Abstract

In this paper we are classifying images according to various facial expressions. The convolutional neural network pre-trained on the ImageNet database is used to extract features from our images and then these are later used for detecting facial expressions. The classification is done by feeding the extracted features from the pre-trained CNN to a linear Support Vector Machine. All experiments are performed on two publicly available datasets such as JAFFE(Japanese Female Facial Expression) and CK+ database. The feature extraction and pre-training on neural networks which was originally used for object recognition can be transferred and used for facial expression recognition. Before feeding the extracted features to the linear SVM classifier PCA transformation is done on it and we experiment with various values of PCA .

# Introduction:

Traditional method of hand-crafted feature extraction are using HOG , SIFT, local binary patterns etc. There is a demand for automatic feature extraction for object detection or image classification and this can be done using Deep Convolution Neural Networks. A CNN can be viewed as a framework combining a feature extractorand a classifier. Convolutional layersbehave as feature extractors that learn the representations automatically from the input data. In the Earlier layers of the CNNwe learn features such as (shapes, edges, and colour blobs ) whereas in the Later layers we learn features more specific to the original dataset. The learned features are fed into the last fully connected layers to classify the data into one of the classes. In this project we use the VGG-19 network pre-trained on the Image Net dataset used for extracting features from the target images dataset this is called Transfer learning.

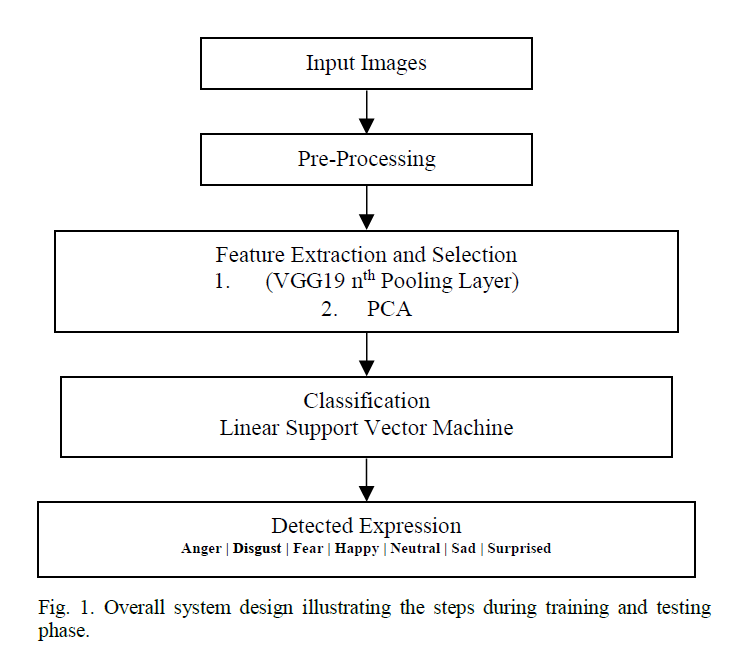
Finding an existing neural network that accomplishes a similar task to the one you are trying to tackle then reusing the lower layers of this network is called transfer learning. It not only speeds up training considerably, but also requires significantly less training data. There are two ways of applying transfer learning - the last fully connected layers are removed, and the rest of the network is used as a fixed feature extractorfor the target domain and the second method is Fine-tuning the weights of the CNNby continuing the backpropagation on the target domain. It can be performed on all layers or it is possible to freeze some of the weights of earlier layers and fine-tune deeper layers of the CNN.

To decide the kind of Transfer learning method to opt for there are a few parameters to consider viz size and the similarity of the target dataset with respect to the original dataset. When the Target dataset is *small and similar* to the original dataset then we go for fixed feature extraction and use a Linear Classifier for training as Fine tuning would lead to overfitting as the dataset is small.When the *Target dataset is large and similar* to the original dataset then we go for fine-tuning the weights of CNN and continue back propagation on target set. When the *Target dataset is small but very different* to the original dataset then we go for feature extraction and use a Linear Classifier for training. When the *Target dataset is small but very different* to the original dataset then we must train the CNN from scratch.

# System Design

The structure of the project consist of four main steps: Pre-processing the read input images, Feature Extraction using the pre-trained VGG-19 on the Imagenet dataset, Feature Selection and dimensionality Reduction using Principal Component Analysis(PCA) and Classifying on a linear SVM to recognise the facial expressions.

The images read from the database are originally of the dimension 226x226 . We must pre-process these images to 224x224 and normalize the intensity. We load the pre-trained VGG-19 model trained on ImageNet data. These images are given as input to the pre-trained VGG19 network. The VGG19 contains five pooling layers. The output of each pooling layer is extracted and stored as a set of features for further processing. This way we have a set of 5 feature vectors. The PCA for each of these feature vectors is computed for dimensionality reduction and feature selection. Now the dataset is then split into train and test data in 80 % to 20% ratio. Thus, training and the test data each consists of a set of 5 feature vectors . These training features are then fed to a linear SVM classifier for recognise the facial expression. Using the test features The testing accuracy, the 10-fold cross validation accuracy , Leave One Out accuracy cross validation is also performed on each of the feature vectors.

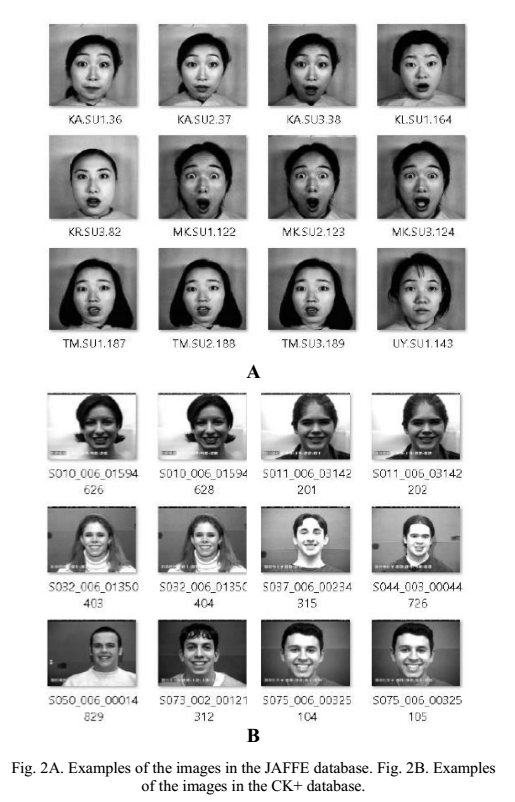


# Dataset Description

This project is implemented on two datasets – JAFFE(Japanese Female Facial Expression) dataset and the CK+(Extended Crohn Kahn) dataset. In JAFFE datasets, the seven expressions were: anger, happy, sad, disgust, fear, surprised and neutral. And in the Ck+ dataset the seven expressions were: anger, happy, sad, disgust, fear, surprised and contempt.

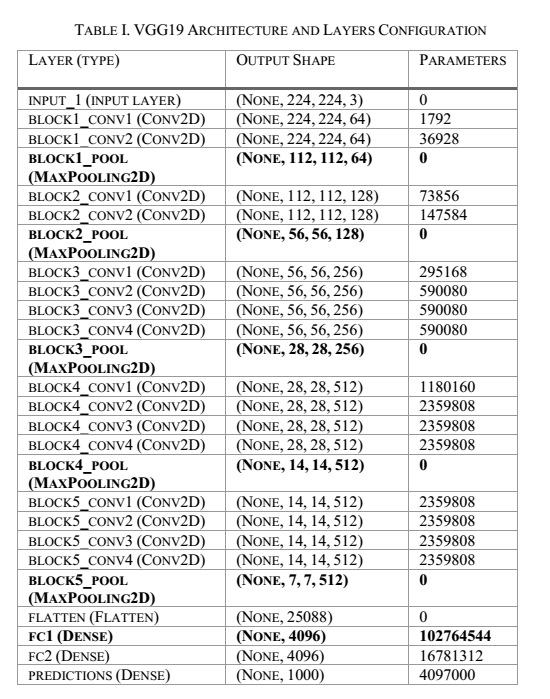
The JAFFE dataset consists of 213 images from 10 Japanese female subjects. For each subject, there are around 4 images for each of the seven expressions (including neutral). All images are grayscale images of size 256 by 256 pixels. The database is separated into seven groups based on the seven classes of expressions for this experiment.

The CK+ dataset comprises of 100 university students with age between 18 and 30 years old. The subjects formed a diverse group of individuals comprising of both male and female subjects of ethnicity belonging to one of Asian, African American, American or South American. All images are grayscale of size 640 by 480 pixels. A subset of the CK+ database with 10 subjects, 5 male and 5 females for each of the seven expressions resulting in a total of 210 images were selected for this study.



# Methodology

The training on both the datasets JAFFE and CK+ was done using the pre-trained VGG19 networks provided in the Keras package for Python. The pre-trained VGG19 networks was used for evaluating the performance of the CNN as a feature extractor, and for transferring the features for facial expression recognition.



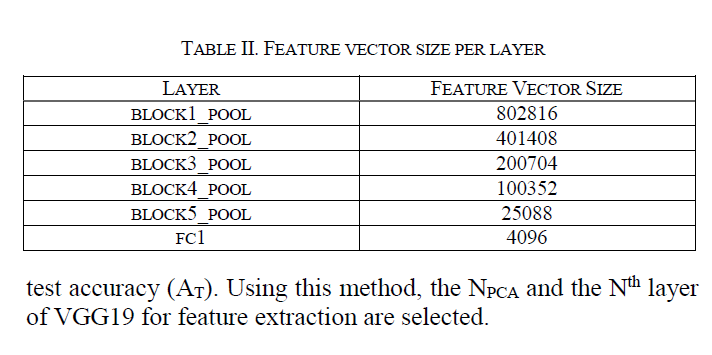
## Feature Extractor

The layer configuration of the pre-trained VGG19 networks consists of convolution layers followed by a pooling layer for 5 such hidden layers and the Fully connected layer. Keras in-built implementation is used in evaluating the features extracted from all the pooling layers (block1\_pool, block2\_pool, block3\_pool, block4\_pool, and block5\_pool) . The size of each of the pooling layer’s output is shown in the next table. The feature vectors are too large to train on a classifier hence PCA is applied to each pooling layer. To choose the best number of PCA components (NPCA) different values such as NPCA = {50,100,150,200} are evaluated.

## B. Feature Selection

The extracted features from the previous step are transformed to a lower dimensional vector by applying PCA transformation based on the chosen NPCA value. In this we must decide which NPCA  value and the Nth VGG19 layer for feature extraction.

The parameter selection is based on a two-step method. The first step is to select those parameters that provide the two highest accuracies on the training set based on a Leave One out validation (ALOO). Next, select the final parameters based on the parameters that have the least difference between the ALOO and corresponding test accuracy (AT). Using this method, the NPCA and the Nth layer of VGG19 for feature extraction are selected.



## 

## Classification

The target datasets used are different from the source dataset and are also smaller in size hence transfer learning is done by means of a linear classifier which is trained on the features extracted from the pretrained CNN. A Support Vector Machine with a linear kernel is used as the classifier.

## Validation

The training data for both datasets were split into 80% for training the classifier and 20% was used for testing. These results are validated based on a 10-fold cross-validation, and due to the small size of the datasets, a leave-one-out validation is also performed. Based on the validation results, the highest accuracies for JAFFE and CK+ datasets were achieved when NPCA = 200 and NPCA = 100 respectively.

# Testing

Two datasets JAFEE and CK+ are taken and using the pre-trained CNN on Imagenet dataset their features are extracted, then the dimensionality of the features is reduced by PCA.

Here we compute the results of the Feature expression for various values of PCA for the extracted features of the datasets with different NPCA=[50,100,150,200] values to see which PCA value yields the best result.

**JAFFE Dataset**

|  |  |  |  |
| --- | --- | --- | --- |
| PCA=200 | 10-Fold Cross Validation Scores | Leave-One-Out cross Validation Scores | Test Accuracy |
| block pool 1 | 81.76 | 83.52 | 72.093 |
| block pool 2 | 86.47 | 87.64 | 74.418 |
| block pool 3 | 88.823 | 90.588 | 79.06 |
| **block pool 4** | **90.** | **91.76** | **86.046** |
| block pool 5 | 90. | 90.588 | 86.046 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PCA=150 | 10-Fold Cross Validation Scores | | Leave-One-Out cross Validation Scores | | Test Accuracy | |
| block pool 1 | 82.941 | | 85.882 | | 74.418 | |
| block pool 2 | 87.058 | | 88.823 | | 76.744 | |
| block pool 3 | 87.647 | | 90.588 | | 76.744 | |
| **block pool 4** | **88.235** | | **90.588** | | **86.046** | |
| block pool 5 | 89.411 | | 91.176 | | 83.720 | |
|  |  | |  | |  | |
| PCA=100 | | 10-Fold Cross Validation Scores | | Leave-One-Out cross Validation Scores | | Test Accuracy | |
| block pool 1 | | 80.0 | | 84.117 | | 79.069 | |
| block pool 2 | | 87.05 | | 89.4117 | | 81.395 | |
| block pool 3 | | 87.05 | | 89.4117 | | 74.418 | |
| **block pool 4** | | **90.0** | | **91.764** | | **88.372** | |
| block pool 5 | | 88.235 | | 90.588 | | 86.046 | |

|  |  |  |  |
| --- | --- | --- | --- |
| PCA=50 | 10-Fold Cross Validation Scores | Leave-One-Out cross Validation Scores | Test Accuracy |
| block pool 1 | 79.411 | 80.0 | 72.093 |
| block pool 2 | 84.705 | 87.05 | 81.395 |
| block pool 3 | 89.411 | 91.176 | 81.395 |
| **block pool 4** | **90.588** | **92.352** | **93.023** |
| block pool 5 | 87.0588 | 89.411 | 86.046 |

**CK+ Dataset**

|  |  |  |  |
| --- | --- | --- | --- |
| PCA=200 | 10-Fold Cross Validation Scores | Leave-One-Out cross Validation Scores | Test Accuracy |
| block pool 1 | 95.2941 | 97.0238 | 90.476 |
| block pool 2 | 95.88 | 97.619 | 90.476 |
| block pool 3 | 96.470 | 97.6190 | 92.857 |
| **block pool 4** | 96.470 | 98.214 | 92.857 |
| block pool 5 | 96.470 | 98.80 | 100 |

|  |  |  |  |
| --- | --- | --- | --- |
| PCA=150 | 10-Fold Cross Validation Scores | Leave-One-Out cross Validation Scores | Test Accuracy |
| block pool 1 | 95.29 | 97.023 | 90.476 |
| block pool 2 | 95.882 | 98.214 | 90.476 |
| block pool 3 | 96.470 | 97.6190 | 92.857 |
| **block pool 4** | 96.470 | 98.214 | 92.857 |
| block pool 5 | 96.470 | 98.80 | 100 |

|  |  |  |  |
| --- | --- | --- | --- |
| PCA=100 | 10-Fold Cross Validation Scores | Leave-One-Out cross Validation Scores | Test Accuracy |
| block pool 1 | 95.29 | 97.02 | 90.47 |
| block pool 2 | 95.88 | 97.61 | 90.47 |
| block pool 3 | 96.47 | 97.61 | 92.85 |
| **block pool 4** | **96.47** | **98.21** | **92.85** |
| block pool 5 | 96.47 | 98.88 | 100 |

|  |  |  |  |
| --- | --- | --- | --- |
| PCA=50 | 10-Fold Cross Validation Scores | Leave-One-Out cross Validation Scores | Test Accuracy |
| block pool 1 | 95.294 | 97.023 | 92.857 |
| block pool 2 | 95.294 | 97.0238 | 90.476 |
| block pool 3 | 97.058 | 97.619 | 92.857 |
| **block pool 4** | 97.058 | 97.619 | 92.857 |
| block pool 5 | 95.88 | 98.214 | 100 |

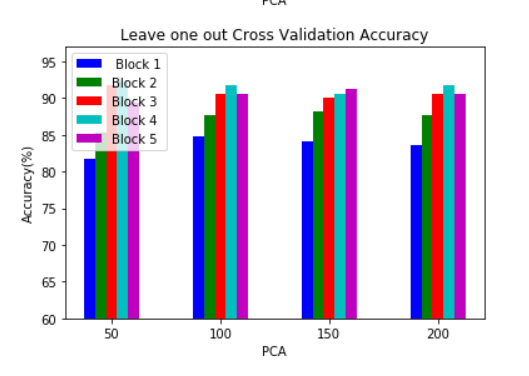
# Results

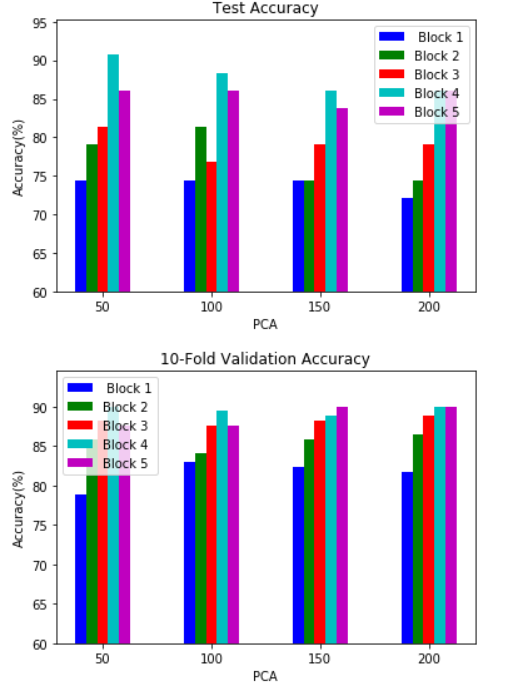
The following are the results for facial recognition using pre-trained CNN on the JAFFE and CK+ datasets . We observe that the earlier layers of the CNN provide a better performance than later layers.

**JAFFE Dataset**

After the feature selection it was found that the best result for the JAFFE dataset is found when PCA=200 in block pool -4 .

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10-Fold Cross Validation Scores | Leave-One-Out cross Validation Scores | Test Accuracy |
| block pool 1 | 81.76 | 83.52 | 72.093 |
| block pool 2 | 86.47 | 87.64 | 74.418 |
| block pool 3 | 88.823 | 90.588 | 79.06 |
| **block pool 4** | **90.** | **91.76** | **86.046** |
| block pool 5 | 90. | 90.588 | 86.046 |

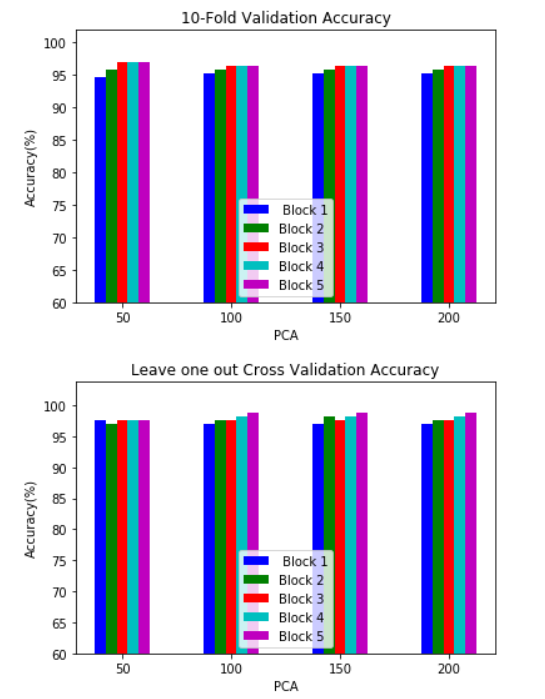
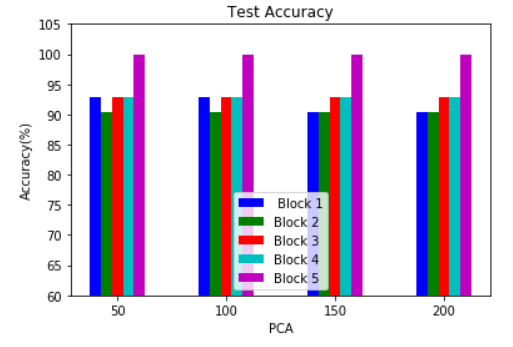




**CK+ Dataset**

After the feature selection it was found that the best result for the CK+ dataset is found when PCA=100 in block pool -4 .

|  |  |  |  |
| --- | --- | --- | --- |
| PCA=100 | 10-Fold Cross Validation Scores | Leave-One-Out cross Validation Scores | Test Accuracy |
| block pool 1 | 95.29 | 97.02 | 90.47 |
| block pool 2 | 95.88 | 97.61 | 90.47 |
| block pool 3 | 96.47 | 97.61 | 92.85 |
| **block pool 4** | **96.47** | **98.21** | **92.85** |
| block pool 5 | 96.47 | 98.88 | 100 |



# Conclusion

In this project we use pre-trained VGG-19 on Imagenet dataset as a fixed feature extractor to extract features form our JAFFE and CK+ datasets and then fed these values to a linear SVM classifier . We also did 10-fold Validation and Leave One Out cross validation to get a better gauge of the performance metrics of the model.

The results suggest that representations learned from pre-trained networks trained for a task such as object detection can be transferred and used for a different task such as facial expression recognition. Furthermore, for a small dataset, using features from earlier layers of the network provide better accuracy.

# References

* Ravi, Aravind. (2018). Pre-Trained Convolutional Neural Network Features for Facial Expression Recognition. <https://arxiv.org/ftp/arxiv/papers/1812/1812.06387.pdf>
* Géron, Aurélien. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (p. 219, 480-482). O'Reilly Media. Kindle Edition.
* Rosebrock, Adrian.” Transfer learning with Keras and Deep Learning”. Pyimagesearch, 20 May 2019, <https://www.pyimagesearch.com/2019/05/20/transfer-learning-with-keras-and-deep-learning/>