VIETNAM NATIONAL UNIVERSITY - HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
**FACULTY OF COMPUTER SCIENCE AND ENGINEERING**


**Assignment for Machine Learning (CO3117)**

# FACIAL EMOTION RECOGNITION

Lecturer: Nguyen Duc Dung

Class: CC01

Student: Le Phong Hao – 2252182

HO CHI MINH CITY, JUNE 2024

# Contents

# 1 Introduction & Motivation

In this report, an emotion recognition model from facial expressions will be developed and be applied to identify emotions from static images or real-time camera. Emotion recognition, refers to identifying expressions such as fear, happiness, and disgust, etc; which is an essential topics which can be applied in various applications, including mental healthcare, human-computer interaction, and security systems due to evolution of Artificial Intelligent. Humans can express their emotions through multiple channels such as speech, linguistic cues, and facial expressions. In intelligent Human-Robot Interaction, understanding emotions can lead to more natural and effective communication, improving user experience and engagement. Emotion analysis is also beneficial in improving mental health, monitoring emotions through AI can provide valuable insights, helping to identify and address issues like depression, anxiety, or stress.

# 2 Related Work

Over the last decade, several solutions have been found and improved in performance day by day to handle the issue of Facial Expression Recognition (FER). The majority of solutions for FER systems focus on six basic emotions types, namely: happy, surprised, fearful, sad, angry, and disgust and consists of three main stages: pre-processing, facial feature extraction, and expression classification.

To process images data, artificial neural networks (ANN) have shown great potential and usually have higher performance on a variety of tasks, such as object recognition, scene classification, and face recognition than other classical machine learning methods. However, classical machine learning methods (k-nearest neighbor, support vector machine, ...) are sometimes combined with ANN to enhance performance.

Wang and Xiao (2013) proposed a new technique of facial expression classification based on neural network ensembles. To extract features from face images they used Principal Component Analysis and Gabor filters. MD-Adaboost is used to combine the results of neural network classifiers. Classifiers are trained using different feature sets to improve the classification rate and stability of the classifier. The experiments demonstrate the positive effect of the neural network ensemble based classifier, and show that the MD-Adaboost based classifier was more efficient and firm than other algorithms.

Hai *et al.* (2015) proposed a model for facial expression classification by combine Artificial Neural Networks and K-nearest neighbor. They used Independent Component Analysis to extract facial features. The input of Artificial Neural Network classifier is all feature vectors processed by ICA. The input of K-NN classifier is the distance ratios of the local region of face. To combine the output of ANN and K-NN classifier the minimum function is used.

Yang *et al.* (2017) leveraged a partial VGG16 network and a shallow CNN to extract two feature vectors from facial grayscale images and LBP facial images, respectably. Then the two feature vectors were fused to fully use complementary facial information.

Tang *et al.* (2018) extracted features by twelve convolutional and pooling layers which were more efficient and provided a great improvement, compared with the 78 dimensions geometric features.

Zheng *et al.* (2018) presented a VGG16 + 1D-CNN model for FER. In their framework, representations of each frame of a video were extracted with VGG16 network followed by four 1D-CNN networks. Then the features were concatenated and were fed to two fully connected (FC) layers to predict facial expressions.

# 3 Experiments

This report presents an approach based on deep neural network (VGGNet in specific) and we will try to find optimal model among architectures and hyperparameters. The dataset used in this report is FER-2013, a dataset widely used for assessing the performance of facial expression recognition systems.

## 3.1 Dataset Description

The Facial Emotion Recognition 2013 (FER-2013) dataset was created by Pierre Luc Carrier and Aaron Courville and was introduced in the ICML 2013 workshop's facial expression recognition challenge. This dataset consists of 35887 images (size of $48 \times 48$ pixels) of 7 different emotions: anger, neutral, disgust, fear, happiness, sadness, and surprise; which is divided into 3 parts: 28709 images are used for training, 3589 images used for public testing and 3589 images used for private testing. The dataset includes 3 columns, which are "pixels", "emotion", "usage"; where:

- pixels: each row contains 2034 integers ranging from 0 to 255, representing the colors of every pixel in the image.
- emotion: each row displays a number corresponding to a specific emotion.
- usage: each row indicates how this image can be utilized (training, public testing, or private testing).


**Figure 1:** *Example images from FER-2013 dataset*

The dataset contains faces greatly vary in age, pose and occlusion conditions. In addition, the average accuracy of human recognition model testing on this dataset is approximately $65 \pm 5\%$.

### 3.2 Dataset preprocessing and Augmentation

During the preprocessing step, each pixel array of every image is transformed into a 2-dimensional array of size $48 \times 48$, with each pixel value normalized by rescaling with a factor of $\frac{1}{255}$. For emotions, the one-hot encoding method is employed to convert the original output into an array of 7 elements, where all elements are set to 0 except for the one at the index corresponding to the original output, which is set to 1.

To improve performance and increase the model's robustness against noise and unbalanced data, significant amount of data augmentation is applied in training dataset. This augmentation includes rescaling the images up to $\pm 20\%$ of its original scale, horizontally and vertically shifting the image by up to $\pm 20\%$ of its size, and rotating it up to $\pm 20$ degrees.

### 3.3 Model Architecture

Network structure in this report is built based on VGG, also known as VGGNet. VGG is a classical convolutional neural network architecture used in large-scale image processing and pattern recognition. VGG was developed to increase the depth of such CNNs to increase the model performance.

Since the size of images used in original VGGNet are $224 \times 224$ and size of images in FER-2013 are $48 \times 48$, convolutional stages and number of filters in each convolutional block also be tested and changed to speed up training process.

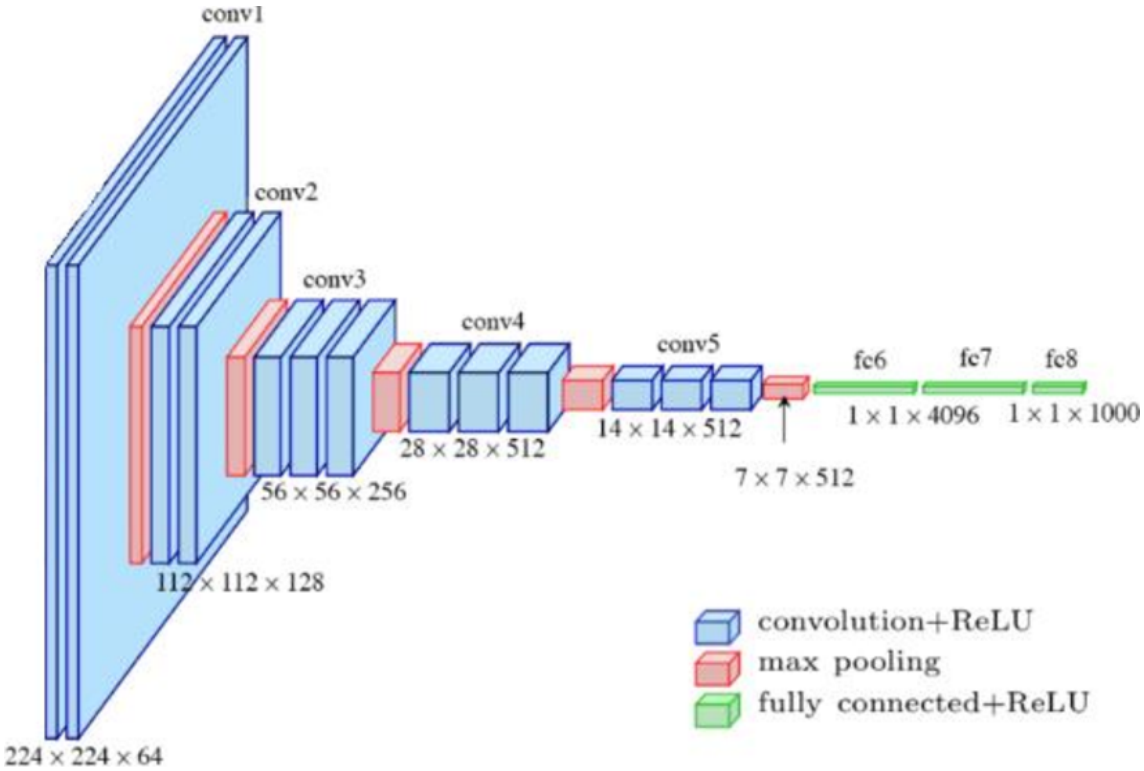



**Figure 2:** *Example of VGGNet architecture*

The network consists of 4 convolutional stages and 3 fully connected layers. Each of convolutional stages contains two convolutional blocks and a max-pooling layer. Each convolution block consists of a convolutional layer, a ReLU activation, and a batch normalization layer. Batch normalization is used to reduce the internal covariance shift, and prevent gradient vanishing or explosion. Batch normalization has been shown to be extremely effective at reducing the number of epochs to train a neural network, reducing the internal covariance shift, and preventing gradient vanishing or explosion. The first two fully connected layers are followed by a ReLU activation. The third fully connected layer is used for classification followed by softmax activation. The predicted label will be the index where its corresponding value is highest in last layers.

The convolutional stages are responsible for feature extraction, dimension reduction, and non-linearity. The fully connected layers are trained to classify the inputs as described by extracted features.

Parameters of our model architecture are:

| Layer | Feature map | Size | Kernel size | Stride | Activation |
|---|---|---|---|---|---|
| Image (Input layer) | 1 | 48 x 48 x 1 | - | - | - |
| Conv2D + BatchNorm | 64 | 48 x 48 x 64 | 3 x 3 | 1 | ReLU |
| Conv2D + BatchNorm | 64 | 48 x 48 x 64 | 3 x 3 | 1 | ReLU |
| MaxPooling | 64 | 24 x 24 x 64 | 2 x 2 | 2 | - |
| Conv2D + BatchNorm | 64 | 24 x 24 x 64 | 3 x 3 | 1 | ReLU |
| Conv2D + BatchNorm | 64 | 24 x 24 x 64 | 3 x 3 | 1 | ReLU |
| MaxPooling | 64 | 12 x 12 x 64 | 2 x 2 | 2 | - |
| Conv2D + BatchNorm | 128 | 12 x 12 x 128 | 3 x 3 | 1 | ReLU |
| Conv2D + BatchNorm | 128 | 12 x 12 x 128 | 3 x 3 | 1 | ReLU |
| MaxPooling | 128 | 6 x 6 x 128 | 2 x 2 | 2 | - |
| Conv2D + BatchNorm | 128 | 6 x 6 x 128 | 3 x 3 | 1 | ReLU |
| Conv2D + BatchNorm | 128 | 6 x 6 x 128 | 3 x 3 | 1 | ReLU |
| MaxPooling | 128 | 3 x 3 x 128 | 2 x 2 | 2 | - |
| Flatten | - | 1152 | - | - | - |
| Fully connected | - | 1024 | - | - | ReLU |
| Fully connected | - | 1024 | - | - | ReLU |
| Fully connected (Output layer) | - | 7 | - | - | Softmax |

### 3.4 Training process

The network will be trained for 70 epochs, aiming to minimize the categorical cross-entropy loss between the final layer and the one-hot encoded true values. The batch size is set at 64, and requiring 449 iterations to finish one epoch. Early stopping is implemented to monitor the validation loss, allowing the training process to halt if there are no improvements for 10 consecutive epochs. A fixed weight decay of 0.0001 is applied. While the optimizer and learning rate will vary in each experiment, all other parameters will remain unchanged to determine the best optimal solutions.

## 3.5 Experiment result

We try to find a model which give us the best performance. At first, we use Adaptive Moment Estimation optimizer, or known as Adam optimizer, to find the optimal learning rate (LR) which reach optimal solution faster and have higher performance than training with other LR. Then, using the optimal LR, we try to determine the optimal optimizers among Adam, Adagrad, SGD and RMSprop.

### 3.5.1 Optimizer

At first, we will try find the best optimizer among 3 algorithms with adaptive learning rates, which are Adam, AdaGrad and RMSProp, when training our model. We will fixed the optimal learning rate at 0.001.

With AdaGrad algorithms, after 70 epochs, the model does not reach the converge value and still improve, which means we need more epochs and training time to have the desired performance. The performance when testing on private test is 53.55%.

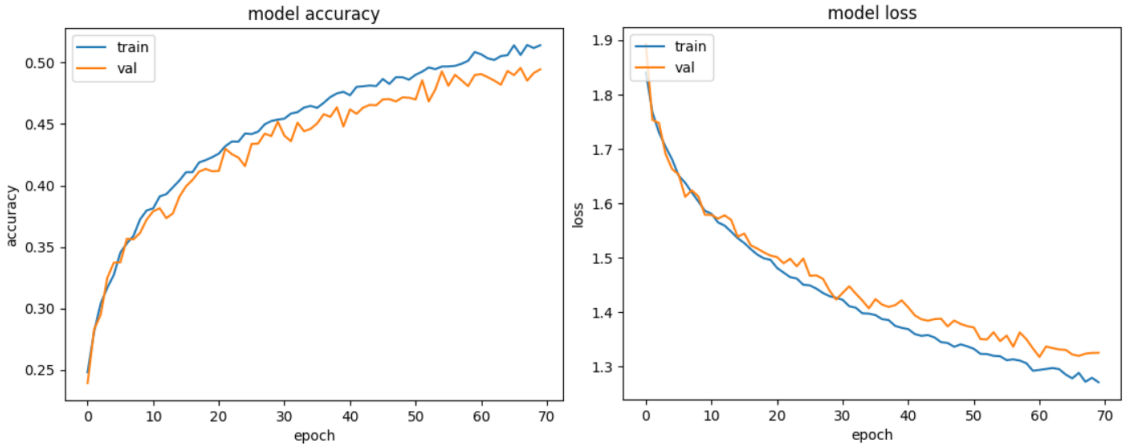



**Figure 3:** *Accuracy and Loss when training with AdaGrad*

Adam and RMSProp give us similar performance (66.65% and 66.23%) and number of epoch required (45 and 52 epochs)

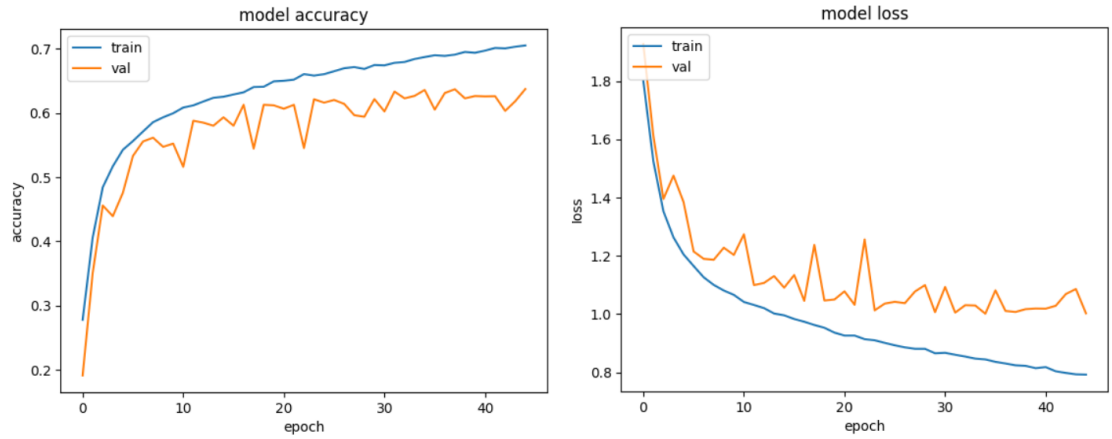



**Figure 4:** *Accuracy and Loss when training with Adam*

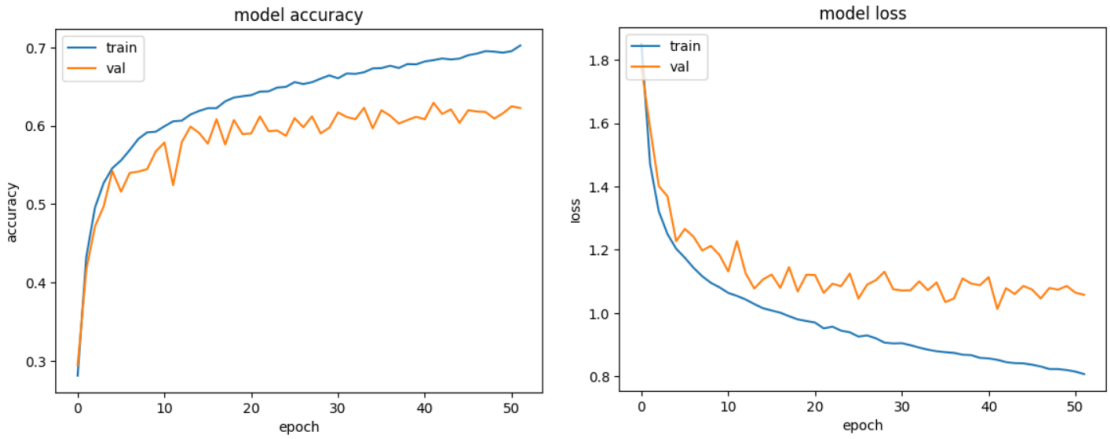



**Figure 5:** *Accuracy and Loss when training with RMSProp*

### 3.5.2 Learning rate

We fix the model architecture and use the Adam optimizer to find the optimal global learning rate. The model will be trained with different LR among $0.01, 0.005$ and $0.001$.

We can see that, 0.01 LR gives us a poor performance with $\approx 25\%$ accuracy, since the model halt very soon (at 12nd epoch) when the model do not have any improvement from 2nd epoch in validation set and trivial improvement in training set.

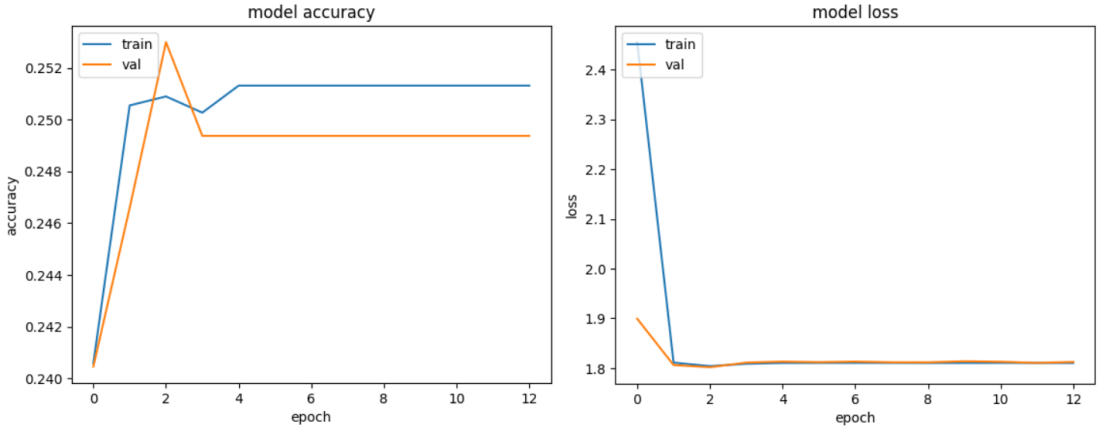



**Figure 6:** *Accuracy and Loss when LR is* 0.01

When testing with private dataset with LR between 0.001 and 0.005, both of them give us similar performances (66.67% and 65.87%). However, when training with 0.001 LR, the model converges and halt faster, which requires 45 epochs, than the other one, which require 60 epochs.

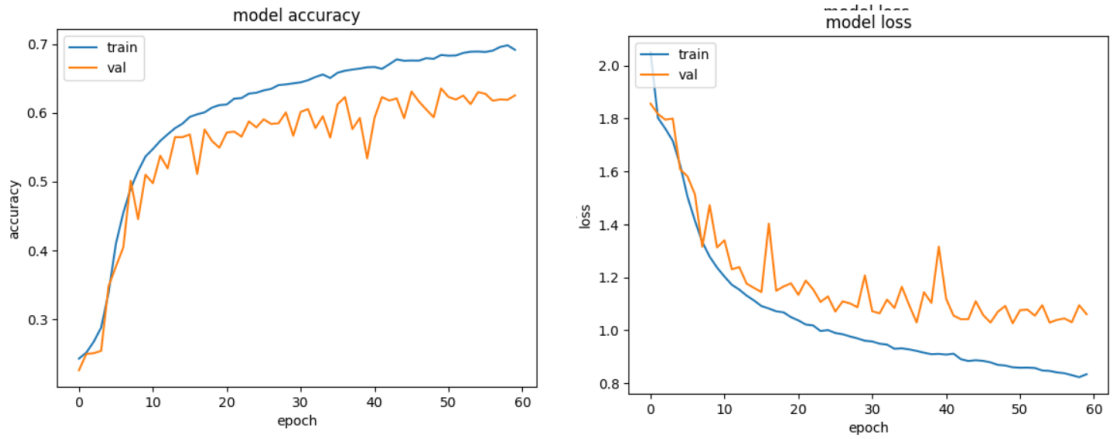
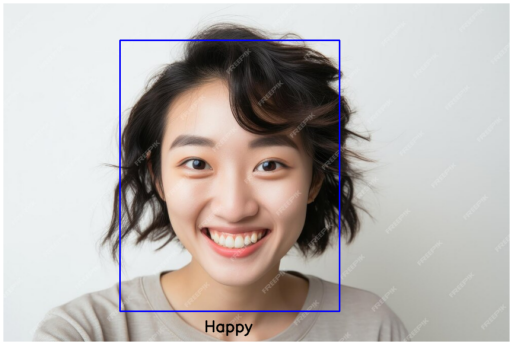


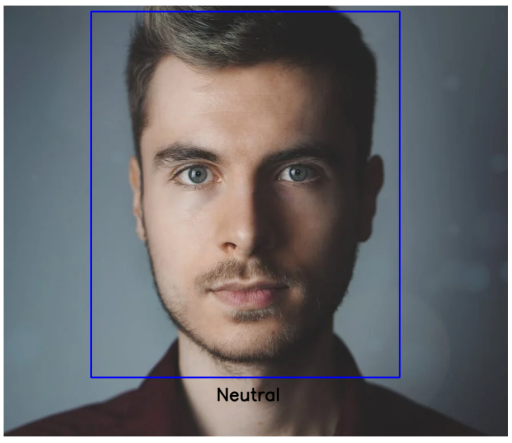

**Figure 7:** *Accuracy and Loss when LR is* 0.005

### 3.6 Apply model on static images

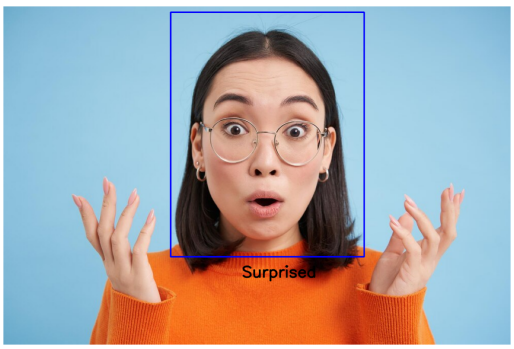
From previous results, we choose model trained with Adam optimizer and global learning rate is 0.001 as our main model. We used the pre-trained Haar Cascade classifier to detect faces information (width, height, origin position) within a raw static image. From these information, new images created from every faces are rescaled and converted to black and white color. Then, these new images are fed into our model to classify final results.



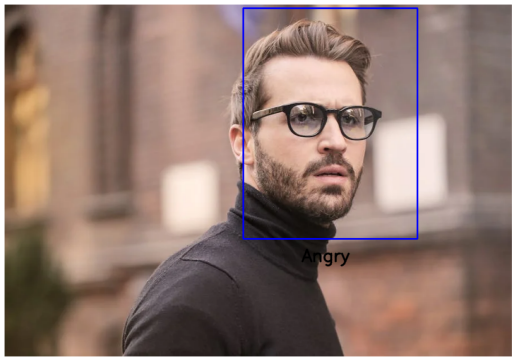

**(a)** *Happy face*




**(b)** *Neutral face*




**(c)** *Surprise face*




**(d)** *Angry face*

**Figure 8:** *Example images with corresponding emotions*

Since this model performs with $\approx 66\%$ accuracy, it can misclassify between similar emotions (e.g., surprised/fearful or neutral/sad). Furthermore, this model, also, can be integrated into real-time camera in the same way. However, camera parameters need to be managed and implementing loops for continuous processing to update the emotion in the most recent frame.

# 4 Comparison with Classical Machine Learning methods

## 4.1 Features extraction process

When working with facial emotion recognition, Neural Networks, particularly Convolutional Neural Networks, excel in image-based tasks by automatically learning features from raw data, such as images. These models effectively learn complex patterns within the inputs. Additionally, feature extraction and classification are integrated into a single model, allowing simultaneous training.

In contrast, classical machine learning methods like Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Conditional Random Fields (CRFs) require manual feature extraction or the use of other techniques to identify patterns within the input when neural networks are not being used. Efficiently locating these patterns can lead to good model performance. However, feature extraction and classification are separate steps, which means we have to handle each process ourselves.

## 4.2 Complexity and Performance

Deep neural networks require a large number of parameters, which result in long training times and increased computational resource demands (GPUs/CPUs). However, with these complex parameters and activation functions provided inside model, neural networks can perform better on large and non-linear datasets than other classical methods.

In contrast, classical machine learning methods require fewer parameters, reducing training time and the resources needed for training. However, to achieve good performance, raw data need to be well pre-processed and feature extraction is necessary before feeding it into the model. These methods work well with smaller datasets.

## 4.3 Flexibility and Adaptability

Deep neural networks can be adapted to various tasks beyond classification, such as face detection and segmentation, by modifying the network architecture. Pre-trained neural networks can also be fine-tuned and used on new tasks, inheriting knowledge from related tasks. In contrast, classical methods are typically constructed to solve problems related to classification and regression tasks. To adapt to other tasks, raw data need to be modified and features extracted.

# 5 Conclusion

In conclusion, this report has shown the process of building and choosing parameters for a deep learning model through experiments to recognize emotions from facial expressions. Through

data pre-processing, model selection, training, and testing steps, we achieved an accuracy of approximately 65%. While this performance is acceptable and can be applied in real-life applications, it also highlights the challenges of classifying similar emotions. In addition, we also discuss other approaches being used to solve these problems and compare classic machine learning methods and deep learning methods, particularly neural networks.

**Project source code:** https://github.com/lphonghao/machine_learning_FER

**FER – 2013 dataset:** `icml_face_data.csv` - https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge

## References

[1] M. A. Rahman and M. H. Seddiqui, "Comparison of classical machine learning approaches on bangla textual emotion analysis," 7 2019.

[2] P. Giannopoulos, I. Perikos, and I. Hatzilygeroudis, *Deep Learning Approaches for Facial Emotion Recognition: A Case Study on FER-2013*, pp. 1–16. 2018.

[3] H. Chouhayebi, J. Riffi, M. A. Mahraz, A. Yahyaouy, and H. Tairi, "Facial expression recognition using machine learning," pp. 1–6, IEEE, 10 2021.

[4] A. I. Siam, N. F. Soliman, A. D. Algarni, F. E. A. El-Samie, and A. Sedik, "Deploying machine learning techniques for human emotion detection," *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–16, 2 2022.

[5] Y. Khaireddin and Z. Chen, "Facial emotion recognition: State of the art performance on fer2013," 5 2021.

[6] G. Boesch, "Very deep convolutional networks (vgg) essential guide."

[7] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2 2015.

[8] Z. Zheng, C. Cao, X. Chen, and G. Xu, "Multimodal emotion recognition for one-minute-gradual emotion challenge," 5 2018.

[9] Y. Tang, X. M. Zhang, and H. Wang, "Geometric-convolutional feature fusion based on learning propagation for facial expression recognition," *IEEE Access*, vol. 6, pp. 42532–42540, 2018.

[10] B. Yang, J. Cao, R. Ni, and Y. Zhang, "Facial expression recognition using weighted mixture deep neural network based on double-channel facial images," *IEEE Access*, vol. 6, pp. 4630–4640, 2018.

[11] A. Polyak and L. Wolf, "Channel-level acceleration of deep face representations," *IEEE Access*, vol. 3, pp. 2163–2175, 2015.

[12] X. Zhu, Z. Lei, J. Yan, D. Yi, and S. Z. Li, "High-fidelity pose and expression normalization for face recognition in the wild," pp. 787–796, IEEE, 6 2015.

[13] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," 3 2015.

[14] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," vol. 27, Curran Associates, Inc., 2014.

[15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," 9 2014.

[16] J. Deng, G. Pang, Z. Zhang, Z. Pang, H. Yang, and G. Yang, "cgan based facial expression recognition for human-robot interaction," *IEEE Access*, vol. 7, pp. 9848–9859, 2019.