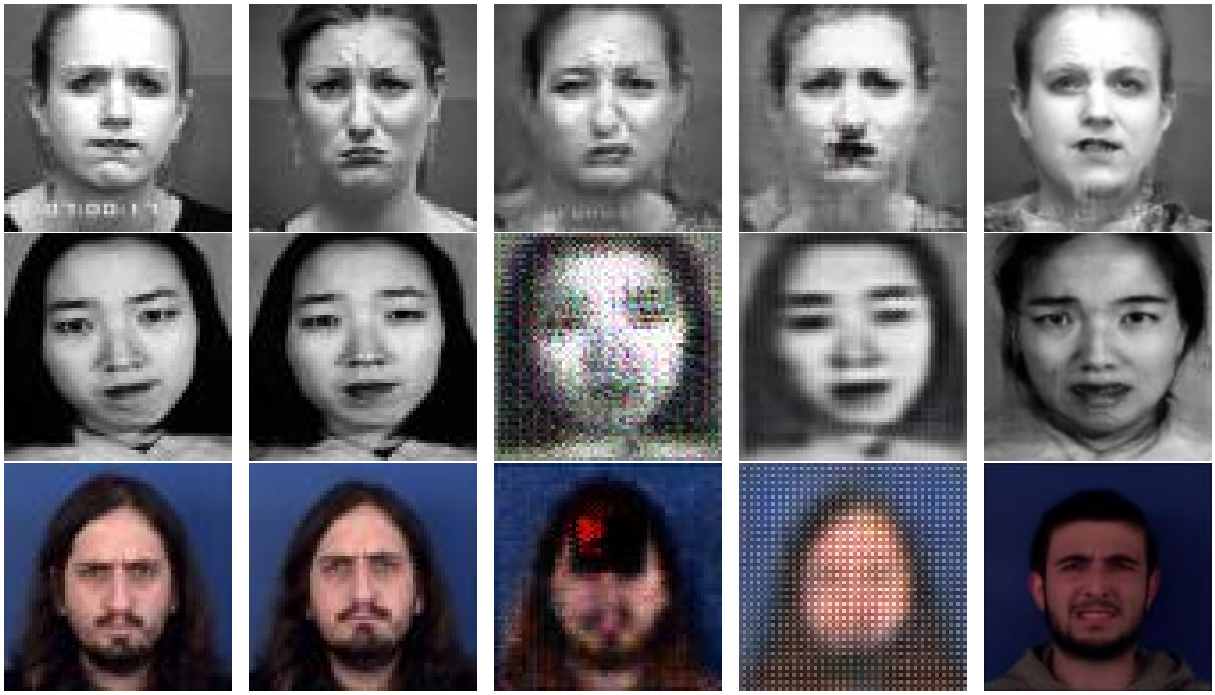
(a) (b) (c) (d) (e)

Figure B.2: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the *'**anger**'* emotion, and c_j refers to the *'**fear**'* class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



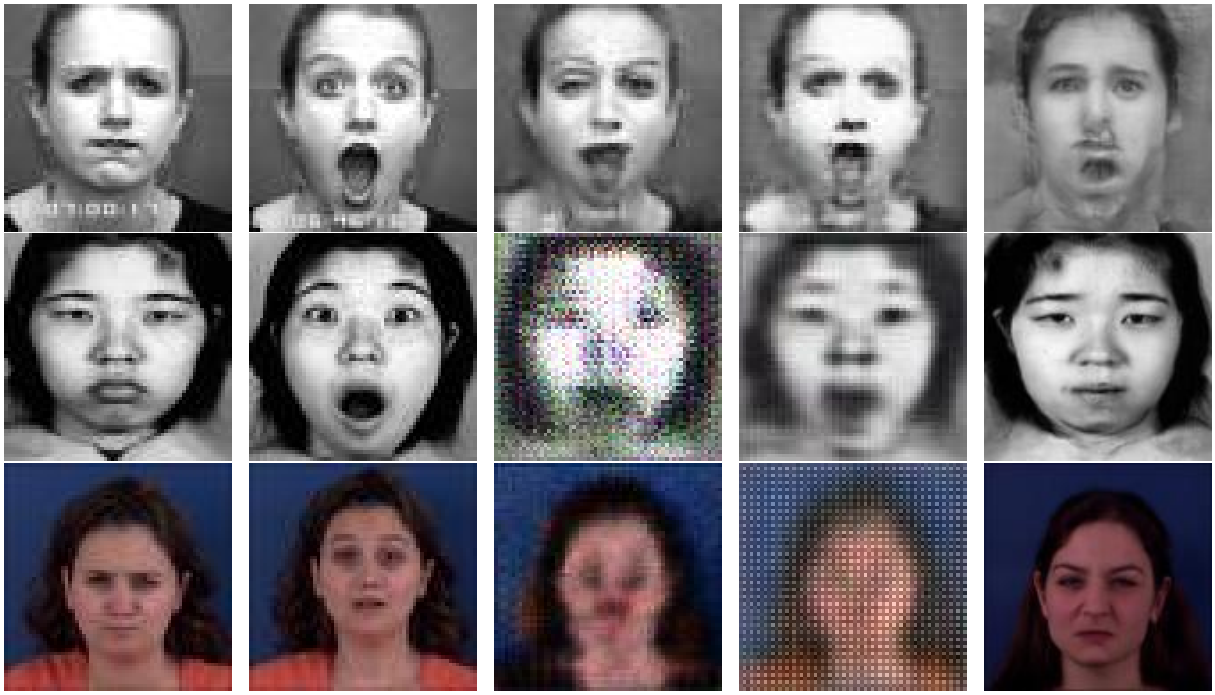
(a) (b) (c) (d) (e)

Figure B.3: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'anger'*** emotion, and c_j refers to the ***'happiness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



(a) (b) (c) (d) (e)

Figure B.4: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ $(c_i \neq c_j)$. In this figure, c_i indicates the *'**anger**'* emotion, and c_j refers to the *'**sadness**'* class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



(a) (b) (c) (d) (e)

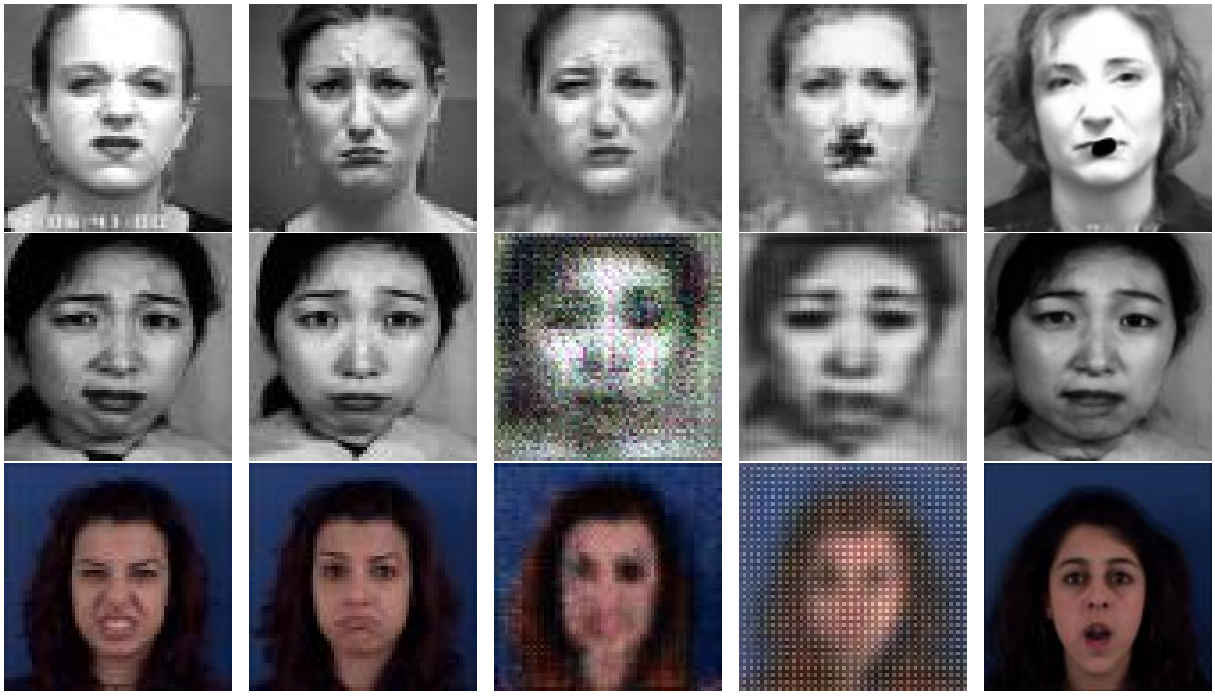
Figure B.5: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the *'**anger**'* emotion, and c_j refers to the *'**surprise**'* class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).

(a) (b) (c) (d) (e)

Figure B.6: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the *'**disgust**'* emotion, and c_j refers to the *'**fear**'* class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).

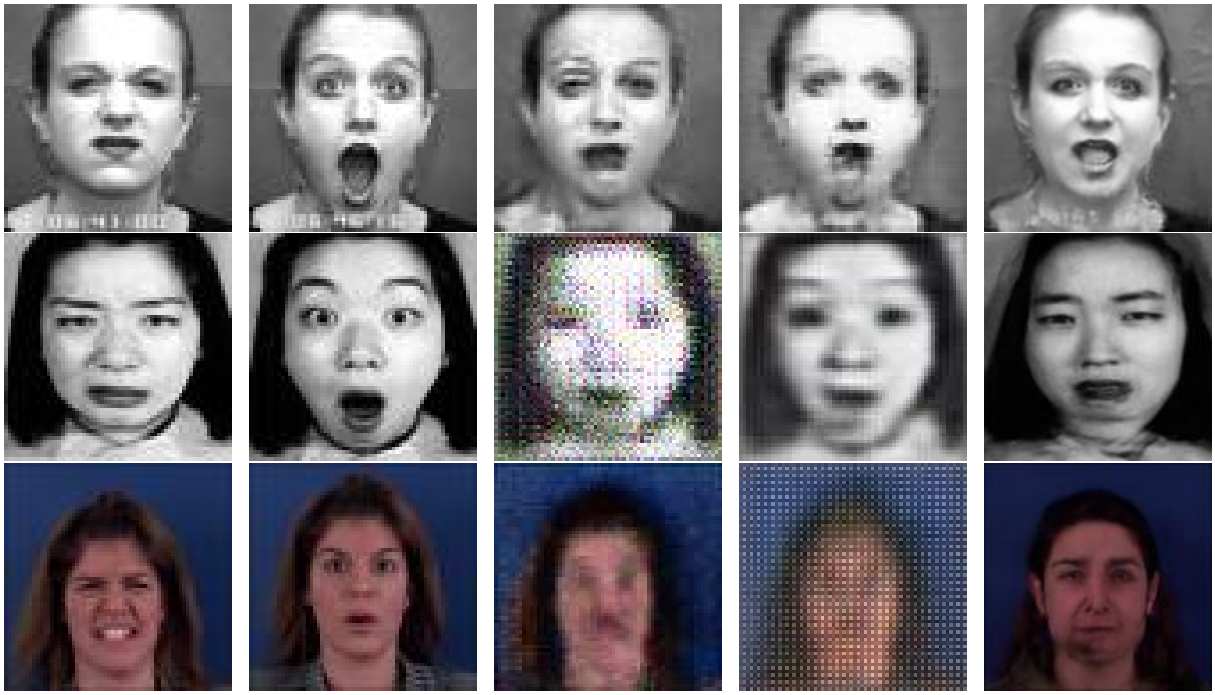
(a) (b) (c) (d) (e)

Figure B.7: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'disgust'*** emotion, and c_j refers to the ***'happiness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



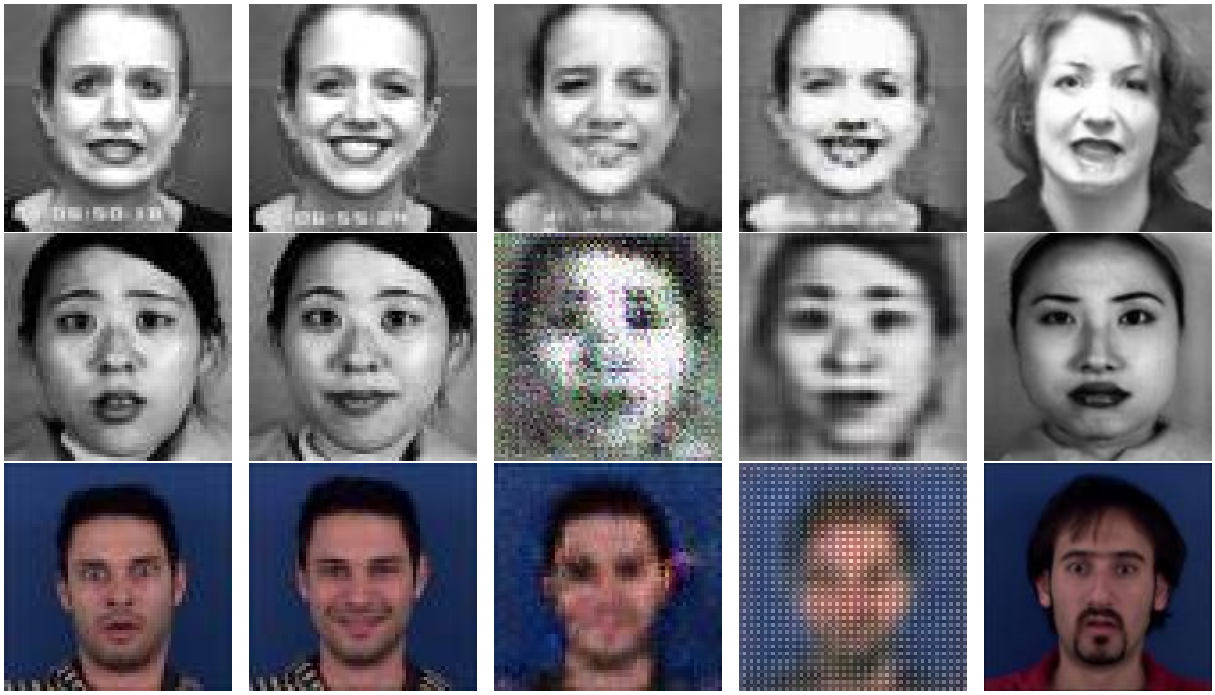
(a) (b) (c) (d) (e)

Figure B.8: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'disgust'*** emotion, and c_j refers to the ***'sadness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



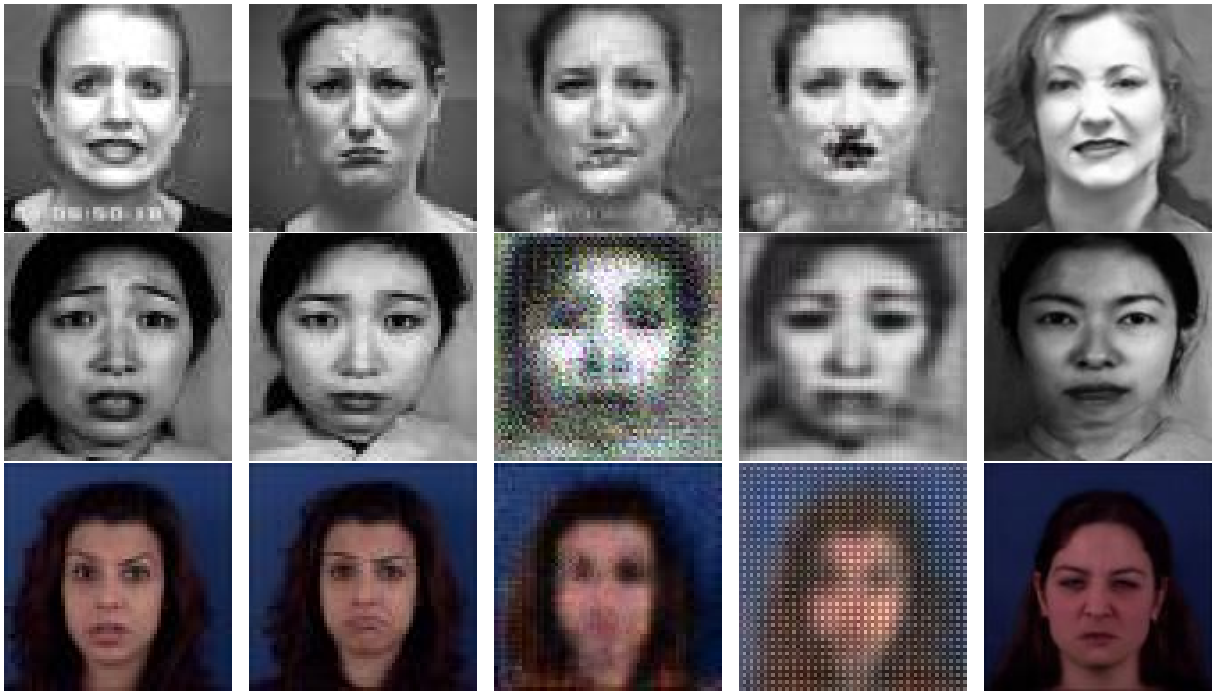
(a) (b) (c) (d) (e)

Figure B.9: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the *'**disgust**'* emotion, and c_j refers to the *'**surprise**'* class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



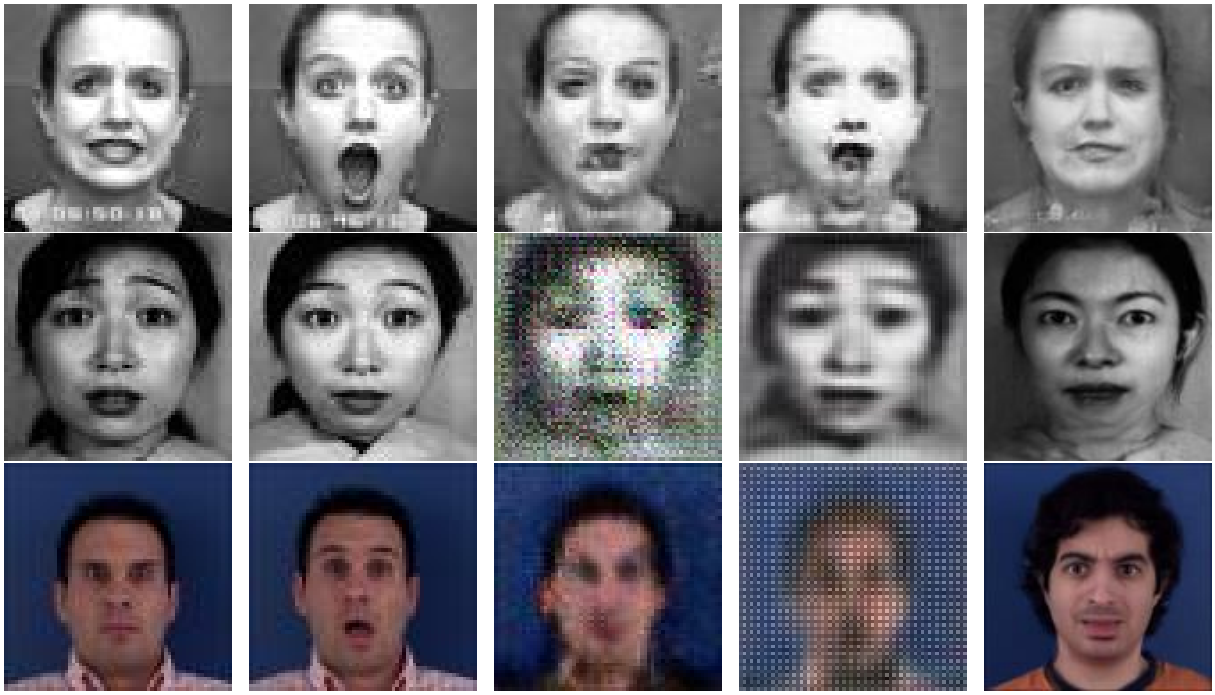
(a) (b) (c) (d) (e)

Figure B.10: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the *'**fear**'* emotion, and c_j refers to the *'**happiness**'* class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



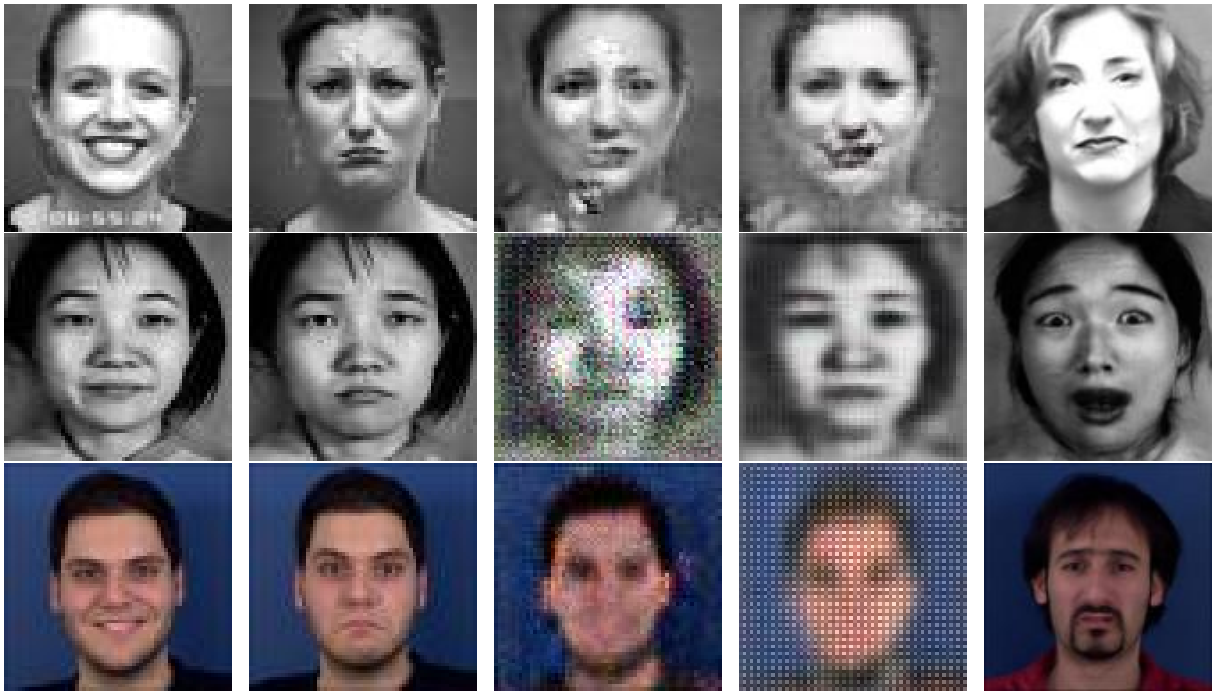
(a) (b) (c) (d) (e)

Figure B.11: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'fear'*** emotion, and c_j refers to the ***'sadness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



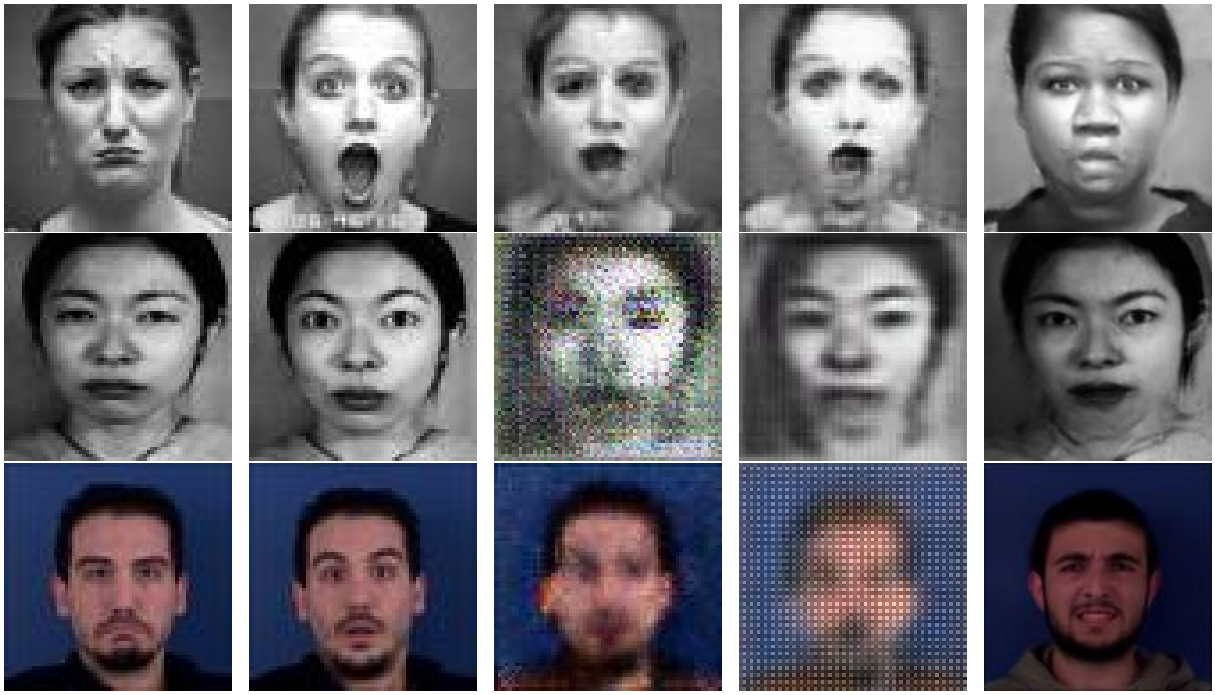
(a) (b) (c) (d) (e)

Figure B.12: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the *'**fear**'* emotion, and c_j refers to the *'**surprise**'* class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



(a) (b) (c) (d) (e)

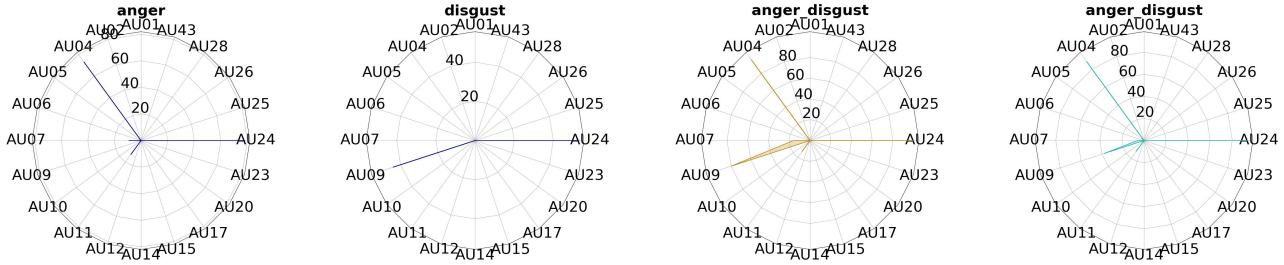
Figure B.13: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the ***'happiness'*** emotion, and c_j refers to the ***'sadness'*** class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).



(a) (b) (c) (d) (e)

Figure B.14: Synthesized images by the trained generator of the respective proposed models. Columns (a) and (b) refer to the image pair the generator takes such as $\mathbf{I}_{c_i}$ and $\mathbf{I}_{c_j}$ ($c_i \neq c_j$). In this figure, c_i indicates the *'**sadness**'* emotion, and c_j refers to the *'**surprise**'* class. Columns (c) and (d) are images synthesized by the generator of the corresponding models EmoGANs and EmoGANs1 with given input ($\mathbf{I}_{c_i}$, $\mathbf{I}_{c_j}$).

B.2 Full Evaluation Results of Conditional Emotion Generative Adversarial Networks (EmoGANs3) For Each Mixture of Emotions Class

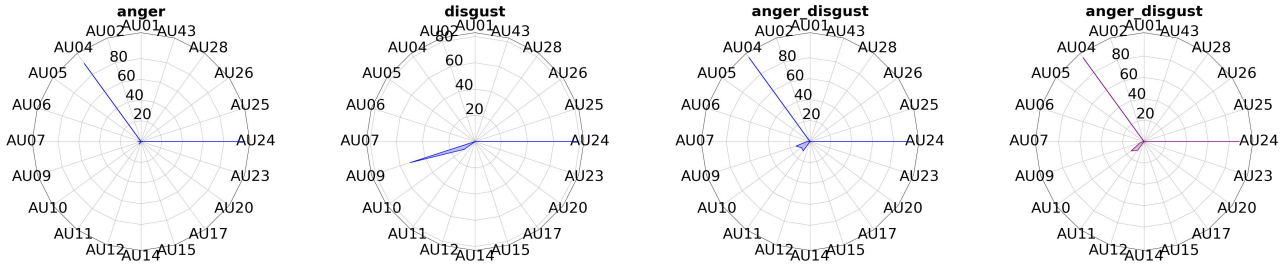


(a) Basic emotion for the '*anger*' class

(b) Basic emotion for the '*disgust*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

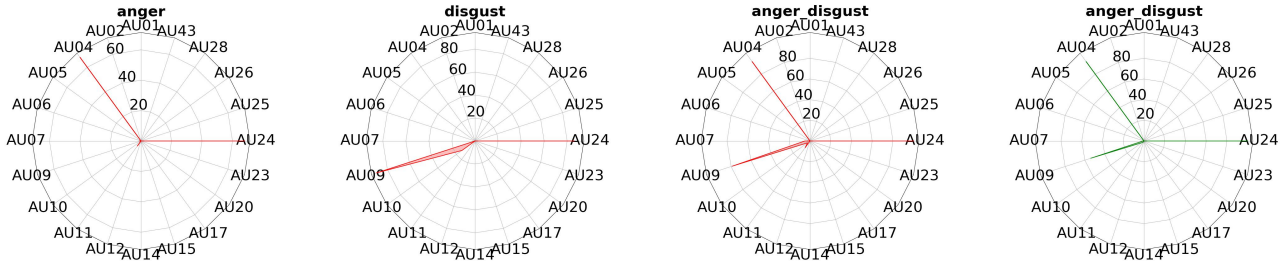


(e) Basic emotion for the '*anger*' class

(f) Basic emotion for the '*disgust*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



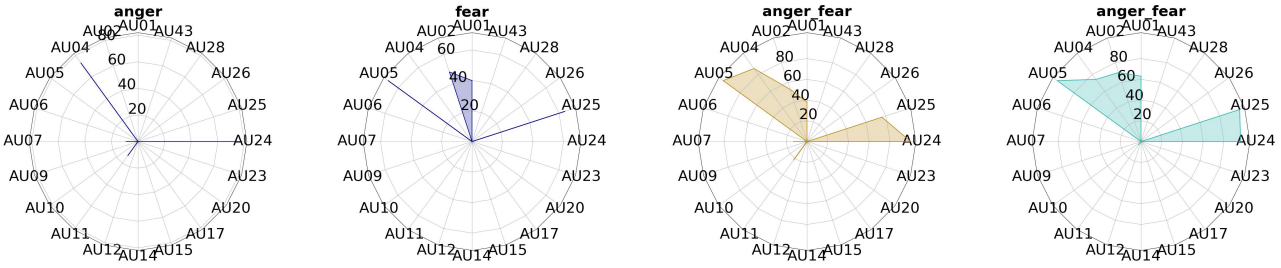
(i) Basic emotion for the '*anger*' class

(j) Basic emotion for the '*disgust*' class

(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

Figure B.15: Prototypical action units (AUs) for basic emotions ('***anger***' and '***disgust***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

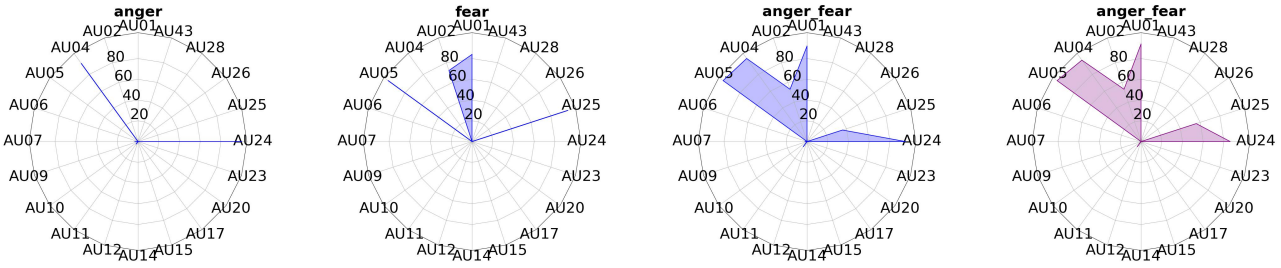


(a) Basic emotion for the '*anger*' class

(b) Basic emotion for the '*fear*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

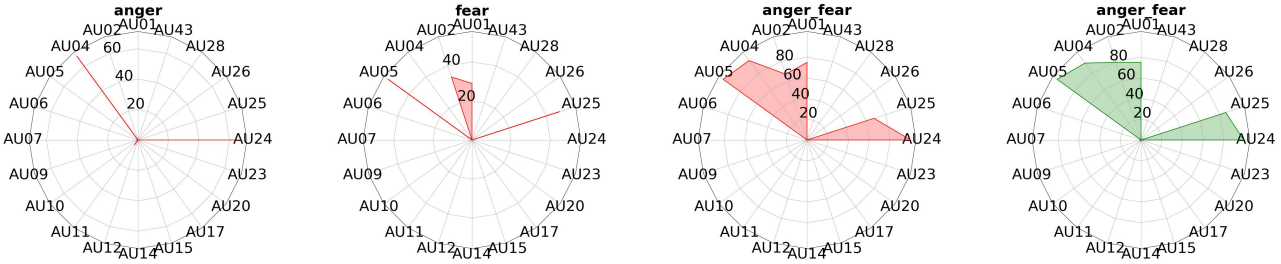


(e) Basic emotion for the '*anger*' class

(f) Basic emotion for the '*fear*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



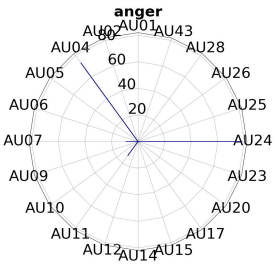
(i) Basic emotion for the '*anger*' class

(j) Basic emotion for the '*fear*' class

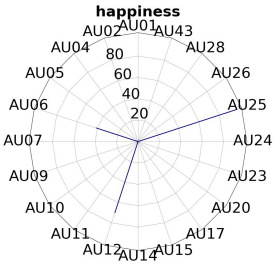
(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

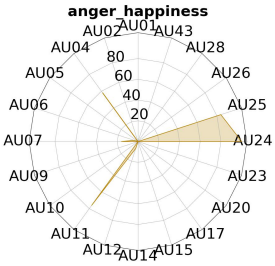
Figure B.16: Prototypical action units (AUs) for basic emotions ('***anger***' and '***fear***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.



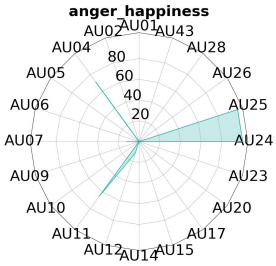
(a) Basic emotion for the '*anger*' class



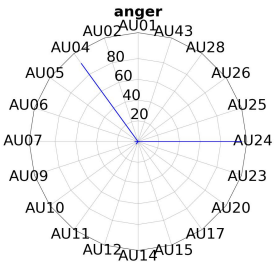
(b) Basic emotion for the '*happiness*' class



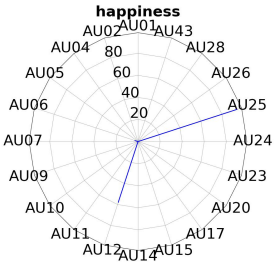
(c) Mixed emotion (training images)



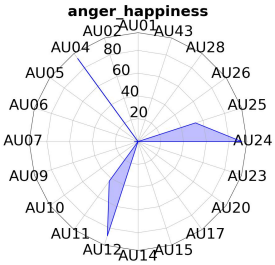
(d) Mixed emotion (generated images)



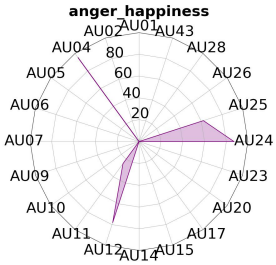
(e) Basic emotion for the '*anger*' class



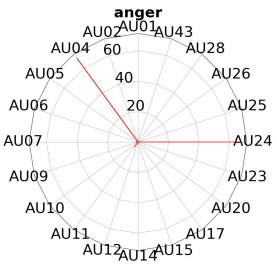
(f) Basic emotion for the '*happiness*' class



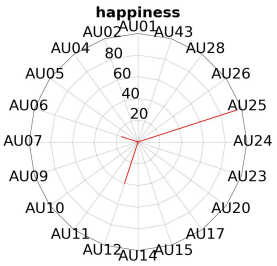
(g) Mixed emotion (training images)



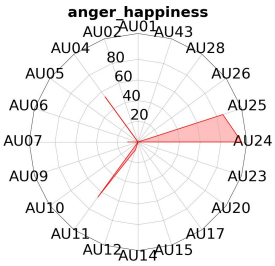
(h) Mixed emotion (generated images)



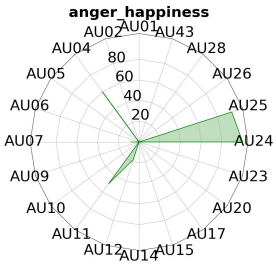
(i) Basic emotion for the '*anger*' class



(j) Basic emotion for the '*happiness*' class

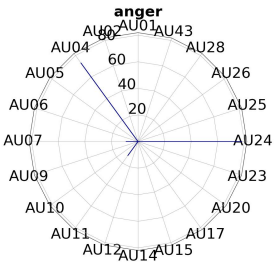


(k) Mixed emotion (training images)

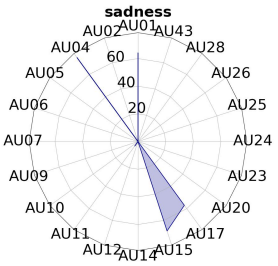


(l) Mixed emotion (generated images)

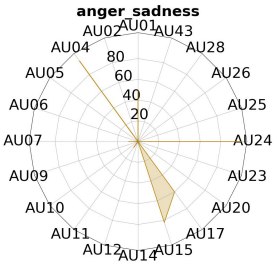
Figure B.17: Prototypical action units (AUs) for basic emotions ('***anger***' and '***happiness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.



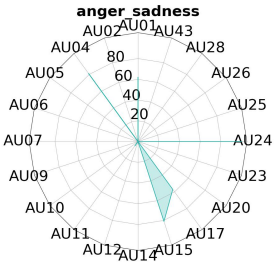
(a) Basic emotion for the '*anger*' class



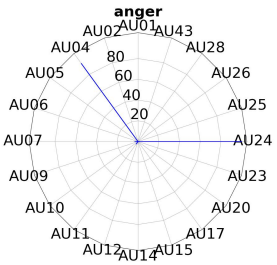
(b) Basic emotion for the '*sadness*' class



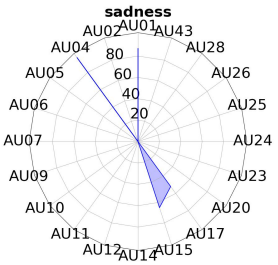
(c) Mixed emotion (training images)



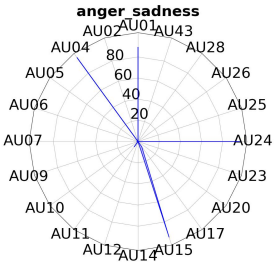
(d) Mixed emotion (generated images)



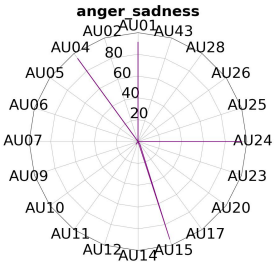
(e) Basic emotion for the '*anger*' class



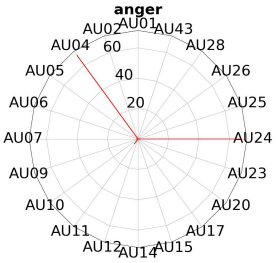
(f) Basic emotion for the '*sadness*' class



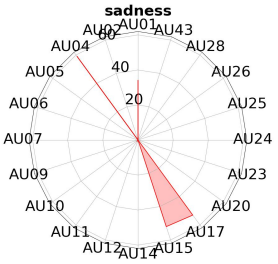
(g) Mixed emotion (training images)



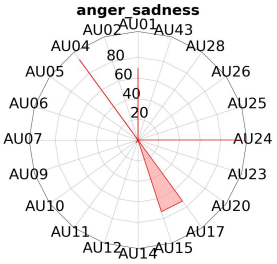
(h) Mixed emotion (generated images)



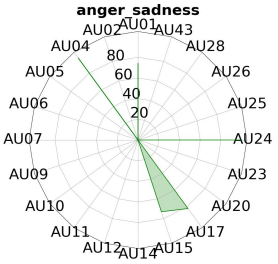
(i) Basic emotion for the '*anger*' class



(j) Basic emotion for the '*sadness*' class

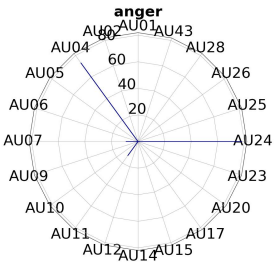


(k) Mixed emotion (training images)

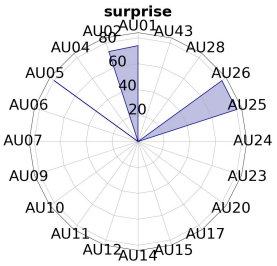


(l) Mixed emotion (generated images)

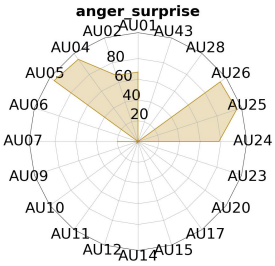
Figure B.18: Prototypical action units (AUs) for basic emotions ('***anger***' and '***sadness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.



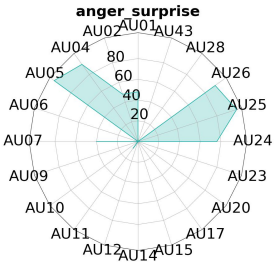
(a) Basic emotion for the '*anger*' class



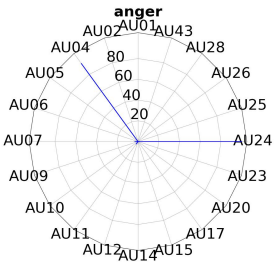
(b) Basic emotion for the '*surprise*' class



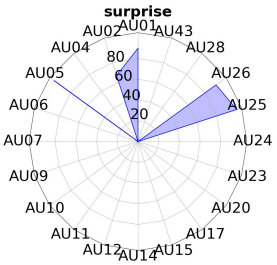
(c) Mixed emotion (training images)



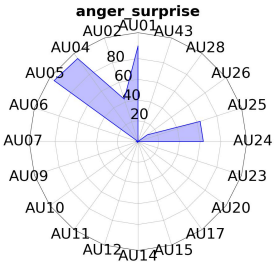
(d) Mixed emotion (generated images)



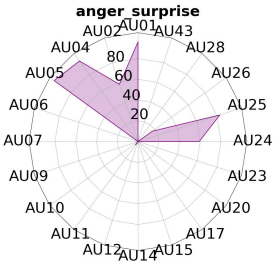
(e) Basic emotion for the '*anger*' class



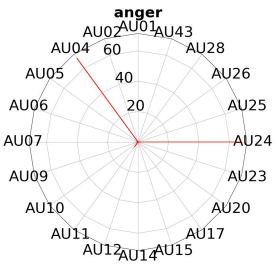
(f) Basic emotion for the '*surprise*' class



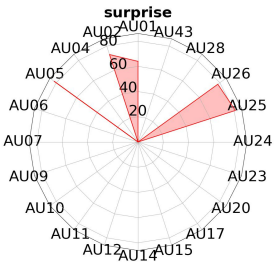
(g) Mixed emotion (training images)



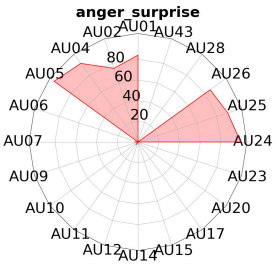
(h) Mixed emotion (generated images)



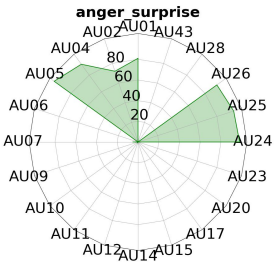
(i) Basic emotion for the '*anger*' class



(j) Basic emotion for the '*surprise*' class

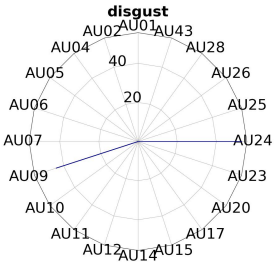


(k) Mixed emotion (training images)

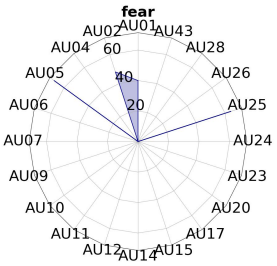


(l) Mixed emotion (generated images)

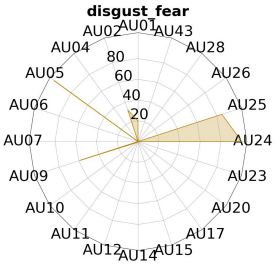
Figure B.19: Prototypical action units (AUs) for basic emotions ('***anger***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.



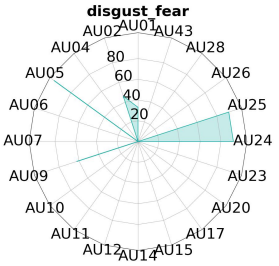
(a) Basic emotion for the '*disgust*' class



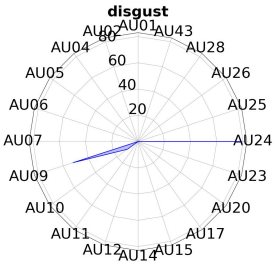
(b) Basic emotion for the '*fear*' class



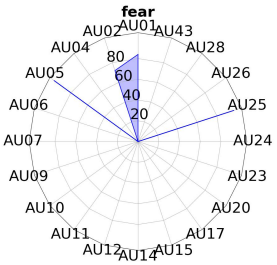
(c) Mixed emotion (training images)



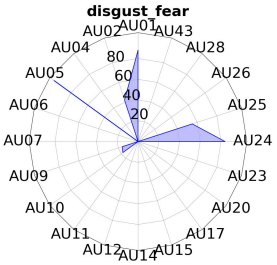
(d) Mixed emotion (generated images)



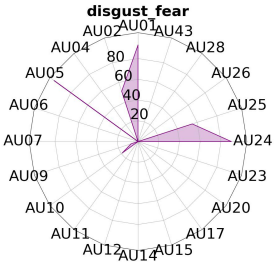
(e) Basic emotion for the '*disgust*' class



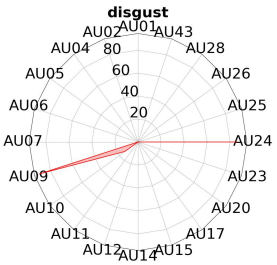
(f) Basic emotion for the '*fear*' class



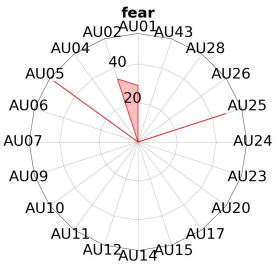
(g) Mixed emotion (training images)



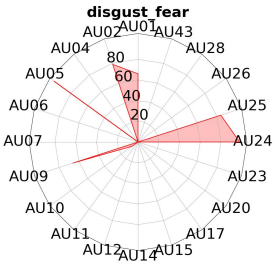
(h) Mixed emotion (generated images)



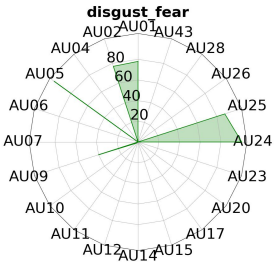
(i) Basic emotion for the '*disgust*' class



(j) Basic emotion for the '*fear*' class

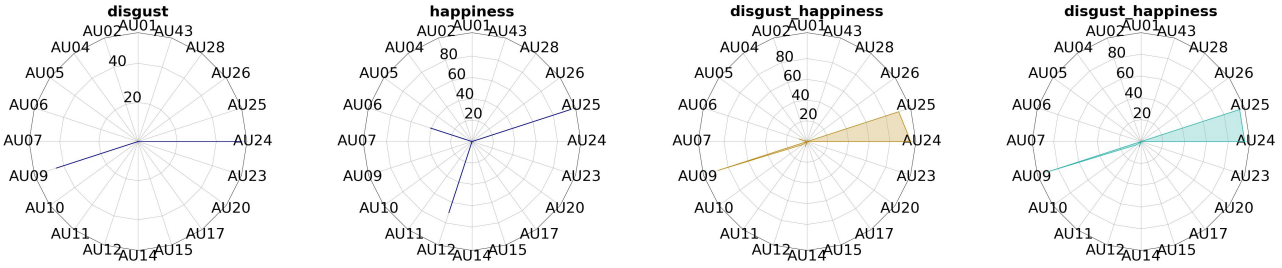


(k) Mixed emotion (training images)



(l) Mixed emotion (generated images)

Figure B.20: Prototypical action units (AUs) for basic emotions ('***disgust***' and '***fear***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

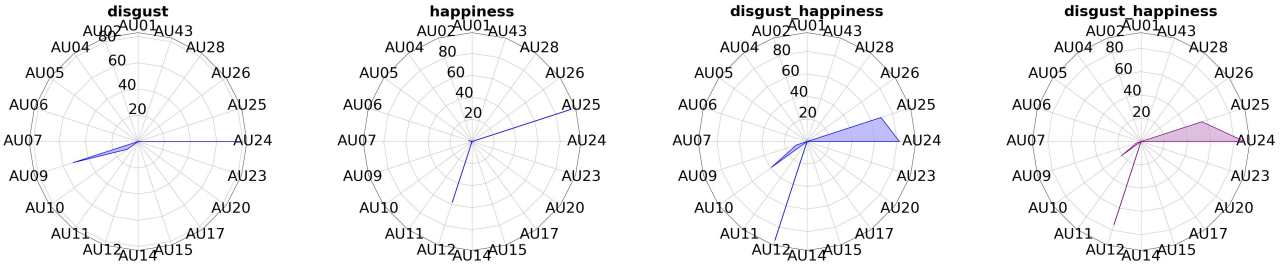


(a) Basic emotion for the '*disgust*' class

(b) Basic emotion for the '*happiness*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

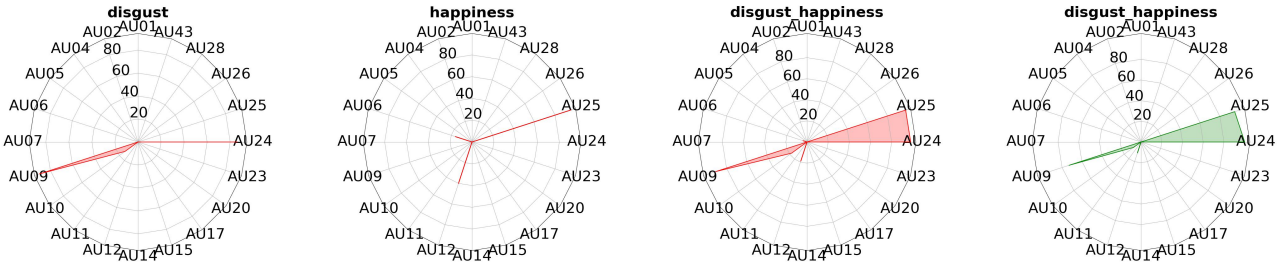


(e) Basic emotion for the '*disgust*' class

(f) Basic emotion for the '*happiness*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



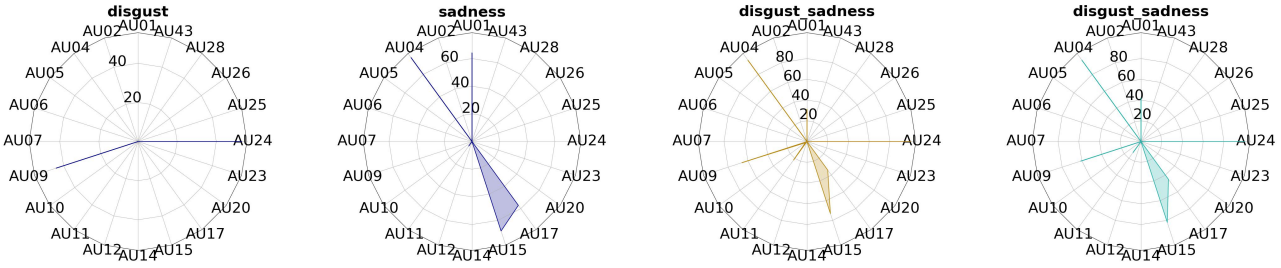
(i) Basic emotion for the '*disgust*' class

(j) Basic emotion for the '*happiness*' class

(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

Figure B.21: Prototypical action units (AUs) for basic emotions ('***disgust***' and '***happiness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

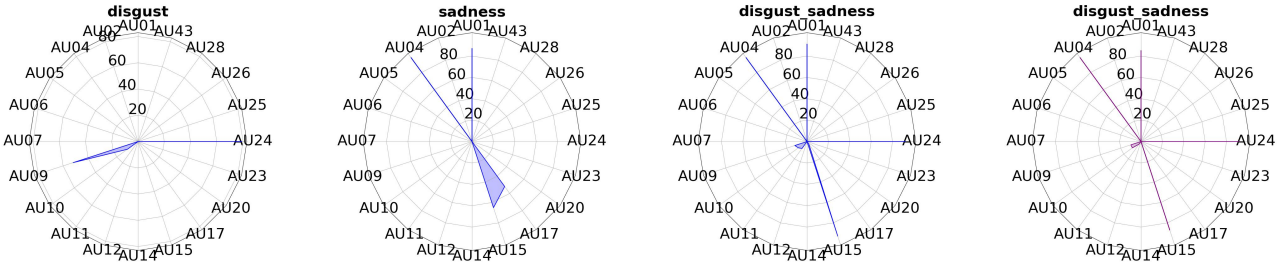


(a) Basic emotion for the '*disgust*' class

(b) Basic emotion for the '*sadness*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

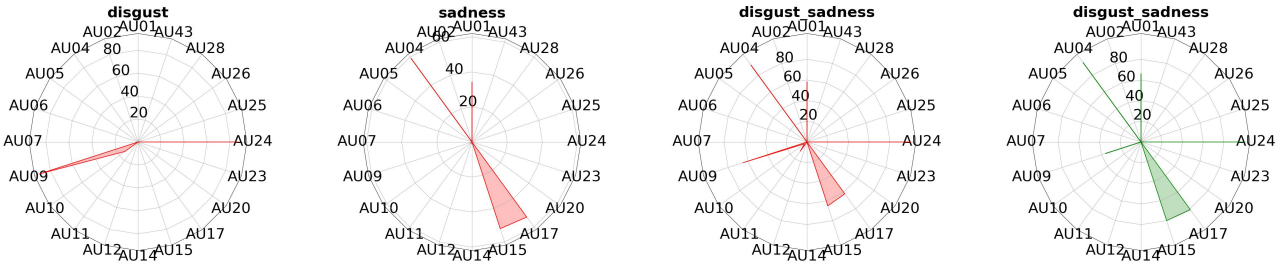


(e) Basic emotion for the '*disgust*' class

(f) Basic emotion for the '*sadness*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



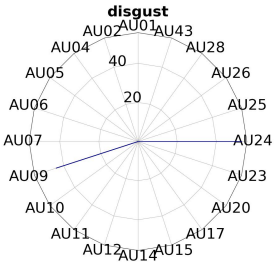
(i) Basic emotion for the '*disgust*' class

(j) Basic emotion for the '*sadness*' class

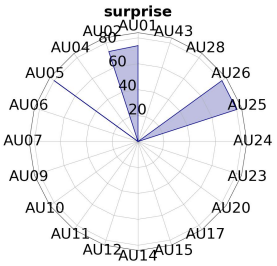
(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

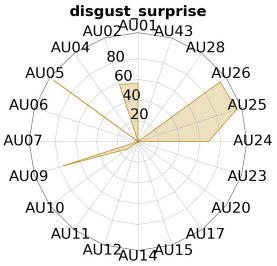
Figure B.22: Prototypical action units (AUs) for basic emotions ('***disgust***' and '***sadness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.



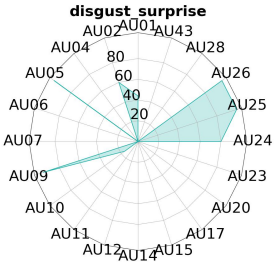
(a) Basic emotion for the '*disgust*' class



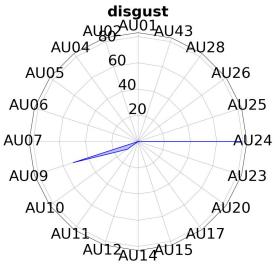
(b) Basic emotion for the '*surprise*' class



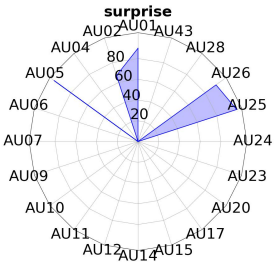
(c) Mixed emotion (training images)



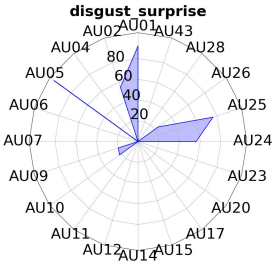
(d) Mixed emotion (generated images)



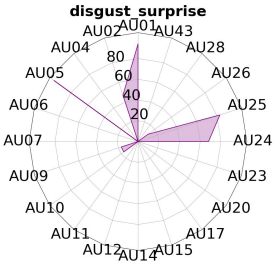
(e) Basic emotion for the '*disgust*' class



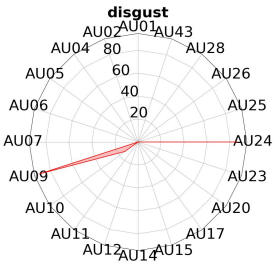
(f) Basic emotion for the '*surprise*' class



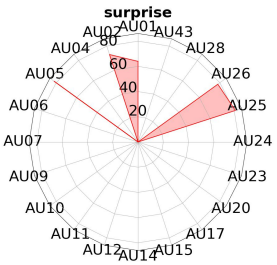
(g) Mixed emotion (training images)



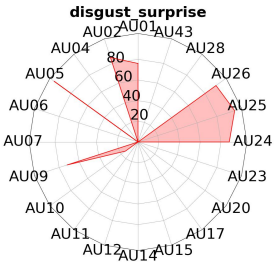
(h) Mixed emotion (generated images)



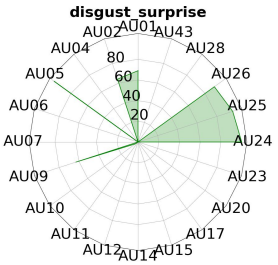
(i) Basic emotion for the '*disgust*' class



(j) Basic emotion for the '*surprise*' class

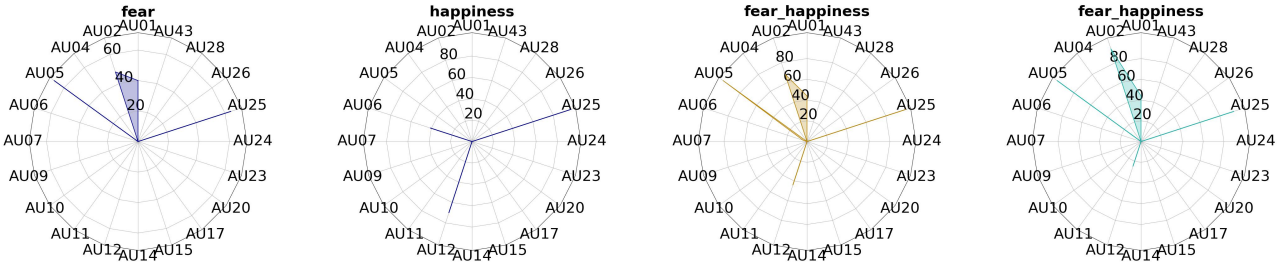


(k) Mixed emotion (training images)



(l) Mixed emotion (generated images)

Figure B.23: Prototypical action units (AUs) for basic emotions ('***disgust***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

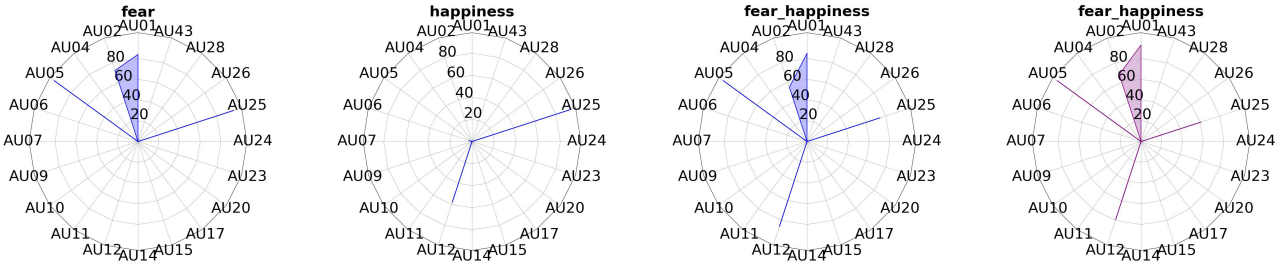


(a) Basic emotion for the '*fear*' class

(b) Basic emotion for the '*happiness*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

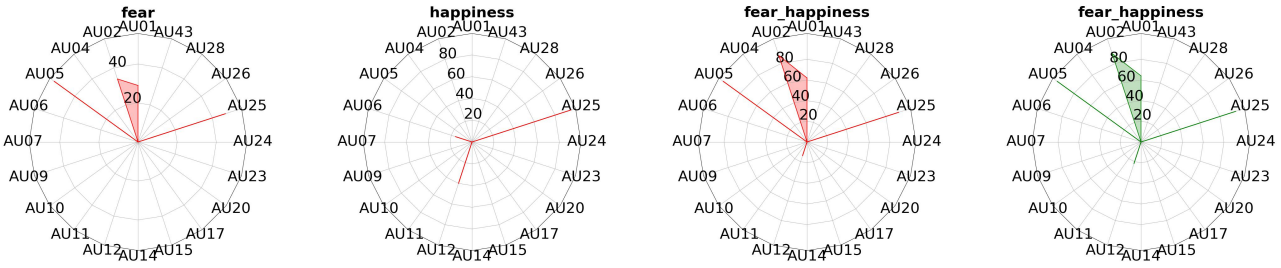


(e) Basic emotion for the '*fear*' class

(f) Basic emotion for the '*happiness*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



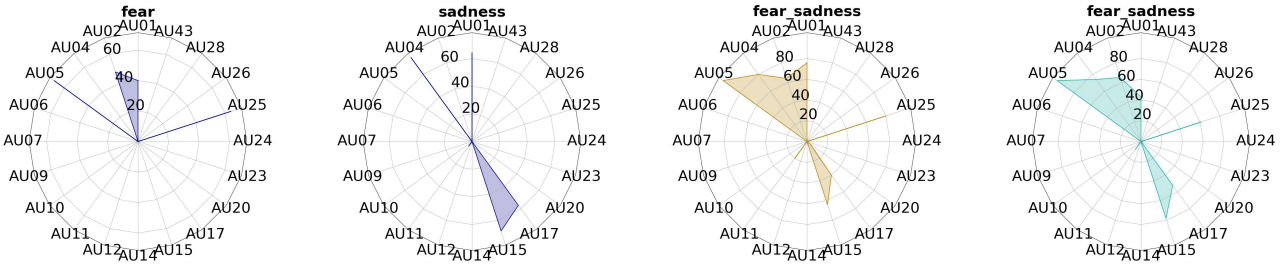
(i) Basic emotion for the '*fear*' class

(j) Basic emotion for the '*happiness*' class

(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

Figure B.24: Prototypical action units (AUs) for basic emotions ('***fear***' and '***happiness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

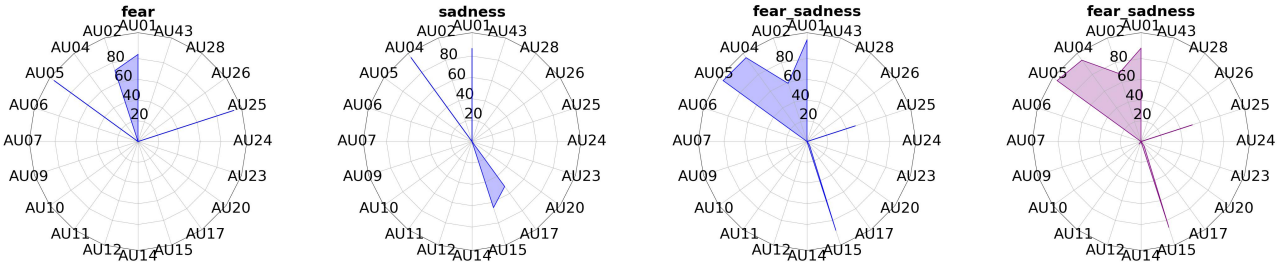


(a) Basic emotion for the '*fear*' class

(b) Basic emotion for the '*sadness*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

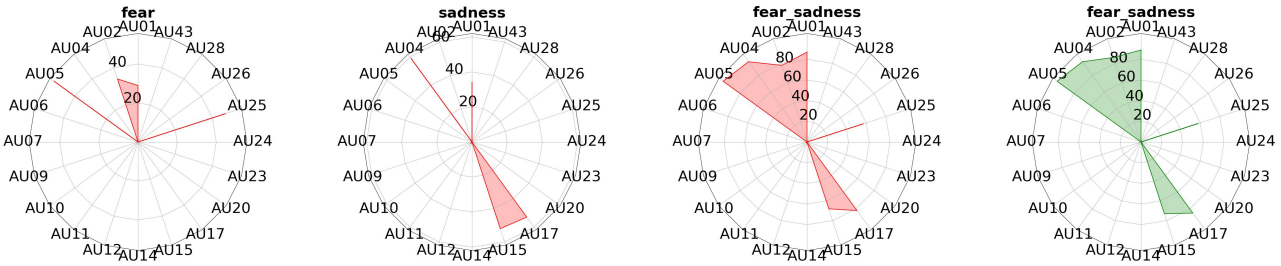


(e) Basic emotion for the '*fear*' class

(f) Basic emotion for the '*sadness*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



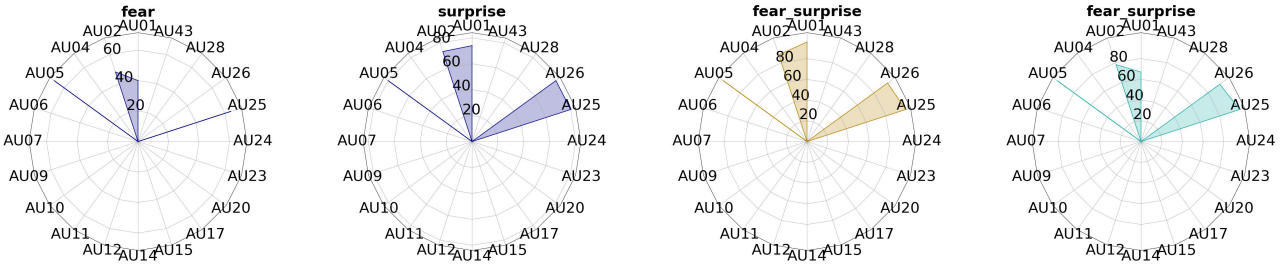
(i) Basic emotion for the '*fear*' class

(j) Basic emotion for the '*sadness*' class

(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

Figure B.25: Prototypical action units (AUs) for basic emotions ('***fear***' and '***sadness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

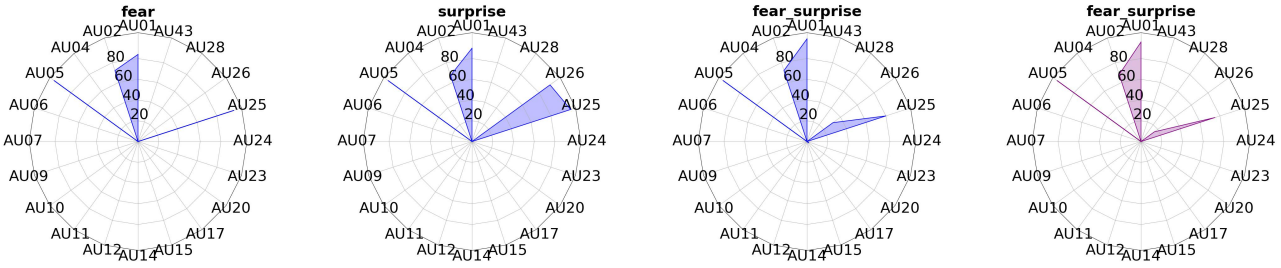


(a) Basic emotion for the '*fear*' class

(b) Basic emotion for the '*surprise*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

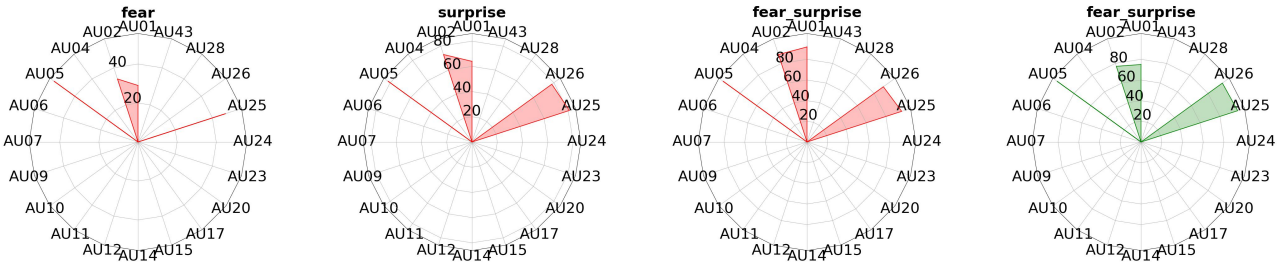


(e) Basic emotion for the '*fear*' class

(f) Basic emotion for the '*surprise*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



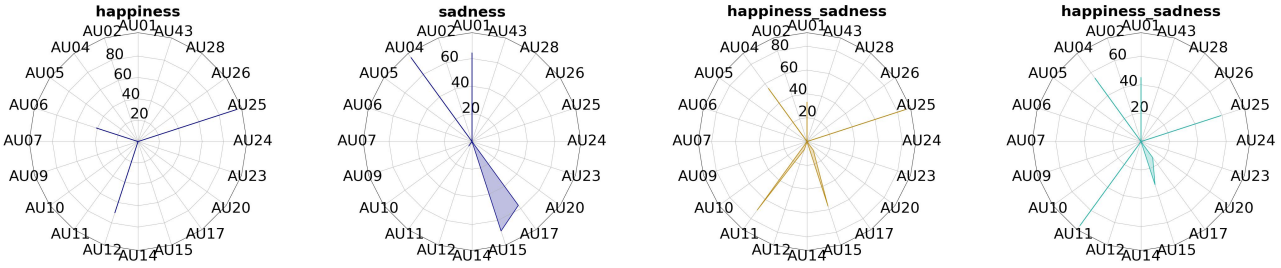
(i) Basic emotion for the '*fear*' class

(j) Basic emotion for the '*surprise*' class

(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

Figure B.26: Prototypical action units (AUs) for basic emotions ('***fear***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

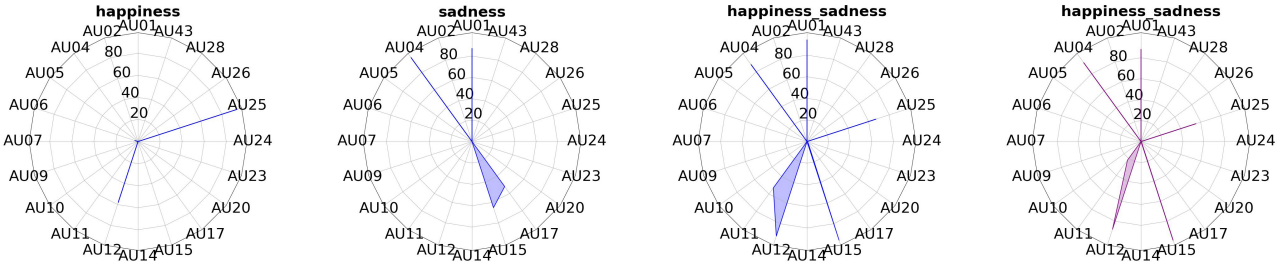


(a) Basic emotion for the '*happiness*' class

(b) Basic emotion for the '*sadness*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

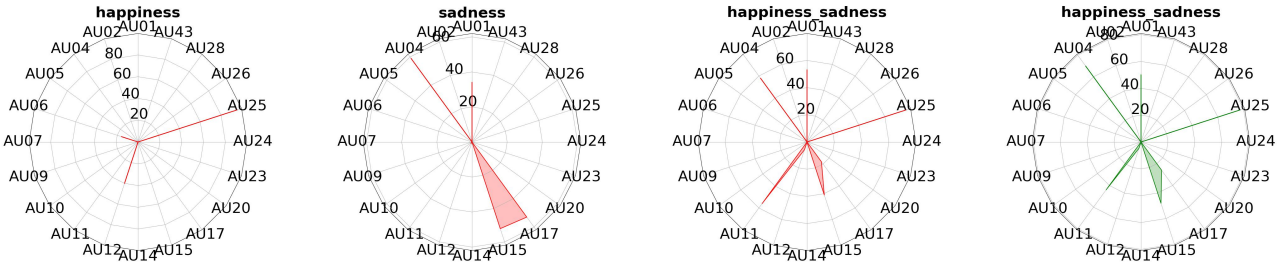


(e) Basic emotion for the '*happiness*' class

(f) Basic emotion for the '*sadness*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



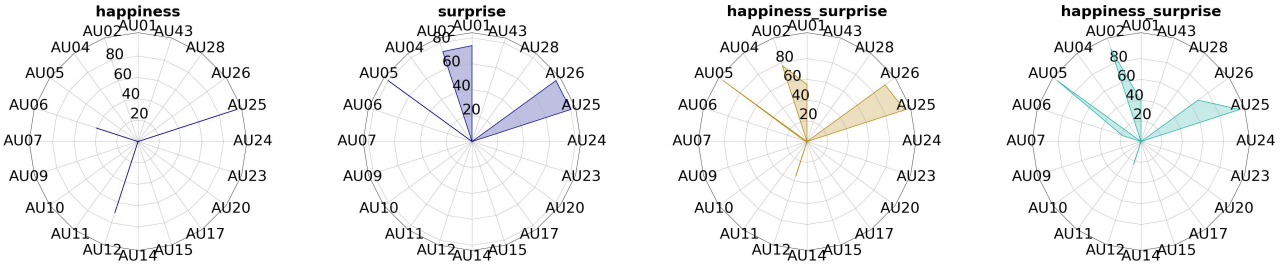
(i) Basic emotion for the '*happiness*' class

(j) Basic emotion for the '*sadness*' class

(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

Figure B.27: Prototypical action units (AUs) for basic emotions ('***happiness***' and '***sadness***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

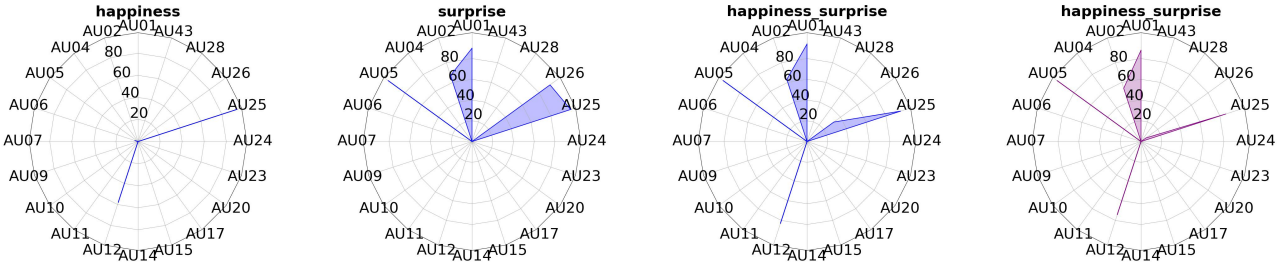


(a) Basic emotion for the '*happiness*' class

(b) Basic emotion for the '*surprise*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

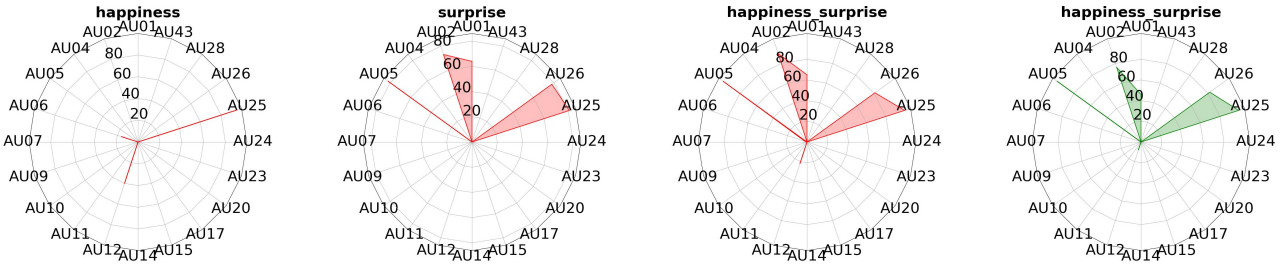


(e) Basic emotion for the '*happiness*' class

(f) Basic emotion for the '*surprise*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



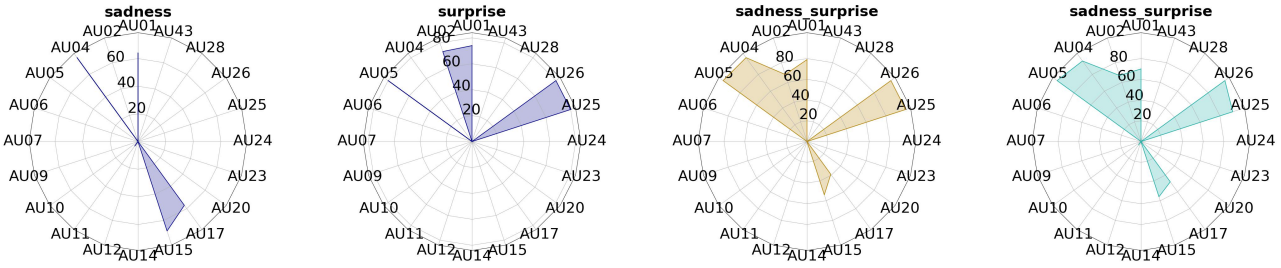
(i) Basic emotion for the '*happiness*' class

(j) Basic emotion for the '*surprise*' class

(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

Figure B.28: Prototypical action units (AUs) for basic emotions ('***happiness***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

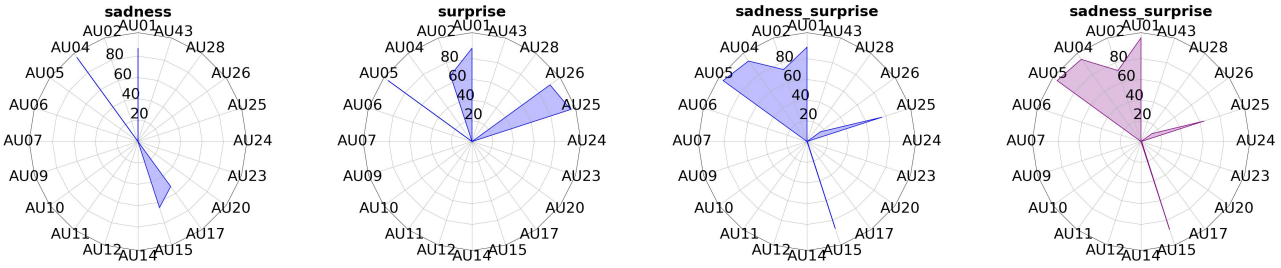


(a) Basic emotion for the '*sadness*' class

(b) Basic emotion for the '*surprise*' class

(c) Mixed emotion (training images)

(d) Mixed emotion (generated images)

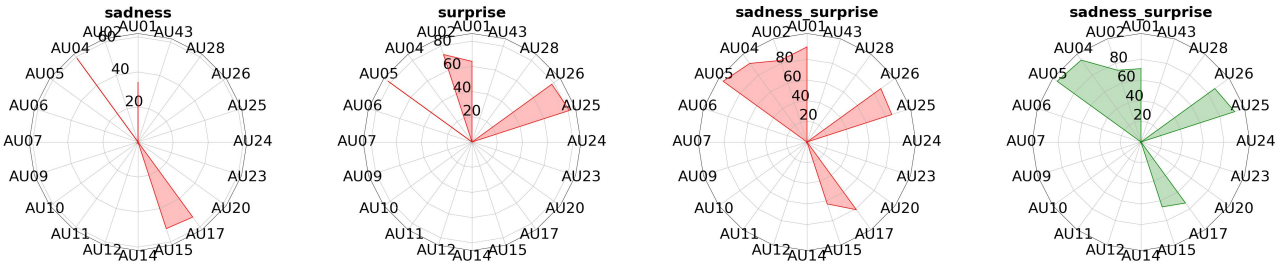


(e) Basic emotion for the '*sadness*' class

(f) Basic emotion for the '*surprise*' class

(g) Mixed emotion (training images)

(h) Mixed emotion (generated images)



(i) Basic emotion for the '*sadness*' class

(j) Basic emotion for the '*surprise*' class

(k) Mixed emotion (training images)

(l) Mixed emotion (generated images)

Figure B.29: Prototypical action units (AUs) for basic emotions ('***sadness***' and '***surprise***' in this figure) and a mixture of emotions among them. The circumference axis refers to the AUs number and number of images with prototypical AUs in %. The upper row indicates the result for the CK dataset. The middle row is for JAFFE and the lower row is for the MUG dataset. Morphing images are used during training.

B.3 Complete Experimental Results for Manipulation of Facial Expressions Attributes in Latent Space

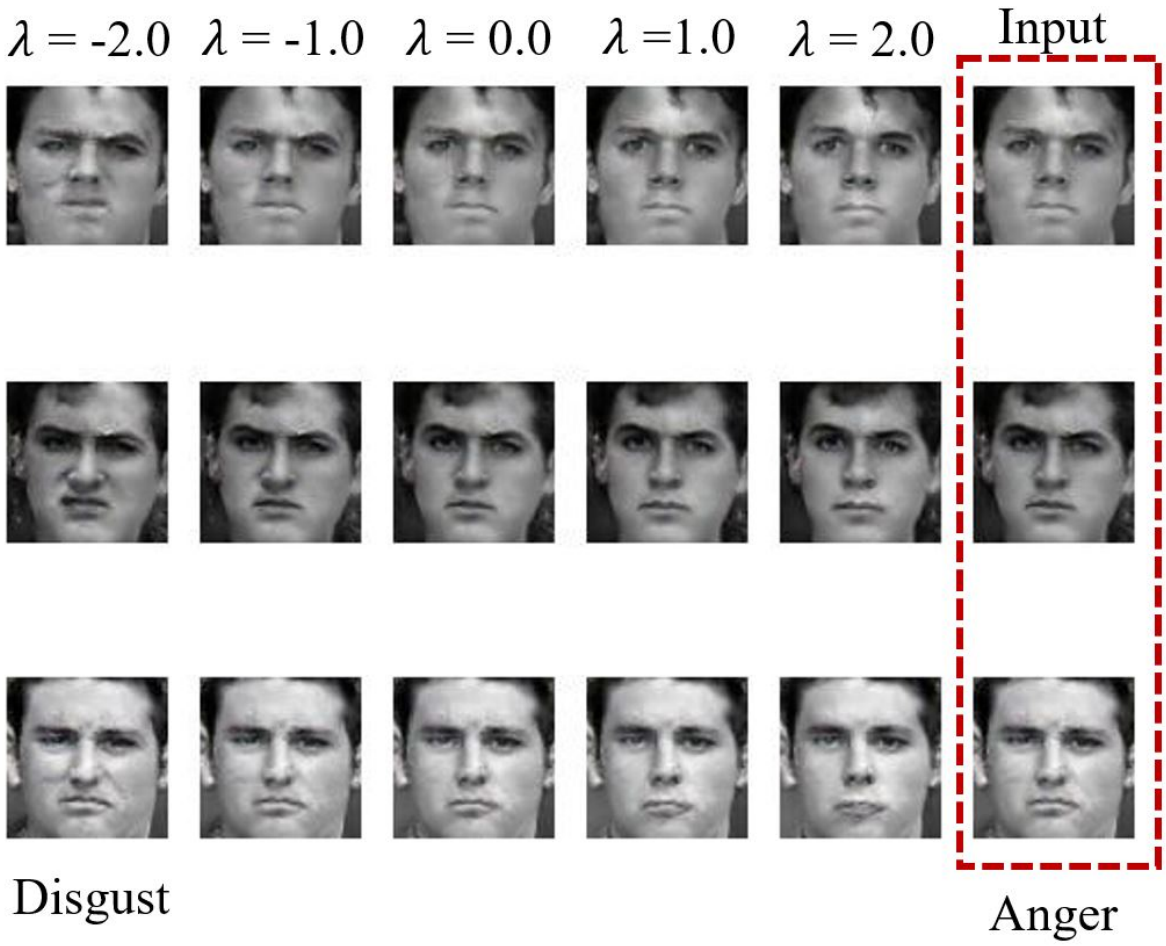


Figure B.30: An example output of facial expressions manipulation from '***anger***' class to '***disgust***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

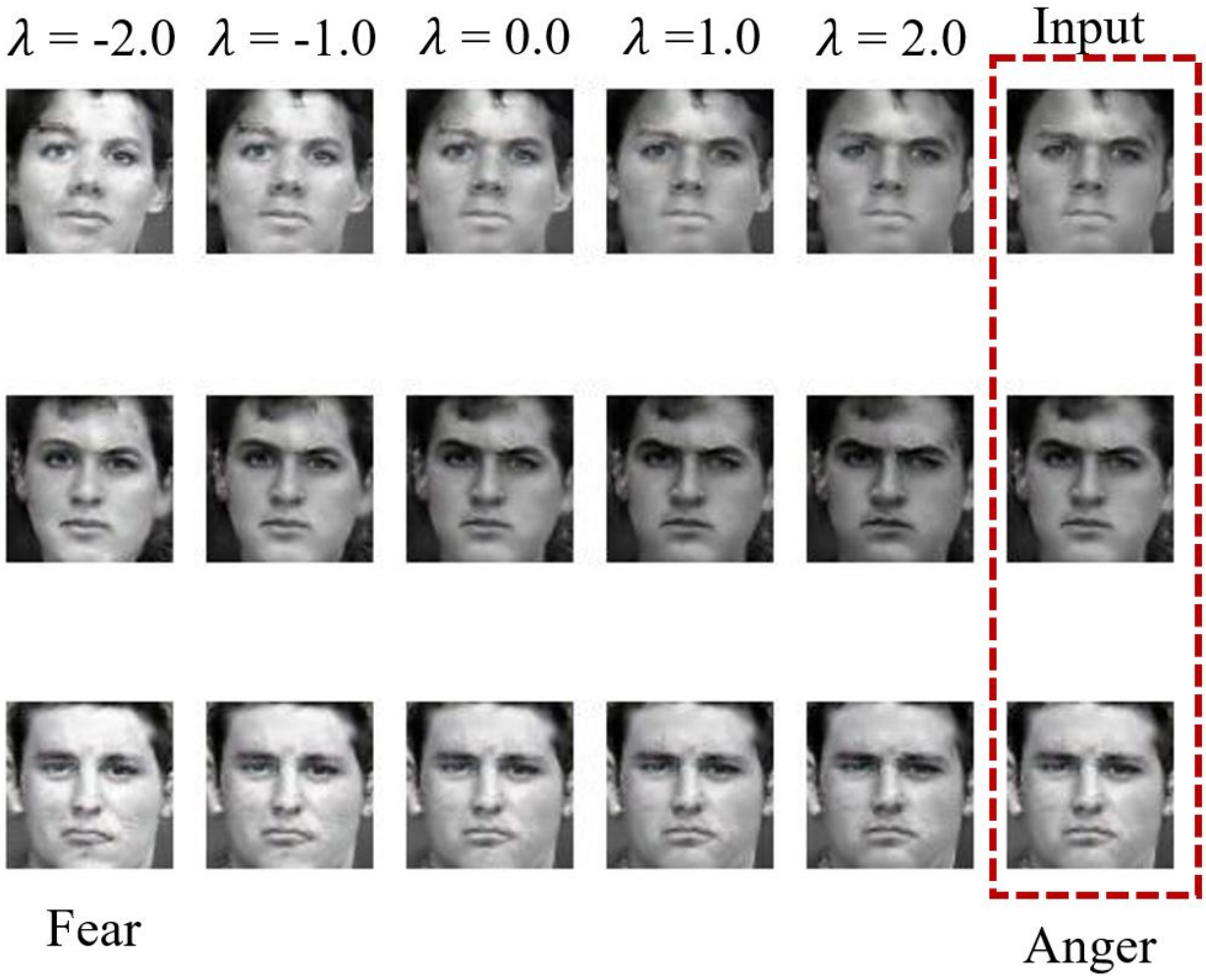


Figure B.31: An example output of facial expressions manipulation from '***anger***' class to '***fear***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

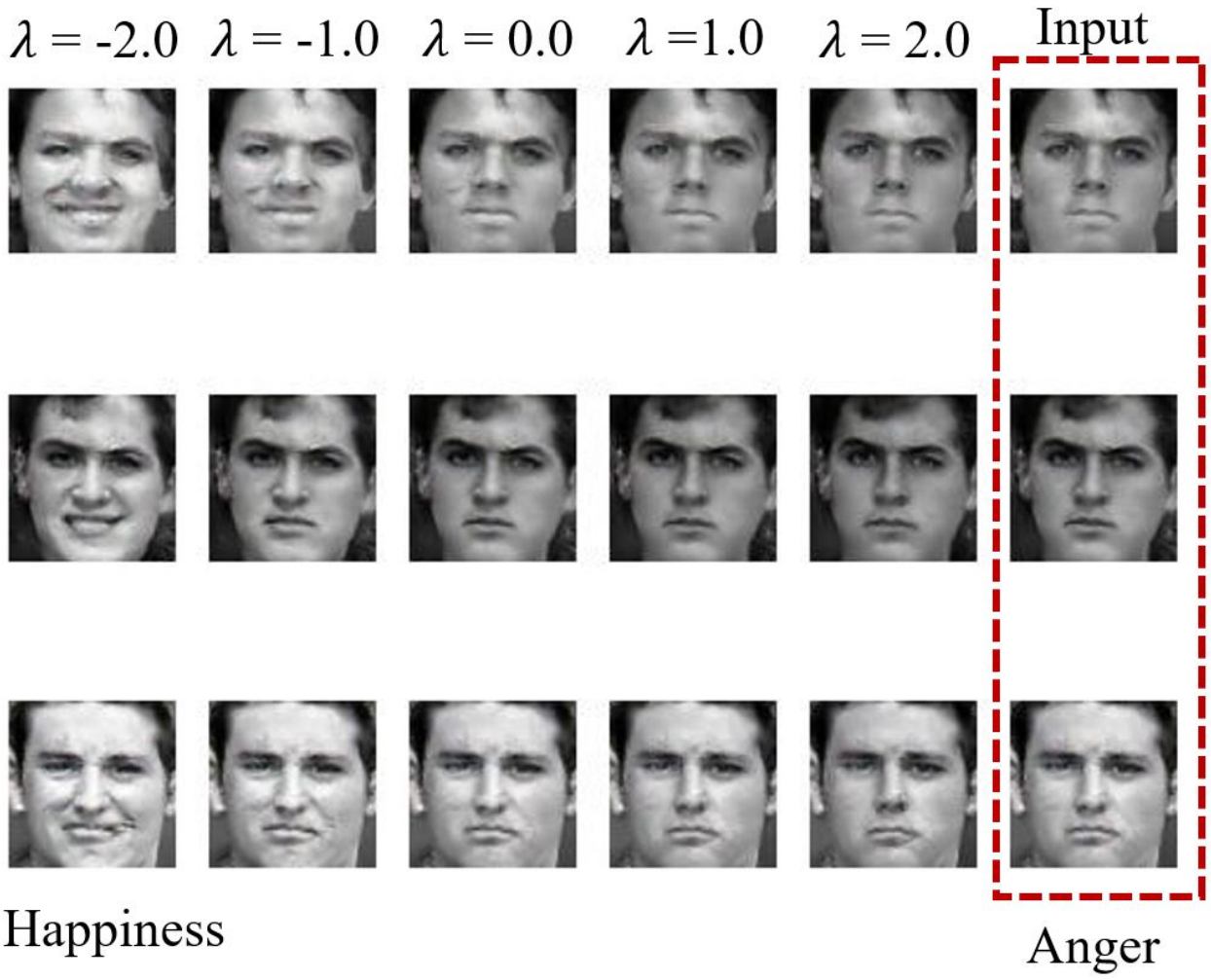


Figure B.32: An example output of facial expressions manipulation from '***anger***' class to '***happiness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

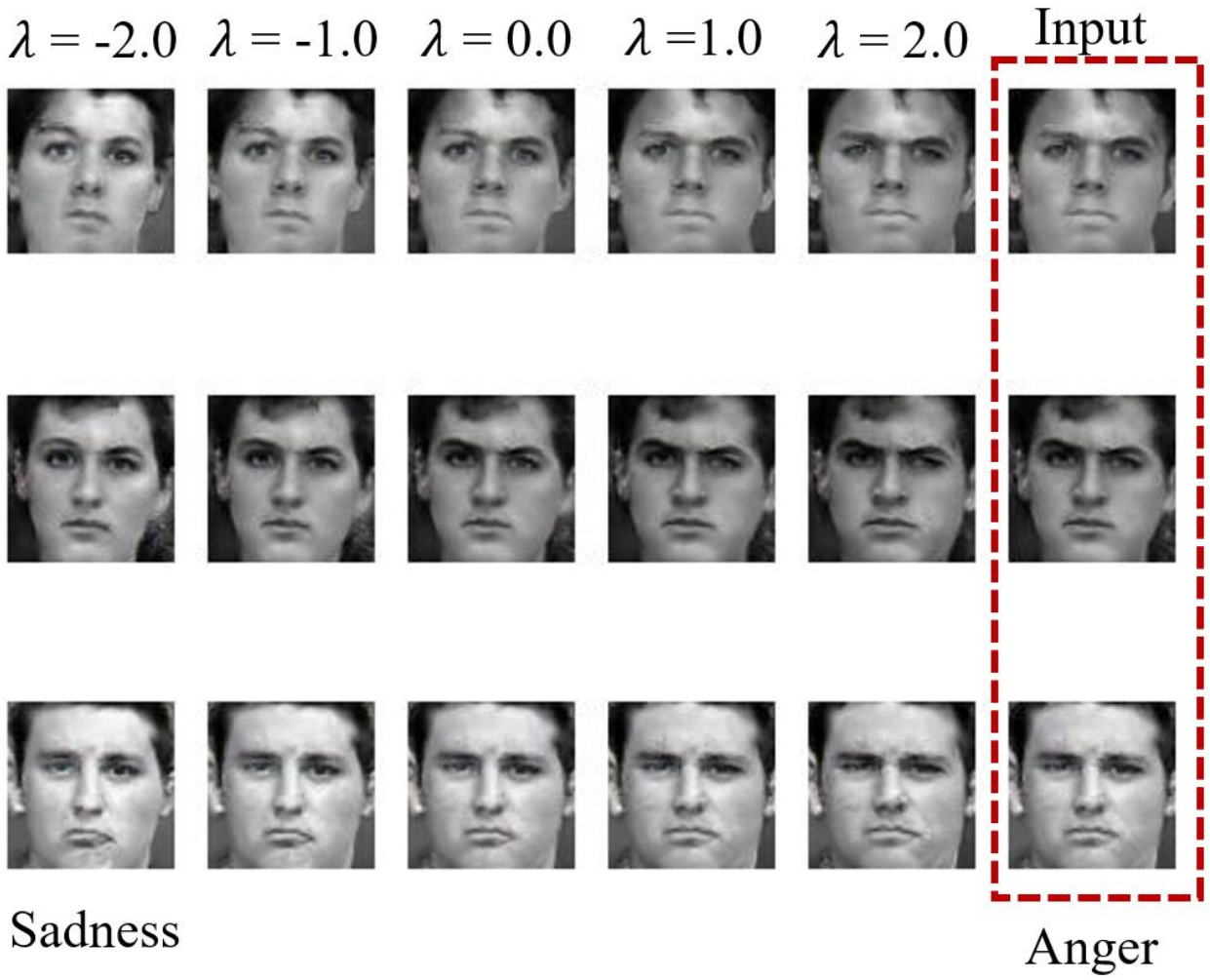


Figure B.33: An example output of facial expressions manipulation from '***anger***' class to '***sadness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

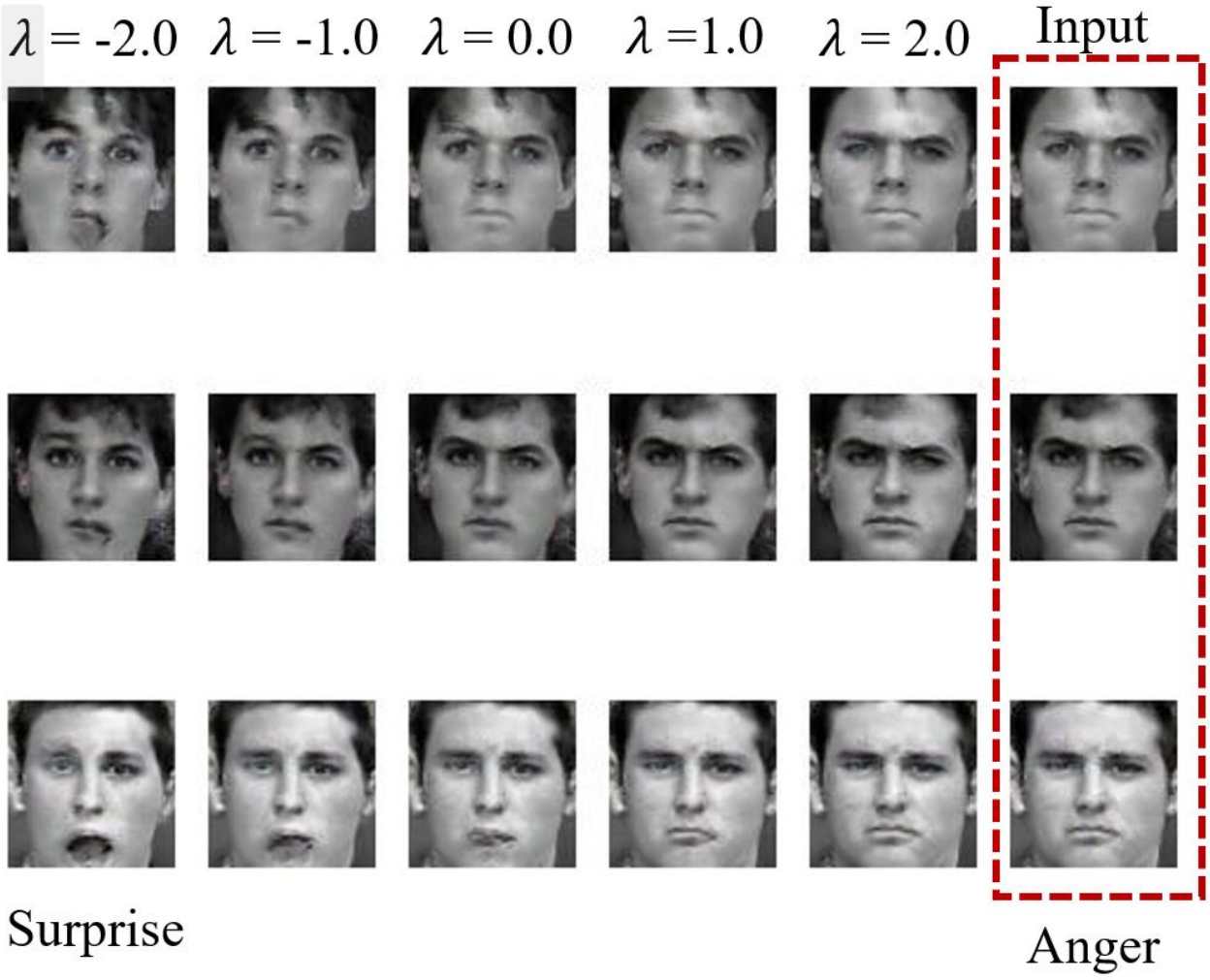


Figure B.34: An example output of facial expressions manipulation from '***anger***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

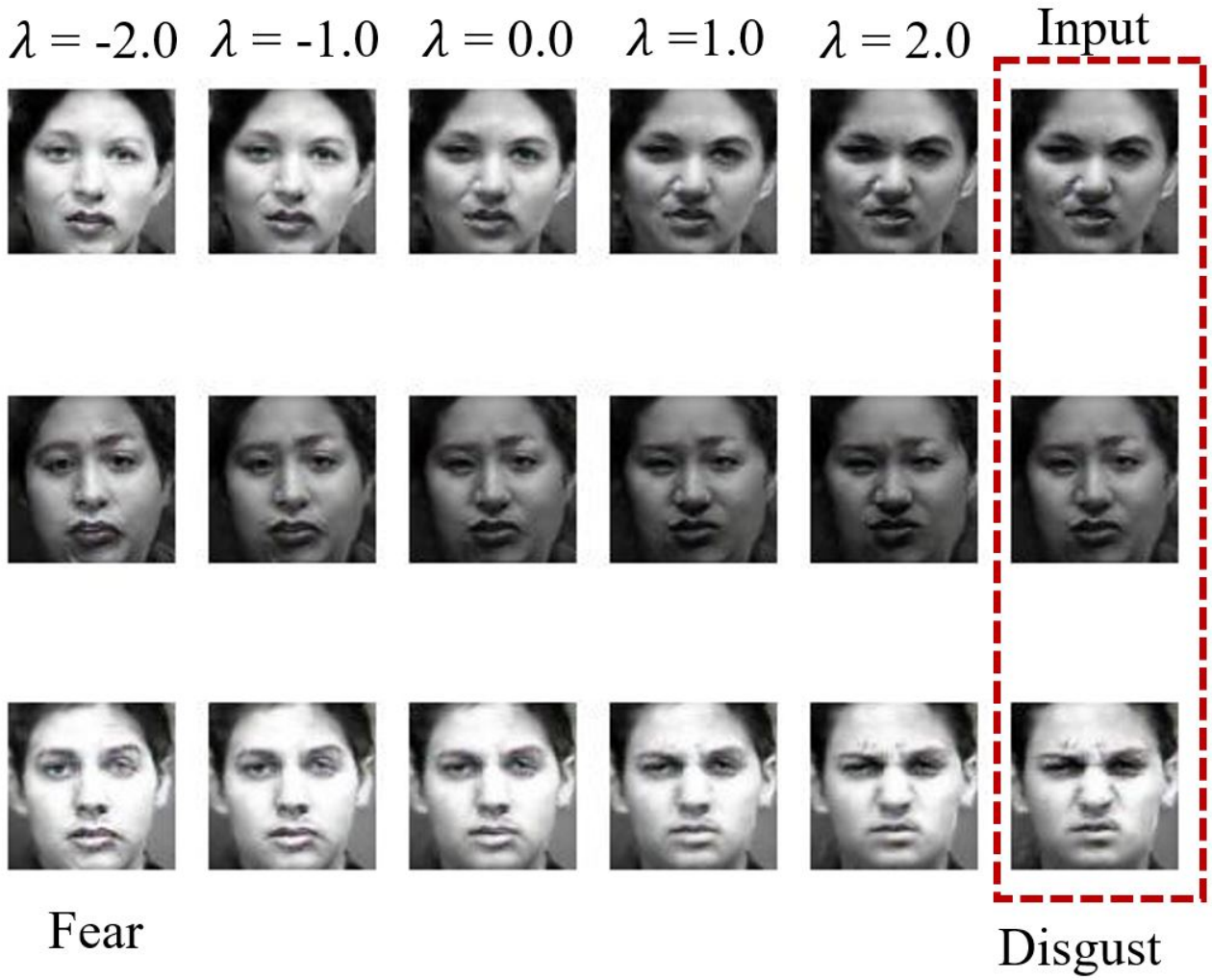


Figure B.35: An example output of facial expressions manipulation from '***disgust***' class to '***fear***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

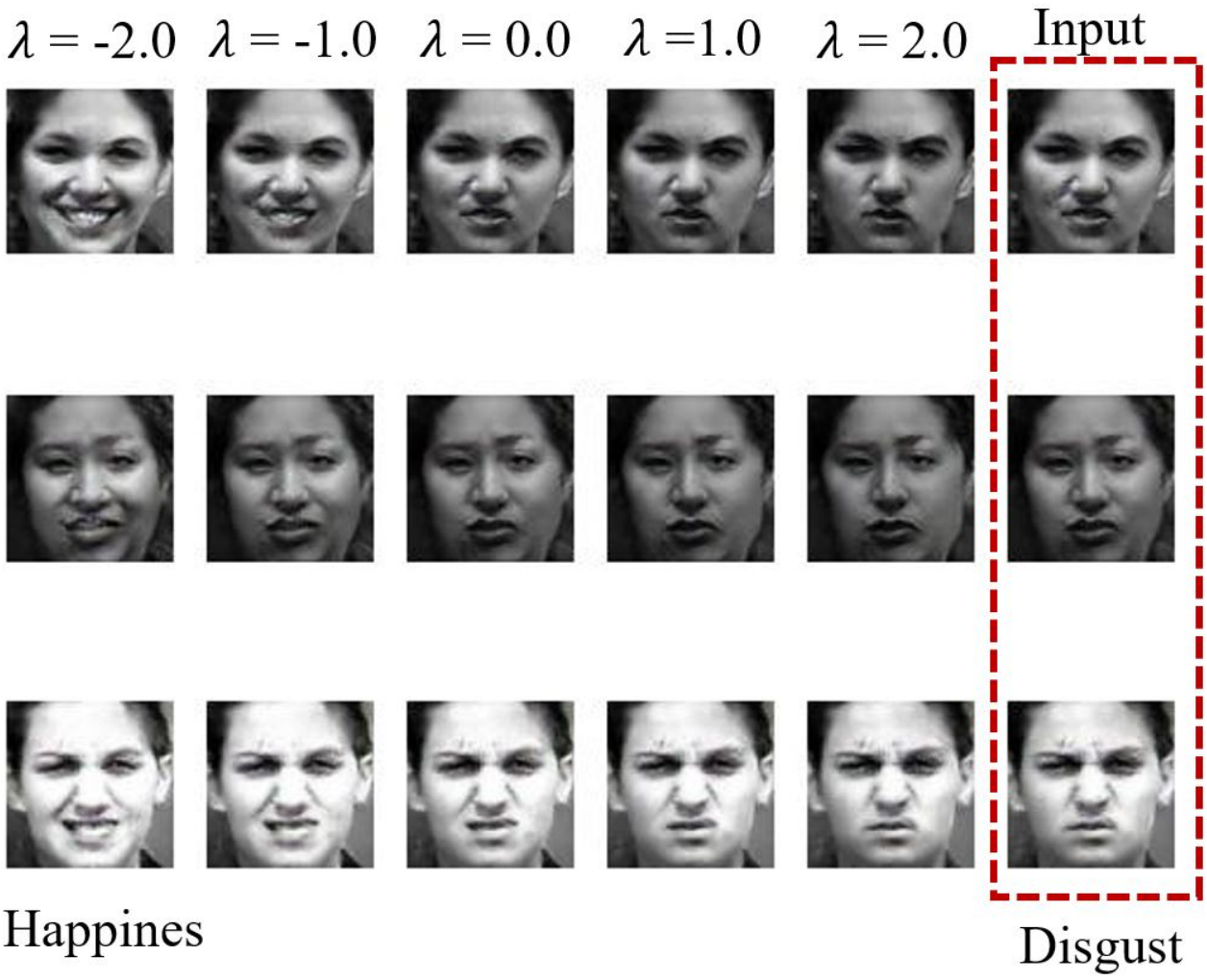


Figure B.36: An example output of facial expressions manipulation from '***disgust***' class to '***happiness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

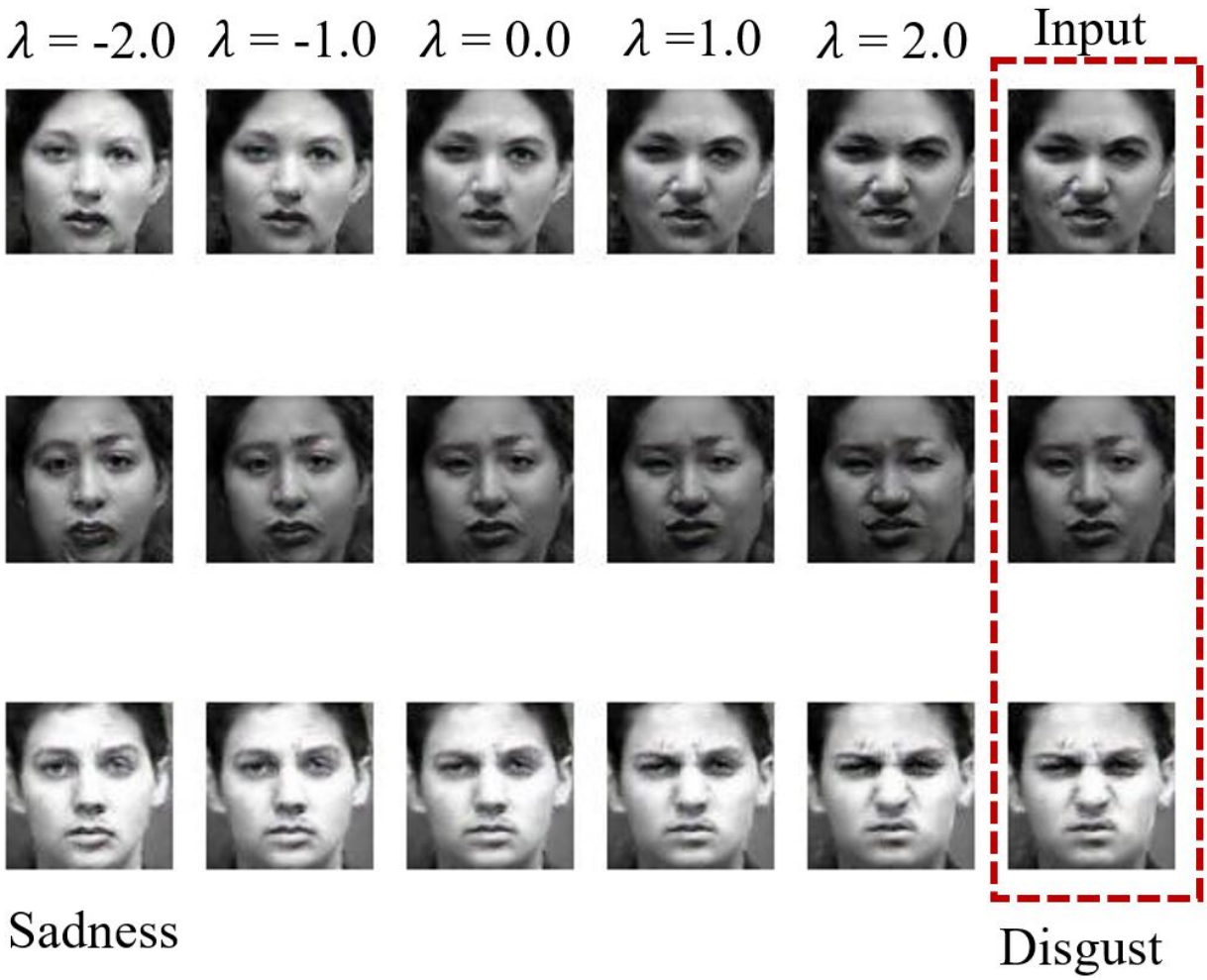


Figure B.37: An example output of facial expressions manipulation from '***disgust***' class to '***sadness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

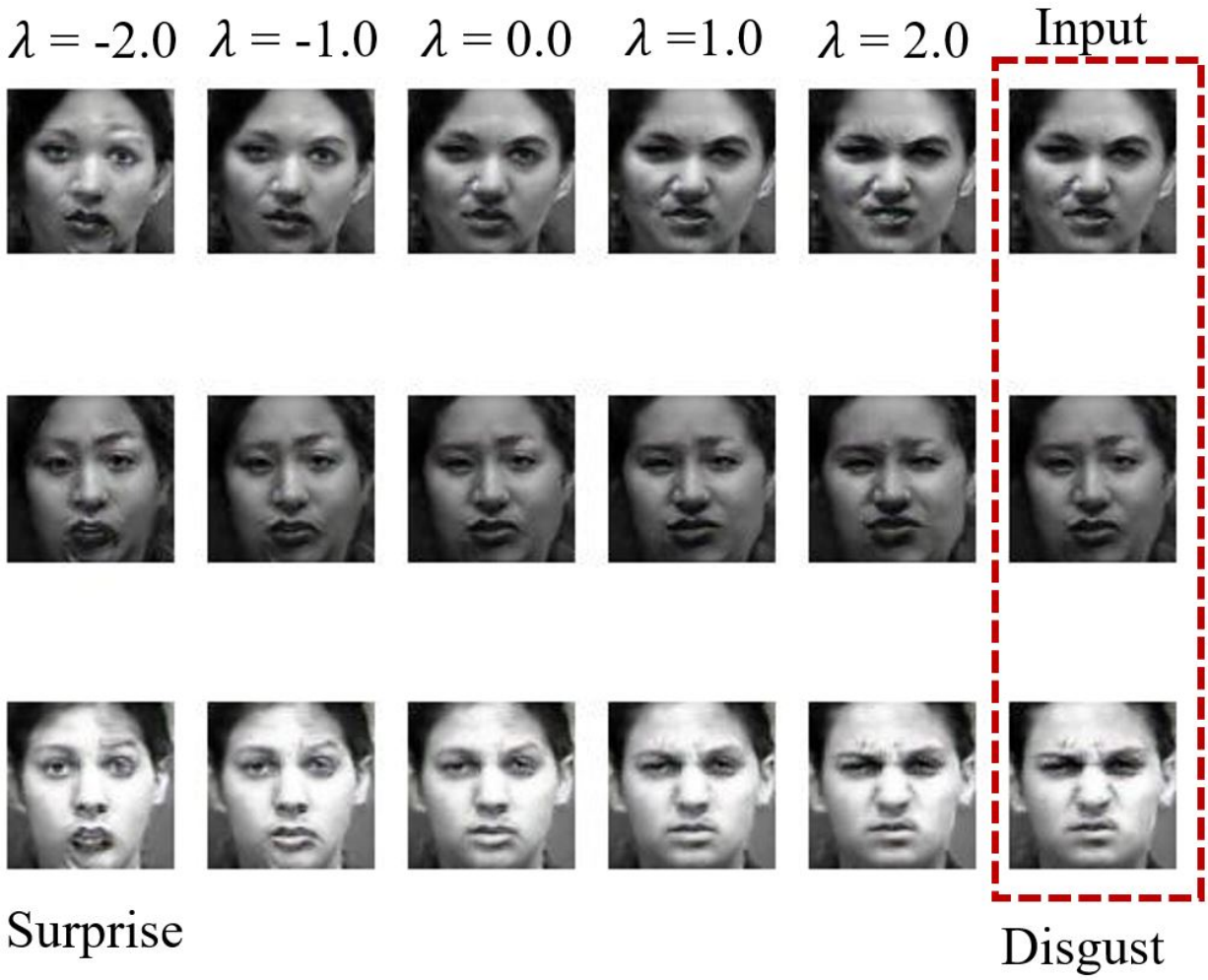


Figure B.38: An example output of facial expressions manipulation from '***disgust***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

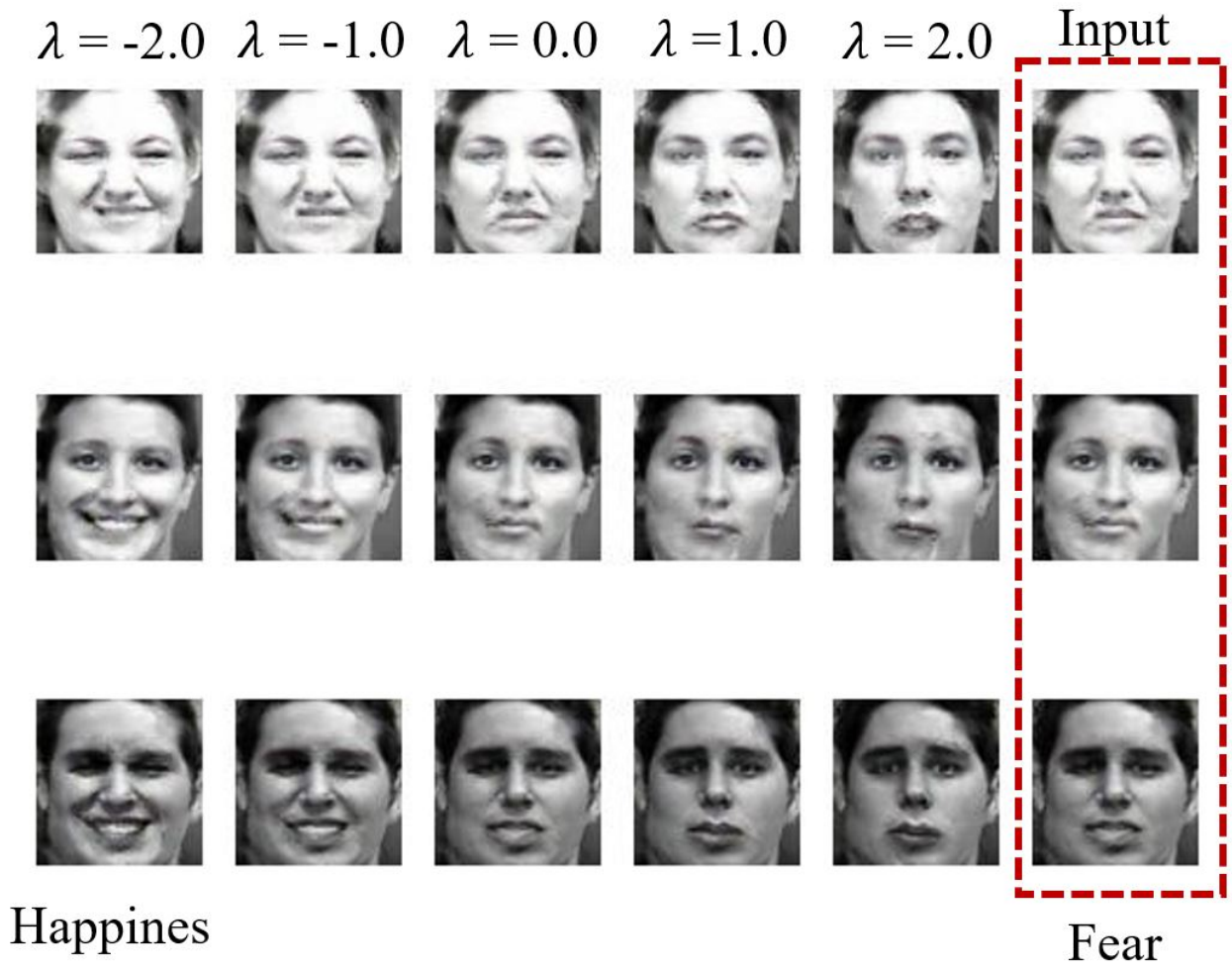


Figure B.39: An example output of facial expressions manipulation from '***fear***' class to '***happiness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

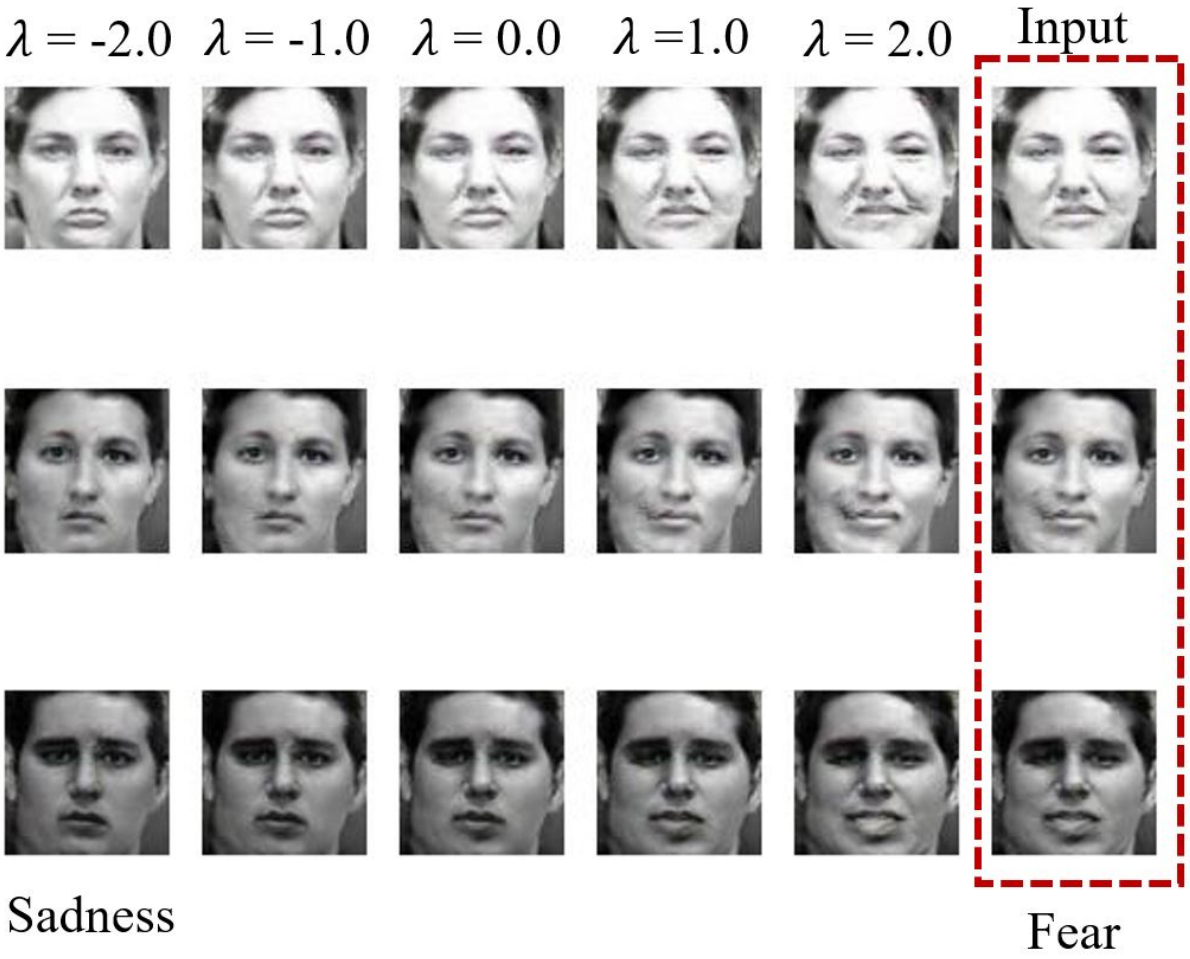


Figure B.40: An example output of facial expressions manipulation from '***fear***' class to '***sadness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

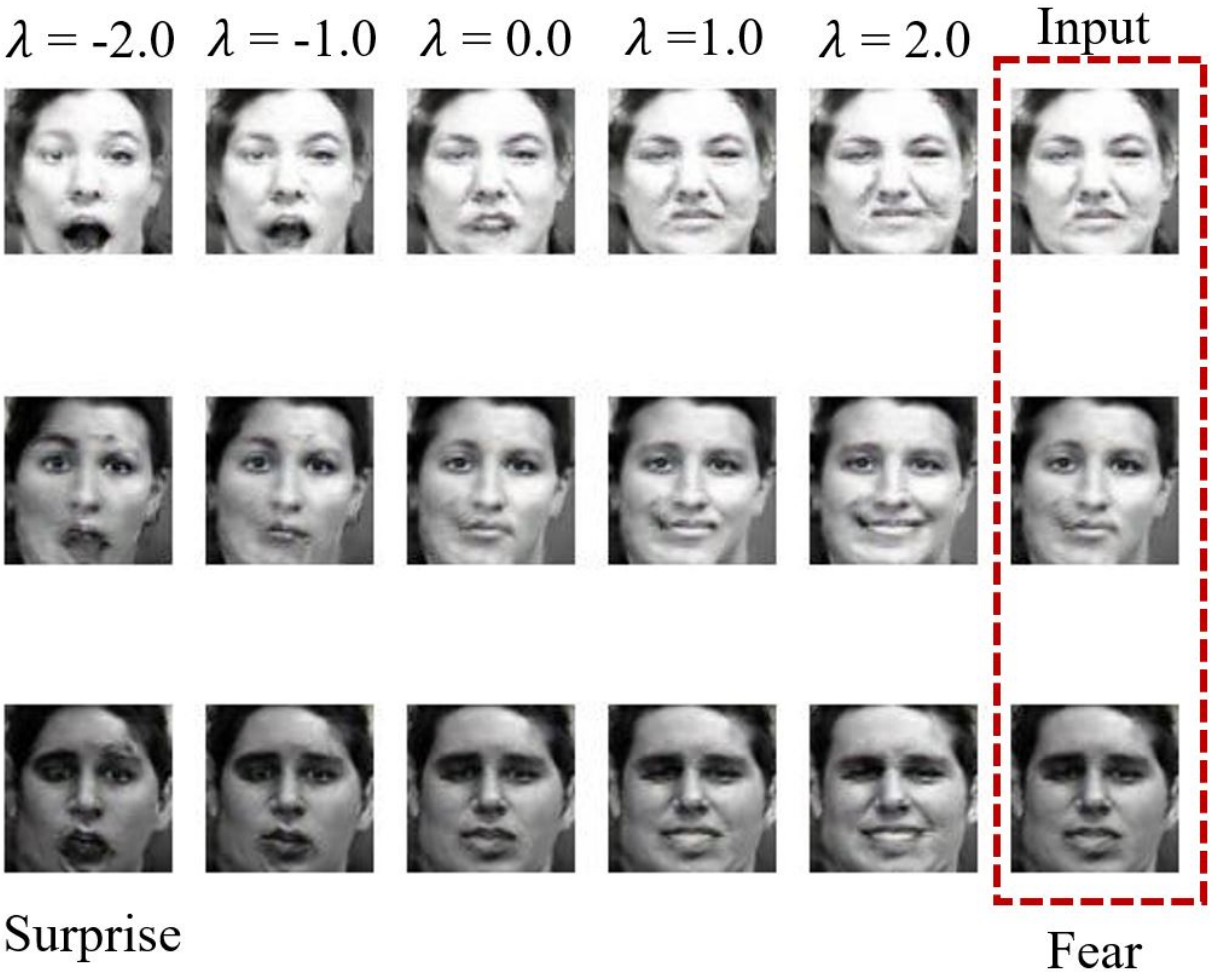


Figure B.41: An example output of facial expressions manipulation from '***fear***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

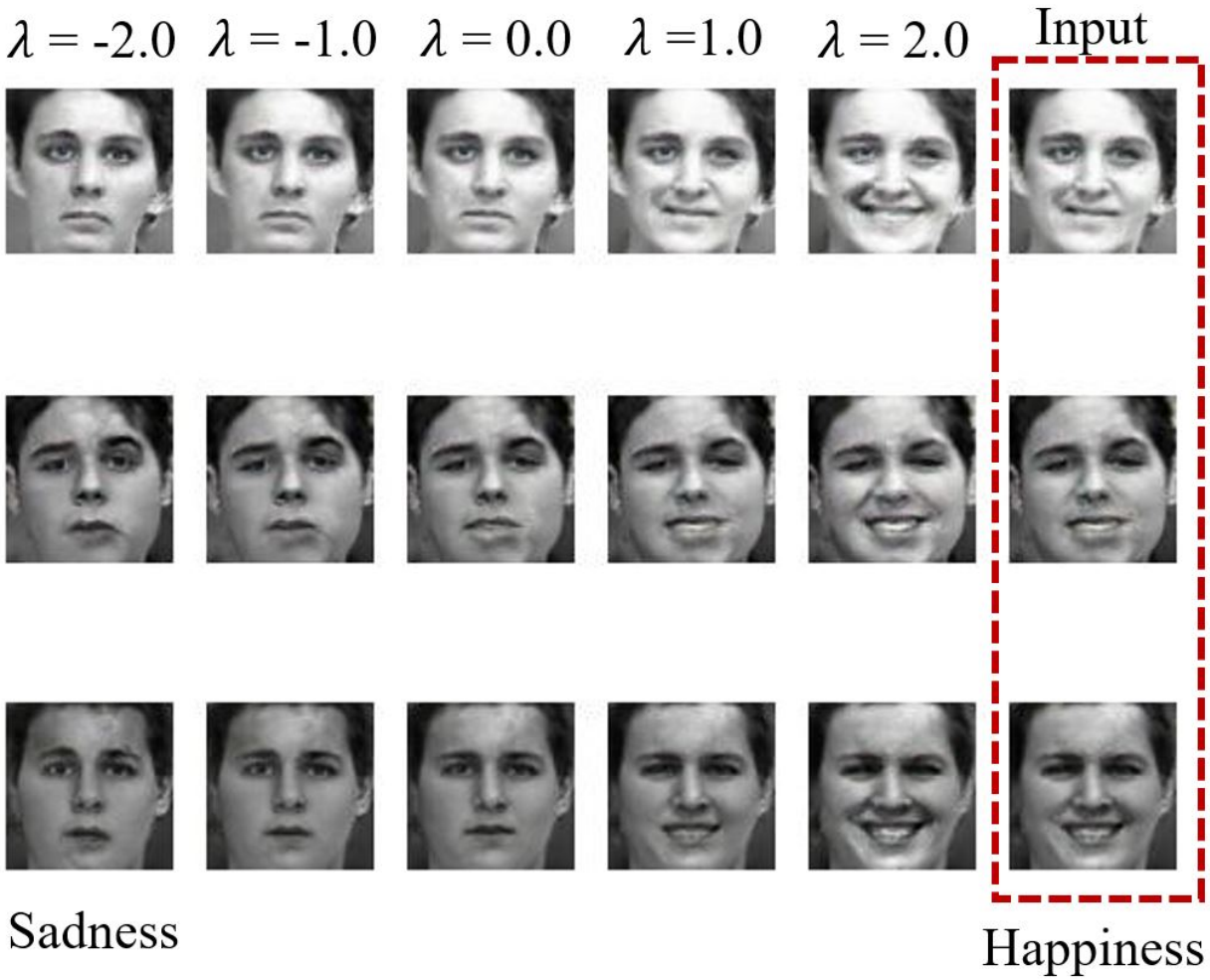


Figure B.42: An example output of facial expressions manipulation from '***happiness***' class to '***sadness***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

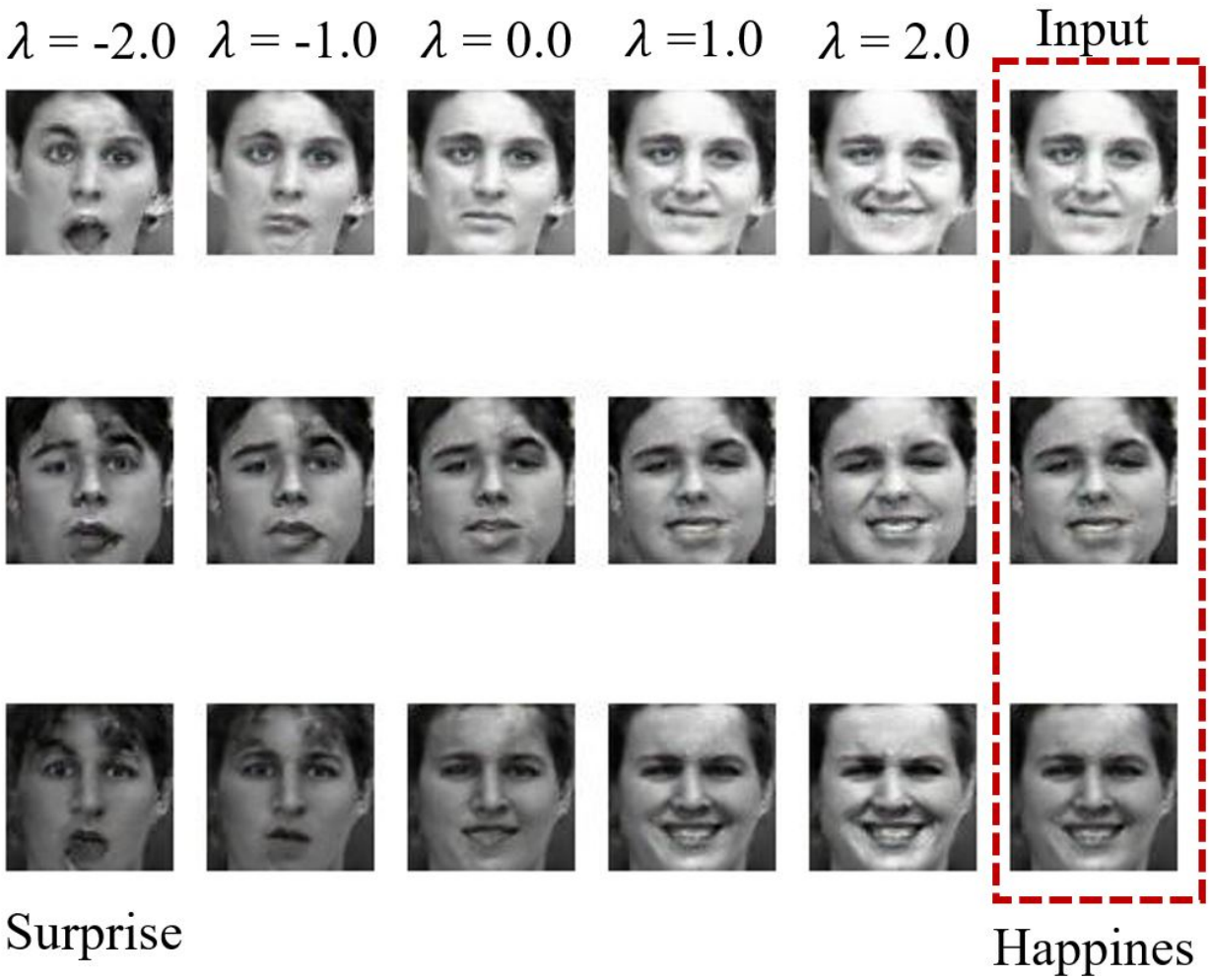


Figure B.43: An example output of facial expressions manipulation from '***happiness***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.

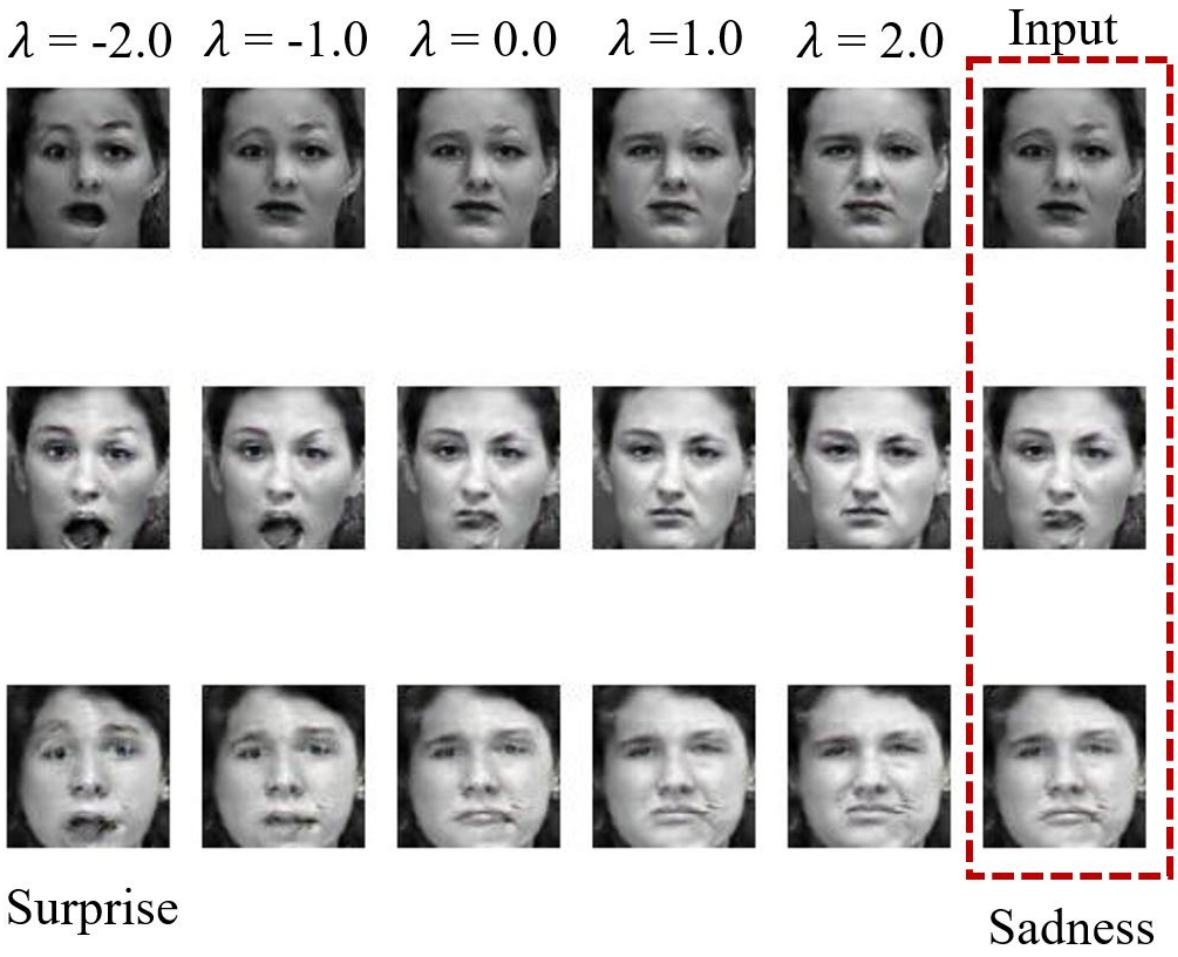


Figure B.44: An example output of facial expressions manipulation from '***sadness***' class to '***surprise***' class. The initial and modified latent are mapped onto image space by trained DCGANs' generator G for illustration purpose. λ can be any real number and is set from -2.0 to 2.0 in this experiment.