Improved Facial Expression Recognition Based on DWT Feature for Deep CNN

Ankush Dey MDS202108

Ritirupa Dey MDS202136

Anjali Pualia MDS202107

Instructor
Dr. Kavita Sutar

May 3, 2023

Abstract

Facial expression recognition (FER) has become one of the most important fields of research in pattern recognition. In this paper, we propose a method for the identification of facial expressions of people through their emotions. It has many different applications in various fields such as security-surveillance, artificial intelligence, military and police services, and psychology, among others. Facial expressions are classified into six basic categories; namely, anger, disgust, fear, sadness, happiness, and surprise —a neutral expression was also added to this group. Experiment was performed on the CK+ database and JAFFE face database. This paper combines 4 steps to create an efficient emotion detection algorithm. Which are :

1. Viola Jones Algorithm to locate face and facial features
2. Using Contrast Limited Adaptive Histogram Equalization (CLAHE) for facial image enhancement.
3. Discrete Wavelet Transformation (DWT) for extracting facial features.
4. Deep CNN which directly uses the extracted features for training.

Contents

Work Contribution

Ankush Dey	Cascade classifiers, Discrete wavelet transformation, Current state of matters
Anjali Pugalia	Integral Images, Adaboost Algorithm
Ritirupa Dey	Convolutional Neural Network, Limitations of Viola Jones, Implementation of of the algorithm(Coding Part)

Viola-Jones Algorithm

Viola Jones brings together new algorithms and insights to construct a framework for robust and extremely rapid visual detection. It constructs a frontal face detection system working only with information present in a single grey scale image. There are three key contributions.

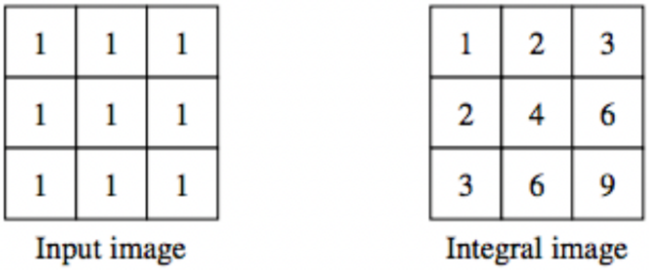
1. Introduction of a new image representation called the “Integral Image”.
2. A simple and efficient classifier which is built using the AdaBoost learning.
3. A method for combining classifiers in a “cascade” which allows back- ground regions of the image to be quickly discarded

Integral Images

The first contribution of this paper is a new image representation called an integral image this enables evaluation of facial features very quickly. Also known as summed area table, it is an algorithm for quickly and efficiently computing the sum of values in a rectangle subset of a pixel grid. The integral image at location x, y contains the sum of the pixels above and to the left of x, y inclusive :

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

where $i(x,y)$ is the pixel value of the original image and $ii(x', y')$ is the corresponding image integral value. The image below shows an example of integral image.



1	1	1
1	1	1
1	1	1

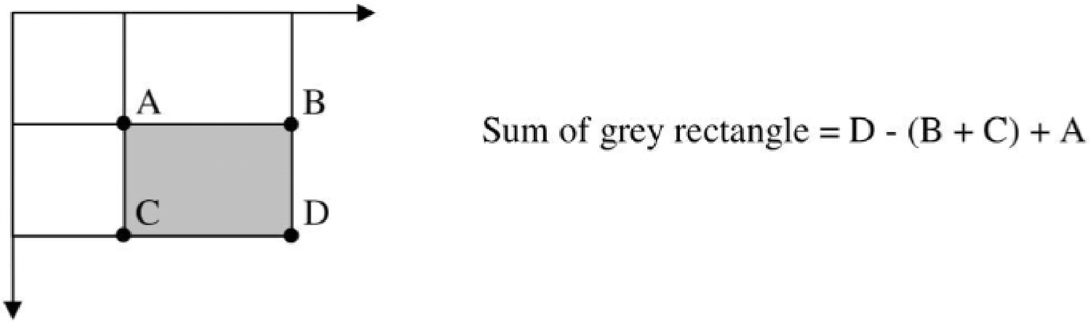
Input image

1	2	3
2	4	6
3	6	9

Integral image

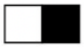
Using this integral image we can compute the sum of any rectangular area efficiently. The sum of pixels in rectangle ABCD can be calculated with only four values from integral image:

$$\sum_{(x,y) \in ABCD} i(x,y) = ii(A) + ii(D) - ii(B) - ii(C)$$

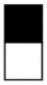


Instead of using pixels this paper uses features. It obtains them by applying these Haar like filters to sub-windows of the images. Value of the feature is the difference between the sum of the pixels within the rectangular regions. We subtract the sum of the pixels in the white rectangle from the black one. The paper uses three kind of features. The value of a two-rectangle feature is the difference between the sum of the pixels within two rectangular regions.

Type 1

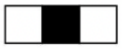


Type 2

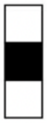


A three-rectangle feature computes the sum within two outside rectangles subtracted from the sum in a center rectangle and also vice versa.

Type 3

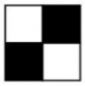


Type 4



Finally a four-rectangle feature computes the difference between diagonal pairs of rectangles.

Type 5



The paper makes a convincing case of using these features instead of pixels directly. As these features can encode ad hoc domain knowledge which is usually difficult to learn using a finite quantity of training data. Moreover a feature based system is faster than a pixel based one. The Viola Jones Algorithm scans the input image at many scales starting at the base scale in which the faces are detected at a size 24 × 24. So a 384 × 288 pixel image is scanned at 12 scales where each scale is a factor 1.25 larger than the last.

AdaBoost Algorithm

We generate approximately 160000 rectangle features associated with each image sub window. It is evident that the number of features is far larger than the number of pixels which makes it a time consuming and expensive process. Not all features are important and a small number of them can be combined to form an effective classifier. Some features are expected to give consistently high values and Viola Jones uses a modified version of AdaBoost to find these features. AdaBoost is an aggressive mechanism for selecting a small set of good classification functions which nevertheless have significant variety.
A single weak classifier is defined as:

$$h(f, x, p, \theta) = \begin{cases} 1 & pf(x) < p\theta \\ 0 & \text{otherwise} \end{cases}$$

Where f is the feature θ is the threshold p is the polarity indicating the direction of the inequality and x is a 24 × 24 pixel sub window of the image.

Now an important part of modified AdaBoost is determining the best feature, polarity and threshold for calculation. To find these values Viola Jones uses simple brute force.

Cascade Classifiers

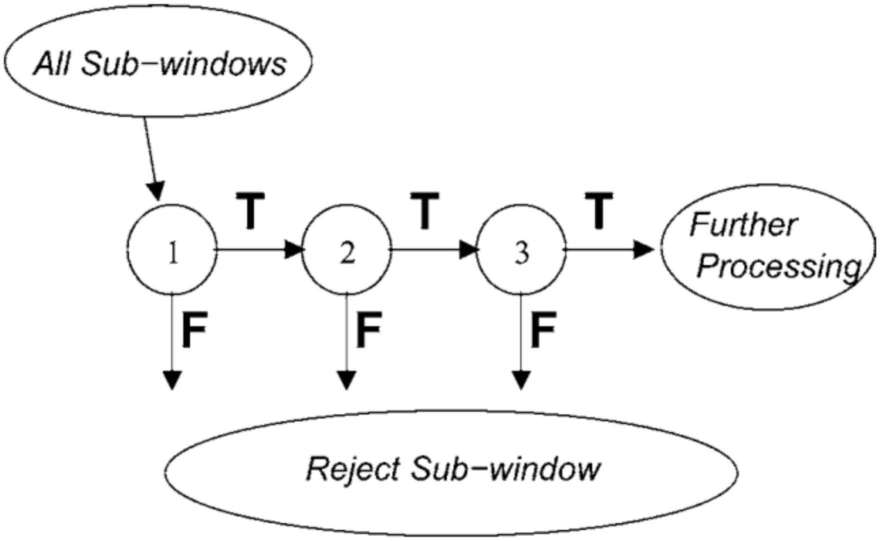
The Attentional cascade

Cascading in classifiers is a technique used in machine learning and computer vision to improve the efficiency and accuracy of classification algorithms. It involves using a series of classifiers that are arranged in a hierarchical fashion, with each subsequent classifier becoming more specialized and accurate in its classification task.The key insight is that smaller, and therefore more efficient, boosted classifiers can be constructed which reject many of the negative sub-windows while detecting almost all positive instances. The stages are constructed by training classifiers using Adaboost. The Algorithm learns a series of weak classifiers to combine them to make a strong classifier. The detection performance of the two-feature classifier is far from acceptable as a face detection system. Nevertheless the classifier can significantly reduce the number of sub-windows that need further processing with very few operations:

1. Evaluate the rectangle features (requires between 6 and 9 array references per feature)

2. Compute the weak classifier for each feature (requires one threshold operation per feature).

3. Combine the weak classifiers (requires one multiply per feature, an addition, and finally a threshold)



Training

This is an overview of each step in the training process for a cascade of classifier:

1. Collecting positive and negative samples: This involves gathering a set of examples that are representative of the classes you want to classify.

2. Preparing the samples: The samples need to be preprocessed in order to extract features that are relevant to the classification task. This might involve resizing, cropping, or filtering the images, or extracting numerical features from other types of data.

3. Creating a positive samples vector: A positive samples vector is a numerical representation of the positive samples that can be used as input to a classifier.

4. Creating a negative samples vector: Similar to step 3, a negative samples vector is created by encoding the features extracted from negative samples as a set of numerical values.

5. Training the classifier: The classifier is trained using the positive and negative sample vectors, along with a set of labels that indicate the class of each sample. The training process involves adjusting the parameters of the classifier to minimize the error rate on the training data.

6. Testing the classifier: The trained classifier is tested on a separate set of validation data to evaluate its performance. This helps to determine how well the classifier generalizes to new data.

7. Tweaking the classifier: If the classifier is not performing well, it may be necessary to adjust its parameters or features to improve its performance.

8. Using the classifier: Once the classifier has been trained and tested, it can be used to classify new data.

The false positive rate of the cascade classifier is given by:

$$F = \prod_{i=1}^{K} f_i$$

F is the positive rate of the cascaded classifier, K is the number of classifiers, f_i is the false positive rate of the ith classifier.

The detection rate of the cascade is:

$$D = \prod_{i=1}^{K} d_i$$

where d_i is the detection rate of the ith classifier.

Given concrete goals for overall false positive and detection rates, target rates can be determined for each stage in the cascade process. For example a detection rate of 0.9 can be achieved by a 10 stage classifier if each stage has a detection rate of 0.99 (since $0.9 \approx 0.99^{10}$). While achieving this detection rate may sound like a daunting task, it is made significantly easier by the fact that each stage need only achieve a false positive rate of about $30\%(0.30^{10} \approx 6 \times 10^{-6})$

We can also evaluate the number of features since it is a probabilistic process. Analysis of the image distribution allows us to predict how the process will behave. The expected number of features can be given as:

$$N = n_0 + \sum_{i=1}^{K} (n_i \prod_{j<i} p_j)$$

where N = expected number of features evaluated,
n_i = expected number of features of ith classifier.
p_i = positive rate of the ith classifer.

Benefit of cascade classifiers

There are several benefits of using a cascade of classifiers for object detection, which operates pointwise by evaluating each object candidate at multiple stages before making a final decision:

1. Reduced computation time
2. Increased accuracy.
3. Robustness to variation

Drawback of cascade classifiers

1. the training set of negative examples would have to be relatively small.
2. it requires careful tuning of several parameters, including the number of stages, the number of features per stage etc.

Discrete wavelet transformation

Wavelet:-A wavelet is a waveform of effectively limited duration that has an average value of zero and nonzero norm.
It is used to used extract information from many different kinds of data, including audio signals and images.

Why choose DWT

1. Wavelets are better suited for analyzing signals with transient or non-stationary features. Unlike Fourier transformation, which decomposes a signal into its component frequencies at all scales, wavelet transformation decomposes a signal into its component parts at different scales, using different wavelet functions that are appropriate for each scale.
2. Wavelets can provide a more localized representation of a signal. Unlike Fourier transformation, which provides a global representation of a signal in terms of its frequency components, wavelet transformation can capture both local and global features of a signal at different scales. This makes wavelets particularly useful for feature extraction and pattern recognition in images, as they can capture both local and global features of an image at different scales.
3. Wavelet transformation can be used to represent a signal with fewer coefficients than Fourier transformation, while still preserving important features of the signal
4. Wavelets can be more computationally efficient than Fourier transformation. Wavelet transformation requires fewer computations than Fourier transformation for signals with limited bandwidth or sparse frequency components.

Here we will deal with discrete wavelet transformation:
let f(x) is a function on the spatial domain. $f(x) \in L^3(R)$ $\varphi(x)$ is the scaling function and $\psi(x)$ is the wavelet function.

$$f(x) = \sum_k W_\varphi(j_0, k)\varphi_{j_0,k}(x) + \sum_{j=j_0}^{\infty} \sum_k W_\psi(j,k)\psi_{j,k}(x)$$

where the coefficient $W_\varphi(j_0, k)$ and $W_\psi(j,k)$ can be written as:

$$W_\varphi(j_0, k) = \frac{1}{\sqrt{M}} \sum_x f(x)\varphi_{j_0,k}(x)$$

$$W_\psi(j,k) = \frac{1}{\sqrt{M}} \sum_k f(x)\psi_{j,k}(x)$$

Filters of DWT

There are two types of filters that we dealt in DWT:

High pass filter

- used to extract high-frequency component
- contains information about the sharp changes and edges in the signal.

Low pass filter

- used to extract low-frequency components.
- contains information about the overall trend and shape of the signal.

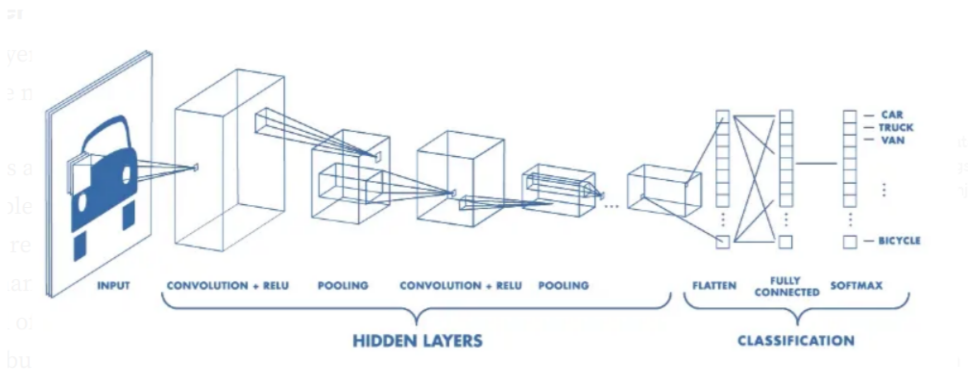
CNNs

A Convolutional Neural Network, also known as a CNN or ConvNet, is a class of Neural Networks that specializes in processing data that has a grid-like topology, such as an image. A digital image is a binary representation of visual data. It contains a series of pixels arranged in a grid-like fashion that contains pixel values to denote how bright and what color each pixel should be.
The human brain processes a huge amount of information the second we see an image. Each neuron works in its own receptive field and is connected to other neurons in a way that they cover the entire visual field. Just as each neuron responds to stimuli only in the restricted region of the visual field called the receptive field, each neuron in a CNN processes data only in its receptive field as well. So the CNN works analogously to the human brain.
CNNs are a type of multi-layer neural network that can discern visual patterns from pixel images. The layers are arranged in such a way so that they detect simpler patterns first (such as lines, curves) and more complex patterns (faces, objects) further along. Thus, CNNs in a way provide vision to computers.

Architecture of ConvNets

A CNN is typically composed of 3 layers:

1. a convolutional layer
2. a pooling layer
3. a fully connected layer



Now let us discuss a bit about each of these layers.

The Convolutional layer

The key components of this convolutional layer are:

1. Filters: A convolutional layer consists of a set of filters, which are small matrices of weights that need to be learnt during training. Each filter is typically square and has a small spatial extent, such as 3 X 3 or 5 X 5 pixels.
2. Sliding Window: To apply the filters to the input data, the filters are slid over the input image or feature map in a process called convolution. At each position, the filter weights are multiplied by the corresponding input values, and the results are summed to produce a single output value.
3. Feature Map: The output of the convolution operation at each position is stored in a matrix, called a feature map. Each filter produces a separate feature map, which represents the presence of a particular local pattern or structure in the input data.
4. Padding: To avoid losing information at the edges of the input data, the input data can be padded with zeros before applying the filters, otherwise pixels on the edges get processed by fewer filters than the pixels on the inner side. It also prevents shrinkage of the image. When an image matrix is multiplied by the filter in the convolution layer, it decreases the dimensions of the image according to the formula $\lfloor \frac{n+2p-f}{s} \rfloor + 1$ where,
   n= dimensions of the image matrix
   p= amount of padding
   f= dimension of the filter
   s= stride
5. Stride: The filter can be applied with a stride, which determines the distance between each position where the filter is applied. A larger stride reduces the size of the output feature map, while a smaller stride preserves more spatial information.

Now let us put all these components together and see how the convolution layer works.

- The convolutional layer executes the convolution operations. The Kernel is the component in this layer that performs the convolution operation. Until the complete image is scanned, the kernel makes horizontal and vertical adjustments dependent on the stride rate. The kernel is less in size than a picture, but it has more depth. This means that if the image has 3 RGB channels, the kernel height and width will be modest spatially, but the depth will span all three.

- This layer has another important component known as the Non-linear activation function. The outputs of the convolution operation are passed through a non-linear activation function called tanh, ReLU,etc.

The Pooling Layer

The pooling layer is in charge of reducing dimensionality. It replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs, thus reducing the amount of computing power required to process data. The most popular types are max pooling in which the max value from the area covered by the kernel is returned and average pooling in which the average of all values in the part of the image covered by the kernel is returned.

Fully Connected Layer(FC)

Fully connected layers or dense layers are used in CNNs to perform Classification/ Regression tasks on the output features learnt by the convolutional layers. The convolutional layers in a CNN learn to identify local patterns in the input layer, but they do not account for the spatial relationships between these patterns or their overall meaning. Fully connected layers take the learned features from the convolutional layers and use them to classify the input data. The fully connected layer flattens the output of the convolutional layers into a 1D vector which can be fed to a traditional neural net to perform classifcation/regression. After that, the flattened vectore is sent via a few additional FC layers, where the mathematical functional operations are normally performed like in an ordinary neural net.

How CNN is used in our case

In our project, we first pass an image to the Viola Jones Algorithm for face recognition, then the detected face is passed onto the Clahe function which performs image enhancement, then this image is processed using DWT which identifies the significant facial features and then these facial features are fed into a **CNN which is used for classification in our case as it classifies the image into one of the 7 emotions.**

Comparison of results between the traditional method and the method proposed in this project

The comparison between different approaches and our approach for the **JAFFE** face database.

Approach	Recognition Rate %
SVM	95.60
Gabor	93.30
2-Channel CNN	94.40
Deep CNN	97.71
Normalization + DL	88.73
Viola-Jones + CNN	95.30
Proposed Method	98.63

The comparison between different approaches and our approach for the **CK+** face database.

Approach	Recognition Rate %
SVM	95.10
Gabor	90.62
2-Channel CNN	95.00
Deep CNN	95.72
Normalization + DL	93.68
Viola-Jones + CNN	95.10
Proposed Method	97.05

As we can see, our proposed method yielded the best results amongst all the traditional ML/DL approaches to this problem of FER.

Limitations of Viola Jones

Though Viola Jones is quite efficient in detecting faces, it does have some shortcomings:

1. Cannot detect faces with goggles, spectacles or any kind of objects which cover the eyes.
2. It can't properly detect side faces.
3. It is unable to detect when the eye is partially closed.
4. Doesn't give proper results when the face is tilted.
5. Cannot detect images with low resolution.
6. Unable to detect faces in a blurred image.
7. Cannot detect small faces in a group photograph.

Current state of matters

Facial emotion recognition (FER) is an emerging and significant research area in the pattern recognition domain. We can use either ML-based method or a deep learning-based method for efficient face recognition.

Conventional ML-based approaches for FER.

references	datasets	accuracy	techniques
Varma et al.	FECR	98.40 (SVM) 87.50 (HMM)	PCA and LDA for feature selection and SVM and HMM classifier
Reddy et al.	CK+	98.00	Haar Wavelet Transform (HWT), Gabor wavelets and nonlinear principal component analysis (NLPCA) feature extractor and SVM classifier
Sajjad et al. Nazir et al.	MMI JAFFE CK+ CK+ MMI	99.1 on MMI accuracy (92) on JAFFE accuracy (90) on CK+ 99.6 by using only 32 features with KNN	ORB SIFT SURF using SVM HOG, DCT features and KNN, SMO and MLP for classification
Siddiqi et al	CK JAFFE USTC-NVIE Yale FEI	99.33 96.50 99.17 99.33 99.50	Features are extracted by using Chan–Vese energy function, Bhattacharyya distance function wavelet decomposition and SWLDA and hidden Markov model (HMM) is used for prediction

Conventional deep-learning based approaches for FER

References	Datasets	Accuracy	Techniques
Li et al	SMIC CASME CASME II	55.49 54.44 59.11	3D flow-based CNN model for video-based micro-expression recognition
Xie et al	CK+ JAFFE	With 10-fold cross-validation accuracy 95.88 99.32	Expressional Region Descriptor (SERD) and Multi-Path Variation-Suppressing Network (MPVS-Net)
Lopes et al.	CK+	Average of C_{6class}(96.76) Average of C_{bin}(98.92)	Convolutional neural network and specific image preprocessing steps
Lopes e al.	JAFFE	Average of C_{6class} (72.89) Average of C_{bin} (90.96)	Convolutional neural network and specific image preprocessing steps
Al-Shabi et al.	CK+	99.1	Dense SIFT and regular SIFT are merged with CNN features

References

- Improved Facial Expression Recognition Based on DWT Feature for Deep CNN:- Ridha Ilyas Bendjillali, Mohammed Beladgham, Khaled Merit and Abdelmalik Taleb-Ahmed
- Robust Real-Time Face Detection:- PAUL VIOLA, MICHAEL J. JONES
- Study of Viola-Jones Real Time Face Detector: Kaiqi Cen
- The wavelet transform:- https://towardsdatascience.com/the-wavelet-transform-e9cfa85d7b34
- Hands-On Machine Learning with Scikit-Learn and TensorFlow:- Aurélien Géron