The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Facial Expressions Identification System by Shallow-Convolutional Neural Network

Contents

1. Introduction

The advancements in computer vision, computing resources and machine learning make our world richer, efficient and innovatory. Thanks to state of the art technologies, no more humans will be needed for working in factory production line and they will not drive the vehicles themselves in the near future. One of great achievements in machine learning area is facial expression recognition (FER). Robot, computer and many other devices can recognize the person's feeling from his face and use this information to boost or relax his emotion. For example, the secretary robot can recognize the user's sad feeling by their camera and turn the proper music to assuage him. With the ability of FER, computer and other devices are considered as friends and counsellors which can understand person's situation and sentiment.

For practical applications, there are three main problems to be solved. First, human possesses diverse emotions. In [1], human possesses at least 27 emotions which are not always isolated (uncorrelated). From this aspect, 27 emotions are sometimes very similar and hard to be distinguished by machines. Thus, most researches about facial emotions are done on 7 basic facial expressions which are basically isolated and easy to be distinguished. The basic 7 expressions are angry, disgust, fear, happy, neutral, sad and surprise. Second problem is the shape of face for same emotion can be different from each person. This problem is caused because of different nation, sex, culture and many other factors, which make hard to build the universal system for FER. Lastly, the computational time is important for real world application. Same person can give diverse shapes of face for same expression, thus machine should do online-training. Without real time processing, we cannot adapt the system of FER in our devices.

In this project, three problems mentioned on above are considered as follows. 7 basic facial expressions are only used as labels because they are less related with each other and data that can be acquired consists of these feelings. For tolerating the variance of facial shape, Japanese woman facial emotion data is used to fix the variable of sex, culture and nation. To get real time computational cost, three types of data are considered. One of them is the facial image without background information and others are the cropped images of former one. More detailed is covered in 3. Methodology.

2. Background

Conventionally, there are three methods used to detect the facial expressions: geometric-based method, appearance-based method and hybrid of geometric and appearance method. Geometric-based method transforms face image into geometric primitives and uses the relationship between those components to construct feature vector for training. It is useful to hold the accuracy invariant for illumination and temporal differences in video data. Late 19th century, one of groups [2] used 35 geometrical features from nose width, nose length, mouth position and chin shape with Bayes classifier to recognize the expressions. This group achieved about 90% accuracy. In [3], they applied Active Shape Model which is one of statistical models for geometric-based approach to fit landmarks and build dynamic features to make real-time facial expression recognition. They used these features with SVM classifier and got 86% of accuracy with 10 fold cross validation. Appearance-based method considers the location and shape of facial features that contribute the variations of appearance among people. Also, it considers the facial textures like skin tone, facial hair, scars and wrinkles. Usually, appearance-based method extracts features from the global face region or specific face regions. From [4], they applied a local binary pattern histogram of global face region and classified the facial emotions by principal component analysis method. They classified 6 basic emotions and achieved 97% accuracy. Compared to features from global face region, features from specific face regions possess different importance because each face region has different contribution to facial expression. For example, eyes and mouth contain much information than nose and cheek. The group [5] extracted the region specific appearance features by dividing the overall face region into domain-specific local regions. Importance of local regions is decided by using incremental search approach. They also got high accuracy which is 97%. For hybrid method, the Active Appearance Model is a well-known method with good performance [6].

Commonly, all conventional approaches require the programmer to determine the features and classifiers. These approaches demand relatively low computational resources and memory, and also give high accuracy. In these aspects, they are suitable for real time processing, but it is actually inappropriate because programmer cannot always do hand-crafted feature extraction and optimize the classifier in real time. For this reason, deep learning model is considered to solve this situation. Jung et al. [7] used Convolutional Neural Network (CNN) to automatically extract the useful facial features. They build two CNN networks: One network is to extract temporal appearance features from image sequences and the other network extracts temporal geometry features from facial landmarks. The group [8] used gradient direction information of depth data with CNN and achieved about 88% accuracy. The other group [9] used the downsampled facial image without background information and got 84% accuracy.

From literature review, shallow depth CNN is applied in this project to achieve less computational time and the ability of automated feature extraction. Also, three types of facial image without background (called cropped image in this report) are considered to see which parts of facial region are important to determine the facial expressions and search the room for improvements in training time.

3. Methodology

3.1 Data

In this project, Japanese Female Facial Expression (JAFFE) data [10] is used for training and testing the network. The data has 213 images of 7 facial expressions which are angry, disgust, fear, happy, sad, surprise (6 basic facial expressions) and neutral posed by 10 Japanese female models. In detail, there are about 3 images in each one of 7 expressions from each subject. All images are 256x256 pixel size with 8 bit precision in gray scale values. The photos were taken at the Psychology Department in Kyushu University and they were captured in controlled environment.

Table 1. Distributions of expressions in JAFFE data

	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
Number of data	30	29	32	31	31	30	30

From table 1, the distribution of 7 expressions are almost uniform even if the size of data is not large. Thus, data augmentation is not needed to control the class imbalance problem. In addition, there was no overfitting when I check the cost curve of training and validation set, thus data augmentation is not required in this project.

3.2 Facial Detector and Landmarking

Facial detector and landmarking are applied to localize the salient regions of the face which are important for interpreting the emotions. The examples of salient regions are eyes, eyebrows, nose, mouth and jawline. There are two steps to do localization: First step is to detect the face in the image (facial detector) and second step is to detect the key facial structures from ROI of face (facial landmarking).

For facial detector, pretrained object detector through Histogram of Gradient (HoG) and linear support vector machines (SVM) in dlib module is used. This detector was trained as followed: 1) Sample 'positive samples' from training data that has object (face) and extract HoG features from these samples. 2) Sample 'negative samples' from training data that has no object and extract HoG features from them. 3) Train Linear SVM on HoG features of positive and negative samples. 4) Apply sliding window technique to slide the window over the positive and negative samples. If sliding window incorrectly classifies an object (false positive), collect the feature vectors with the probability of classification. 5) Sort the false positive feature vectors by probability of classification. 6) Train linear SVM again on these samples to get better classification result.

For facial landmarking, shape predictor function [11] in dlib module is applied. Briefly, manually marked coordinates of salient regions of face and probability on distance between these coordinates are used as train data for constructing this function. Train data goes through the ensemble of regression trees to estimate the facial landmarks. Feature extraction is not needed to build this function and it was trained on iBUG 300-W dataset [12]. The function estimates the location of 68 coordinate points that map to salient regions of the face.

Original image **Cropped image**

Figure 1. Original image vs Cropped image. Cropped image was built using facial landmark information

Using facial detector and facial landmarking, salient regions of faces are detected well in all images in JAFFE. By using this information, images are cropped to eliminate the background information. Figure 1 shows the comparison between the original image and cropped image after facial landmarking.

3.3 Convolutional Neural Network (CNN)

CNN is a deep artificial neural network that is used primarily to classify images, cluster them by similarity, and perform object recognition within scenes. This neural network can identify faces, individuals, street signs, tumors, platypuses and many other aspects of visual data. Usually, CNNs are composed of an initial layer of convolutional filters (a set of weights which are slided over the input), followed by non-linearity (activation function), sub-sampling (pooling), and regularization (dropout). The detail of these concepts will be covered at below.

Convolution in convolution layer uses same concept as signal processing and image processing area. The filter goes through the input data while conducting convolution operation and passing the result to the next layer. The filter consists of weights which are going to be trained with input data and every filter shares same weights. At convolution, the size of output can be controlled by changing the number of filters, the size of filter, and stride. These are called hyperparameters and it should be tuned carefully. Unfortunately, there is no criteria to choose these hyperparameters because they heavily depend on data sample. Thus, I chose the best one from exhaustive searching.

Activation function introduces nonlinearity to the system that basically has just been computing linear operations during the convolution. Without using this function, we cannot guarantee to classify a dataset accurately if its feature cannot be linearly classified. For this project, a Rectified Linear Units (RELU) is used to overcome the vanishing gradient problem, that is the issue where the lower layers of the network train very slowly because the gradient decreases exponentially through the layers.

Pooling is referred to as a downsampling layer since it decreases the size of input to get the smaller output size without losing important structure information of data. Similar with convolution layer, filter applies to the input volume and outputs the maximum number in every subregion that the filter convolves around. This is called maxpooling. There are two main purposes. The first is that the number of parameters (weights) can be reduced, thus computational time can be decreased. Also, it can control overfitting.

Dropout layers is used for controlling the overfitting. It is implemented by dropping out a random set of parameters (weights) in that layer by setting them to zero. This forces the network not to fit too much on training data and it will yield better generalization. The important fact is that this layer only activated during training and not during validating and testing.

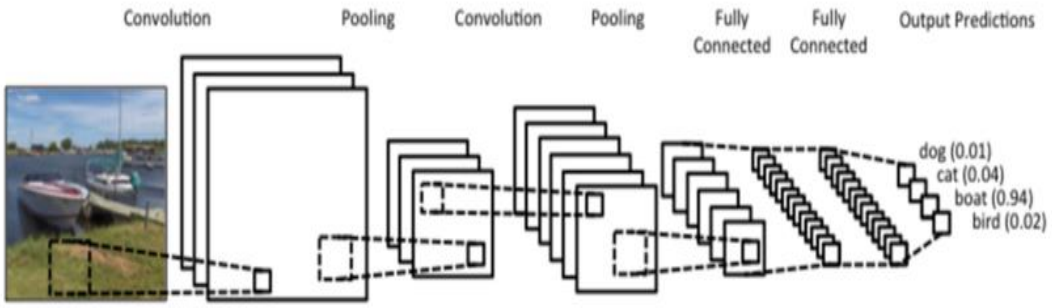


Figure 2. Example of Convolutional Neural Network [13]

3.4 Confusion Matrix

Confusion matrix is a table that is used to describe the performance of a classification model on train and test data. Each row of the matrix represents the number of samples in a predicted class while each column means the number of samples in an actual class.

Table 2. Confusion matrix

	True Data	**False Data**
Predicted as True	True Positive (TP)	False Positive (FP)
Predicted as False	False Negative (FN)	True Negative (TN)

In this project, there are four measurements used to evaluate the performance of models: accuracy, precision, false acceptance rate (FAR), false rejection rate (FRR). All these measurements come from confusion matrix (table 2). Accuracy shows how many samples are predicted correctly from total dataset. Precision means how many samples that we predicted as true is actually true data. FAR measures the ratio of the number of false acceptance and false data. Low FAR ensures the high security system. FRR shows the ratio of the number of false rejection and true data. Low FRR means high success rate to match the true biometric input with a template. At bottom, all these measurements are introduced by mathematical representation.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (1.1)$$

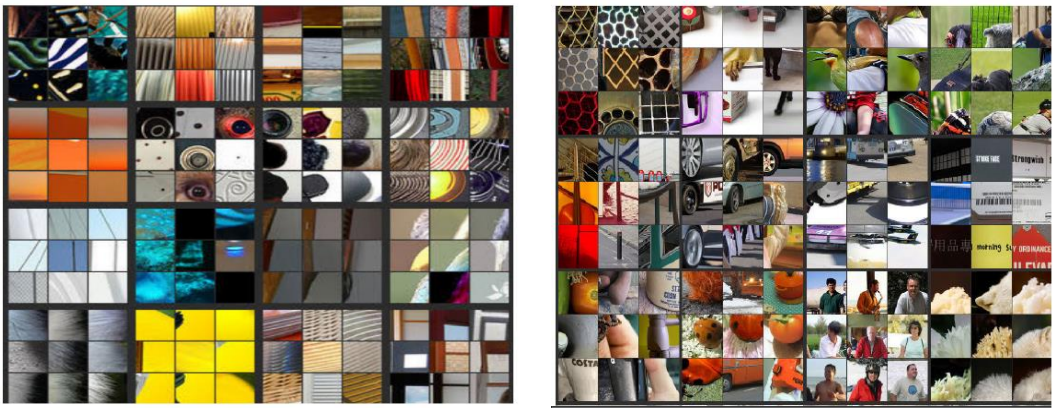
$$Precision = \frac{TP}{TP+FP} \quad (1.2)$$

$$FAR = \frac{FP}{FP+TN} \quad (1.3)$$

$$FRR = \frac{FN}{TP+FN} \quad (1.4)$$

3.5 Feature Visualization

Feature visualization is useful to know how the network works and understands train data. It can provide the intuition about how to interpret the system for better reliability and stabilization. Briefly, feature visualization in CNN is to observe a specific trained feature (also can be layers) to understand what network is really looking for. CNN gradually builds up abstractions: first it detects edges and orientations, then it uses this information to detect textures, the textures to detect patterns, and the patterns to detect parts of objects.



Features from second layer **Features from third layer**

Figure 3. Building up abstractions in CNN [14]. We can see deeper layer has more interpretability and complexity than shallow one

The focus of this project is observing the feature visualization in filters and layers to understand how the network accepts and perceives data. Due to small size of input, it is challenging to get a good shape in filter-level but it is still valuable to see the stacked abstractions. Compared to filter-level, feature visualization in layer-level gives good visualization to understand the interpretation of system. More detail will be covered in 5. Results.

4. Algorithms

4.1 Programming Environment

All the programming is done on python (Jupyter notebook). The main modules used in this project are numpy, tensorflow and dlib module (open-cv). Nvidia Geforce 940MX is used for training and testing the network. This GPU is not high quality, thus execution time has room for improvement.

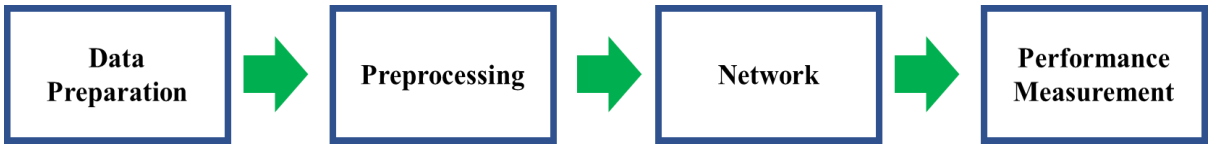


Figure 4. The structure of algorithm

Figure 4 introduces the structure of algorithm used in this project. At below, I explains the details step by step.

4.2 Data Preparation (Three types of data)

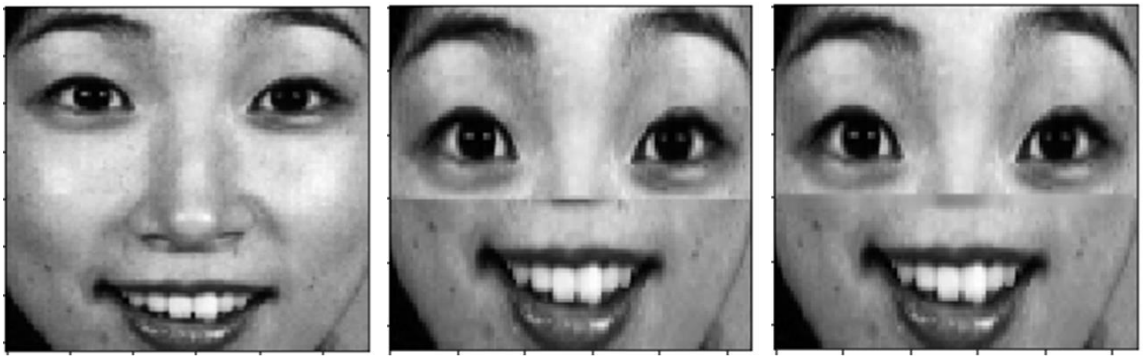


Figure 5. Three types of data. First type is the cropped image after facial landmarking. Second type is made by additional cropping (nose) in first type image. Third type is built by blurring the connection part in second type

There are three types of data considered in this project. First type (figure 5) is the cropped image from eyebrow to mouth. Cropped region is determined from the locations of facial landmarking. Second type is the cropped image from eyebrow to mouth, but except for nose part. To build this data, additional cropping in nose location which can be achieved from facial landmarking is done on first type image. Obviously, the size of the second data type is smaller than first one, thus this data requires less training time. However, the facial expression is mainly determined by eyebrow, eye and mouth, meaning the accuracy of FER is similar between first and second one. When we see second type in figure 5, there is an abnormal connection around middle part of image. CNN can interpret it as edge, thus blurring that part is useful to eliminate the potential degradation. Third type image is made for this reason and it shows similar or better accuracy than second type but has same computational cost. All images should be resized to get same size for each type. After resizing, first type has 144x144 size, while second and third type possess 108x108 size.

4.3 Preprocessing

Each data is normalized by its mean and standard deviation. Normalization is useful that differences between pixel intensities can be reduced, while maintaining the structure information. After normalization, data is randomly shuffled with different seeds. The seed value was given because the order of data should be the same for three types of data to compare the performance. 80% of random shuffled data is used for training, while the other is applied for testing. Since 10-fold cross validation is implemented in this project, there is no validation set. Cross validation is a resampling process used to evaluate the models on a limited data. In detail, data is divided into 10 subsets for 10-fold cross validation. For each fold, one of the 10 subsets is used as the validation set and other subsets are applied for training a model. This process is iterated for 10 times by choosing different subsets and then, one trial of cross-validation is done. The error estimation is averaged over all 10 trials to get total effectiveness of model. Cross validation can reduce bias by using most of data for fitting and reduce variance by also using most of data for validating.

4.4 Network

In this project, 3 layers of CNN are used. Each layer of CNN has convolution, activation, maxpooling and dropout layer. The size of filters applied in CNN is 11x11x1 (first layer), 13x13x32 (second layer) and 15x15x64 (third layer). The number of filters in CNN is 32 (first layer) and 64 (second and third layer). All maxpooling layers have filters with size 2x2 and stride 2. This makes the half of input size with holding structure information. RELU is used for activation function and the amount of dropout is 40%. After CNN layer, one fully connected layer is applied to get classification results for each facial expression. Softmax function is used for converting classification results into probabilities. Then, cross entropy loss is considered to compute loss by comparing between predictions and one-hot encoded true targets, and L2 regularization is applied for preventing overfitting. Regularization constant is fixed as 0.1. All losses are put in ADAM optimizer to fit the network on train data. Learning rate is 0.0001 and epoch is 60 for all cases. All hyperparameters mentioned on this part are resulted from exhaustive searching.

4.5 Performance Measurement

The stabilization and reliability of network is evaluated with various methods. The curve of ‘Epoch vs Cost for validation set in cross validation’ is used to see whether there is overfitting or not. Confusion matrix is applied to calculate the reliability and exactness of network in various aspects. Computational time is measured to see which type of input data has better computational complexity. Finally, feature visualizations are applied in filter-level and layer-level to observe what the network considers and focuses on.

5. Results

The result of different types of data (figure 5) are covered in this part. In this project, 5 different random seeds for each type of data are used, but only few of them are chosen to represent the result. Then, feature visualizations of system in filter-level and layer-level are introduced.

5.1 First Type Data

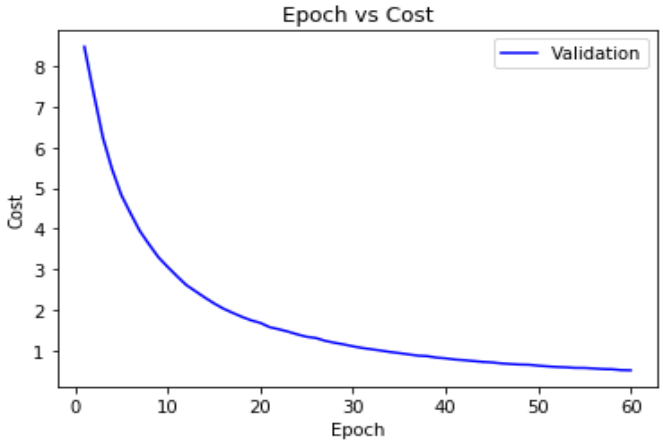


Figure 6. Epoch vs Cost curve for validation set in cross validation (first type image)

Figure 6 shows the cost curve of validation set in cross validation, depending on the number of epoch. According to 5 different random seeds, there are 5 different curves but only one of them is covered here because they are almost similar. From figure 6, the validation cost curve gives clear decreasing shape without any increasement. Thus, we can know there is no overfitting.

Table 3. Confusion matrix of test set in first type image. Results from two of 5 random seeds are introduced

Seed 777		Data						
		Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Predict	Angry	5	0	0	0	0	0	0
	Disgust	1	3	0	0	0	0	0
	Fear	0	0	8	0	0	0	0
	Happy	0	0	0	1	0	0	0
	Neutral	0	0	0	0	5	0	0
	Sad	4	0	1	0	0	7	0
	Surprise	0	0	0	0	0	0	8

Seed 888		Data						
		Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Predict	Angry	7	0	0	0	0	0	0
	Disgust	0	5	0	0	0	0	0
	Fear	0	0	4	0	0	0	0
	Happy	0	0	0	8	0	0	1
	Neutral	0	0	0	0	3	1	1
	Sad	0	0	0	0	0	6	0
	Surprise	1	0	0	0	0	0	6

Table 3 introduces the confusion matrix of test set in first type image. Confusion matrices of train set always show perfect classification result, thus they are not explained in this report. When we consider seed 777, 5 expressions are classified well which are disgust, happy, neutral, sad and surprise. However, there are one miss in fear expression and five misses in angry expression. In seed 888, each of angry and sad expression has one miss, while surprise expression has two misses. From table 3, we can know that the classification result is totally dependent on random seeds, thus we need the average representations to see the performance of network.

Table 4. Five measurements, depending on random seeds in first type image

	Accuracy	Precision	FAR	FRR	Computational Time (sec)
Seed 777	0.86	0.91	0.14	0.09	1731
Seed 888	0.91	0.91	0.09	0.07	1690
Seed 999	0.81	0.81	0.19	0.17	1649
Seed 1111	0.84	0.83	0.17	0.17	1665
Seed 2222	0.88	0.9	0.12	0.11	1700
Average Result	0.86	0.87	0.14	0.12	1687

Table 4 shows the accuracy, precision, FAR, FRR and computational time, according to random seeds in first type image. From table 4, we can find that accuracy, precision, FAR and FRR are quite different, depending on random seeds. Compared to [8] and [9] which used CNN for FER, our average results are similar with them, meaning the network works well for FER tasks. Average computational time is quite long in first type of image which is 1687 seconds, but this will be decreased dramatically in second and third type images.

5.2 Second Type Data

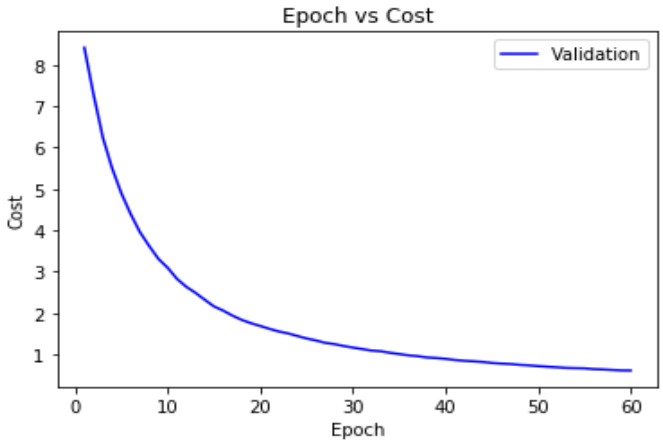


Figure 7. Epoch vs Cost curve for validation set in cross validation (second type image)

Figure 7 covers the cost curve of validation set in cross validation, depending on epoch. As in section 5.1, only one of 5 different curves is introduced because all curves are very similar. From figure 7, the validation cost curve shows clear decreasing graph without any increasement. Thus, it seems there is no overfitting.

Table 5. Confusion matrix of test set in second type image. Results from two of 5 random seeds are shown

Seed 777		Data						
		Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Predict	Angry	8	0	0	0	0	0	0
	Disgust	2	2	0	0	0	0	0
	Fear	0	1	8	0	0	0	0
	Happy	0	0	0	1	0	0	0
	Neutral	0	0	1	0	5	0	0
	Sad	0	0	0	0	0	7	0
	Surprise	0	0	0	0	0	0	8

Seed 888		Data						
		Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Predict	Angry	8	0	0	0	0	0	0
	Disgust	0	5	0	0	0	0	0
	Fear	0	0	4	0	0	0	0
	Happy	0	0	0	8	0	0	1
	Neutral	0	0	0	0	3	1	2
	Sad	0	0	0	0	0	6	0
	Surprise	0	0	0	0	0	0	5

Table 5 introduces the confusion matrix of test set in second type image. Confusion matrices of train set always give perfect classification result, thus they are not covered in this part. In seed 777, 4 expressions are classified well, while angry, disgust and fear expressions have misses. When we consider seed 888, sad expression has one miss and surprise expression possesses three misses. From table 5, we can know that the classification result is totally dependent on random seed as first type data.

Table 6. Five measurements, depending on random seeds in second type image

	Accuracy	Precision	FAR	FRR	Computational Time (sec)
Seed 777	0.91	0.89	0.09	0.09	734
Seed 888	0.91	0.91	0.09	0.07	742
Seed 999	0.88	0.92	0.12	0.12	746
Seed 1111	0.84	0.83	0.16	0.17	746
Seed 2222	0.88	0.87	0.11	0.08	743
Average Result	0.88	0.88	0.11	0.11	742

Table 6 gives the accuracy, precision, FAR, FRR and computational time, according to random seeds in second type image. Except for computational time, all these measurements are totally dependent on how data is shuffled. Compare to table 4, all the results are getting better than first type image which means we can build more stabilize and reliable FER system using second type image. Especially, it requires less than half computational time than first type data, meaning it is more suitable for real time processing.

5.3 Third Type Data

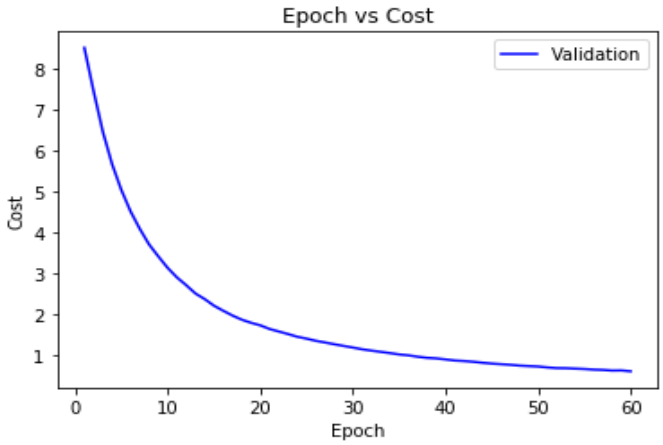


Figure 8. Epoch vs Cost curve for validation set in cross validation (third type image)

Figure 8 introduces the cost curve of validation set in cross validation, depending on epoch. As in section 5.1 and 5.2, only one of 5 different curves is covered. From figure 8, the validation cost curve shows clear decreasing shape, meaning there is no overfitting.

Table 7. Confusion matrix of test set in third type image. Results from two of 5 random seeds are covered

Seed 777		Data						
		Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Predict	Angry	8	0	0	0	0	0	0
	Disgust	2	2	0	0	0	2	0
	Fear	0	1	8	0	0	0	0
	Happy	0	0	0	1	0	0	0
	Neutral	0	0	1	0	5	0	0
	Sad	0	0	0	0	0	5	0
	Surprise	0	0	0	0	0	0	8

Seed 888		Data						
		Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Predict	Angry	8	0	0	0	0	0	0
	Disgust	0	5	0	0	0	0	0
	Fear	0	0	4	0	0	0	0
	Happy	0	0	0	8	0	0	1
	Neutral	0	0	0	0	3	1	2
	Sad	0	0	0	0	0	6	0
	Surprise	0	0	0	0	0	0	5

Table 7 explains the confusion matrix of test set in third type image. As in section 5.1 and 5.2, confusion matrices of train set always give perfect classification result, thus they are not introduced. In seed 777, 3 expressions are classified well, while other 4 expressions have misses. When we consider seed 888, sad expression has one miss and surprise expression possess three misses. From table 7, it shows that the classification result is affected by the random seed.

Table 8. Five measurements, depending on random seeds in third type image

	Accuracy	Precision	FAR	FRR	Computational Time (sec)
Seed 777	0.86	0.87	0.14	0.13	741
Seed 888	0.91	0.91	0.09	0.07	739
Seed 999	0.91	0.93	0.09	0.1	732
Seed 1111	0.86	0.86	0.14	0.12	731
Seed 2222	0.88	0.88	0.12	0.1	730
Average Result	0.88	0.89	0.12	0.1	734

Table 8 shows the accuracy, precision, FAR, FRR and computational time, depending on random seeds in third type image. Except for computational time, all these measurements are affected by shuffle of data. Compare to table 4, all the results are getting better than first type image, meaning we can make FER system more stabilize, reliable and fast using third type image. Compared to table 6, results are almost similar but has slightly better precision, FRR and computational time. From this comparison, we can know that CNN does not recognize the abnormal connection around middle part of second type data as edge.

Table 9. Performance comparison between three types of data

	First type image	Second type image	Third type image
Average Accuracy	0.86	0.88	0.88
Average Precision	0.87	0.88	0.89
Average FAR	0.14	0.11	0.12
Average FRR	0.12	0.11	0.1
Average Computational Time (sec)	1687	742	734

5.4 Feature Visualization

In this part, feature visualization from first type data will only be introduced because all types of data give similar shapes and understanding. First, filter-level feature visualization will be shown and then, layer-level feature visualization will be covered.

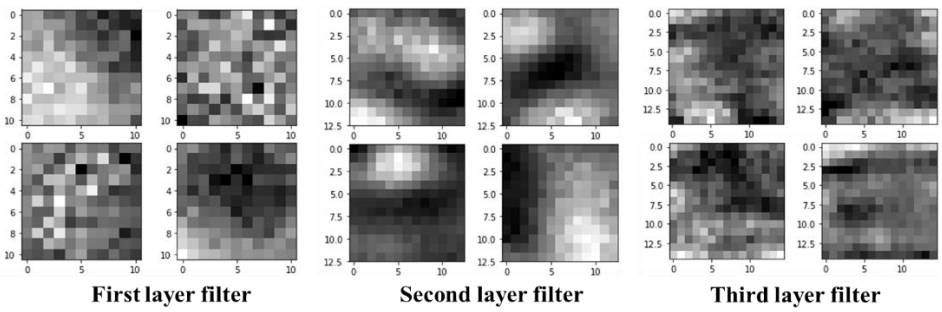


Figure 9. Feature visualization in filter-level. We can see that abstractions are stacked along with layer stacked. X-axis and Y-axis in all images are the width and height of images.

Figure 9 shows the feature visualization in filter-level. It is hard to see the interesting patterns in higher layer but we can know that the abstractions are stacked when layer is getting higher. In other words, first layer filter only gives the distribution of intensity, but second layer shows better shapes like edges and orientations. In third layer, it seems to make some patterns of facial parts but not clear.

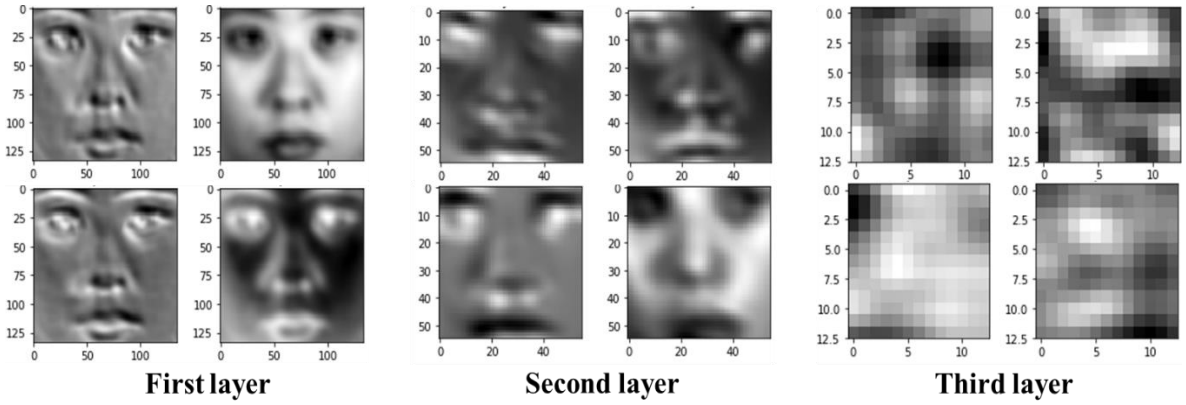


Figure 10. Feature visualization in layer-level. X-axis and Y-axis in all images are the width and height of images.

Figure 10 covers the feature visualization in CNN layer-level. First layer visualization gives almost same shape as input data. In second layer, it focuses on salient regions of face which are eyes, eyebrows, noses and mouth, meaning CNN layer holds the important structure information for facial expressions even if layer is stacked. However, in third layer, the shape of feature visualization is challenging to be interpreted by human, but machine understands in certain way and works well, according to the performance of system.

6. Conclusion

In this project, shallow-CNN with three types of data is used to build the facial expressions identification system. Using first type of image, the network yields 86% accuracy with 14% FAR and 12% FRR. This performance is good enough when we compare with the recent FER system built by CNN [8], [9]. When we consider second type of image, the network gives 88% accuracy with 11% FAR and 11% FRR which shows better performance than state of arts FER system using CNN with less execution time. Third type of data gives almost same performances as second type, meaning CNN can overcome the unrefined connection area caused when second type image was made.

Future work is to build the deeper CNN to get better performance as hand-crafted FER system. Also, deeper network will give better interpretation on feature visualization in filter-level. Good interpretation is getting more and more important to apply in real-world devices because it gives trust to people. Thus, building the CNN deeper is the most important future work to be done.

Code Repository

1. Data Preparation and Preprocessing

https://github.com/eoduself/ECE1512_Project/blob/master/ECE1512_Final%20Report_Preprocess.ipynb

2. Network and Performance Measurement

https://github.com/eoduself/ECE1512_Project/blob/master/ECE1512_Final%20Report_Network.ipynb

Bibliography

[1] Cowen, Alan S., and Dacher Keltner. "Self-report captures 27 distinct categories of emotion bridged by continuous gradients." Proceedings of the National Academy of Sciences 114.38 (2017): E7900-E7909.

[2] Brunelli, Roberto, and Tomaso Poggio. "Face recognition: Features versus templates." IEEE transactions on pattern analysis and machine intelligence 15.10 (1993): 1042-1052.

[3] Suk, Myunghoon, and Balakrishnan Prabhakaran. "Real-time mobile facial expression recognition system-a case study." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2014.

[4] Happy, S. L., Anjith George, and Aurobinda Routray. "A real time facial expression classification system using local binary patterns." 2012 4th International conference on intelligent human computer interaction (IHCI). IEEE, 2012.

[5] Ghimire, Deepak, et al. "Facial expression recognition based on local region specific features and support vector machines." Multimedia Tools and Applications 76.6 (2017): 7803-7821.

[6] Edwards, Gareth J., Christopher J. Taylor, and Timothy F. Cootes. "Interpreting face images using active appearance models." Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition. IEEE, 1998.

[7] Jung, Heechul, et al. "Joint fine-tuning in deep neural networks for facial expression recognition." Proceedings of the IEEE International Conference on Computer Vision. 2015.

[8] Ijjina, Earnest Paul, and C. Krishna Mohan. "Facial expression recognition using kinect depth sensor and convolutional neural networks." 2014 13th International Conference on Machine Learning and Applications. IEEE, 2014.

[9] Lopes, André Teixeira, et al. "Facial expression recognition with convolutional neural networks: coping with few data and the training sample order." Pattern Recognition 61 (2017): 610-628.

[10] Lyons, Michael, et al. "Coding facial expressions with gabor wavelets." Proceedings Third IEEE international conference on automatic face and gesture recognition. IEEE, 1998.

[11] Kazemi, Vahid, and Josephine Sullivan. "One millisecond face alignment with an ensemble of regression trees." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

[12] ″Facial Point Annotations.” i·Bug - Resources - Facial Point Annotations, ibug.doc.ic.ac.uk/resources/facial-point-annotations/.

[13] Gandhi, Rohith, and Rohith Gandhi. “Build Your Own Convolution Neural Network in 5 Mins.” Towards Data Science, Towards Data Science, 18 May 2018, towardsdatascience.com/build-your-own-convolution-neural-network-in-5-mins-4217c2cf964f.

[14] Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." European conference on computer vision. Springer, Cham, 2014.