IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Hand Gesture Recognition Using Machine Learning

Ch. Rajani

Department of Computer Science, SCSVMV, Tamil Nadu, India.

Submitted to R.Radhika(Asst. Professor)

Abstract:

Hand sign recognition is on of the challenges in the present world from small scaled industries to medium scale industries. present the hand gestures are using in human computer interaction system to develop a human level robots, medical, corporate companies, restaurants in some countries, In this paper I am also showing the difference from other gesture recognition system, here I am mainly working to recognise 10 different type of hand gesture system to understand computer using machine learning frame works, developed algorithm works to detect the different gesture in real time. Human computer interaction Hand sign (Gesture) provides a way to understand human body language Enables humans to interface with machine and interact naturally Without any mechanical devices Goal is to interpret hand sign recognition using deep learning algorithms.

Keywords:

Hand gestures, computer vision, image processing, classification, neural networks, deep neural networks, convolutional neural networks, max-pooling, feature extraction, classification metrics.

Introduction:

the main objective of the project to build hand gesture recognition system in real time, to detect the posture of hand gesture and predict the posture sign , and how to we can able to understand the computer human body language here we are working with different tools to detect the different feature from the different images.

First every image in the databased is converted into gray scale and extract features from different images to classify the image.

In the pre-processing stage some operations are applied to extract the hand gesture from its background and prepare the hand gesture image for the feature extraction stage. In the rst method, the hand contour is used as a feature which treats scaling and translation problems (in some cases). The complex moments algorithm is, however

Goal of hand gesture recognition to classify the given hand gesture data which is separated by some important features into some predefined number of hand gesture classes.

The objective is to explore two feature extraction methods: hand contour, and complex moments to solve the hand gesture recognition.

Type of hand gestures:

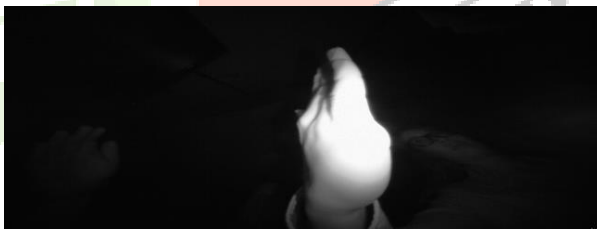
Palm First

L First Moved

Thumb

Index

Ok



Palm Moved

C

Down

Current researches in these hand gestures are limited to the use of glove-based, non-skin colour background and lighten areas with their associated problems. In this research we have shown that the use of feature extraction

method together with a convolution neural networks, max pooling, flatten layers is able to recognize a limited number of gestures accurately correct. We have also shown that the recognition system in real-time.

1. This study develops a real time hand gesture recognition system that can be used for different applications which involves a human computer interaction, robotics.

2. This study investigates the suitability of two different feature extraction approaches to solve the hand gesture recognition problem by identifying the primary advantages and disadvantages of each method. Facing the hand Infront of webcam , it will redirect in the contours to detect the frame in the real time and each frame is named as images using OpenCV , the images can be extracted the image which is extracted can bee go through our trained model to detect the feature extractions and classify the image.

3. method can be overcomes the challenges other previous methods could not handle, namely, working under different conditions such as scaling, translation and rotation. Hand contour method, however, does not handle rotation cases which limits its use in hand gesture-based applications.

4.Convolutional Neural network classifier, which is used as a recognizer for hand gesture images based on the features extracted by the two methods, is also evaluated in terms of accuracy, convergence speed, overfitting and under-fitting. Feature Extraction Technique for Static Hand Gesture Recognition.

Data set information:

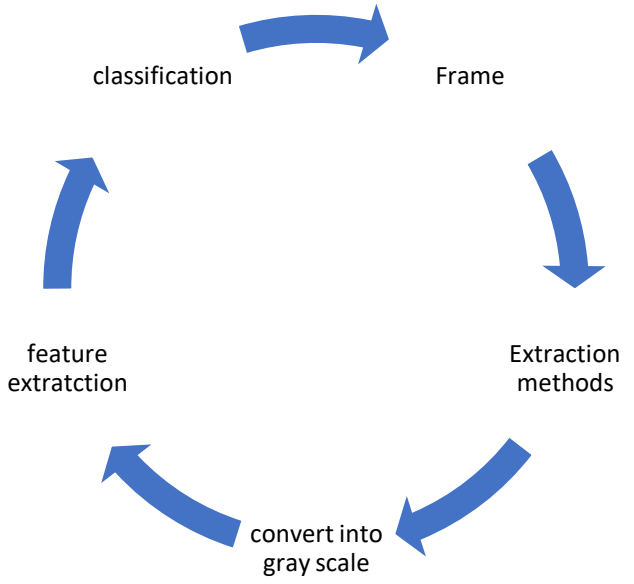
The project used the dataset with 10 different types of gesture available on Kaggle. It contains 40000 images with different hands and hand gestures. There is a total of 10 hand gestures of 10 different people presented in the dataset. There are 5 female subjects and 5 male subjects.

Extraction methods and converting image to grey scale image:

First we need to load images and checked if it's everything we expected, we have to prepare the images to train the algorithm. We must load all the images into an array that we will call X and all the labels into another array called Y.

Next Converting images into grey scale images, to identify the hand gestures clearly, which can be converted the image into colour RGB using OpenCV library.

Hand Gesture Recognition system steps:



Creating model:

To detect the gesture shape

To simplify the idea of the model being constructed here, we're going to use the concept of Linear Regression. By using linear regression, we can create a simple model and represent it using the equation ```y = ax + b```.

```a``` and ```b``` (slope and intercept, respectively) are the parameters that we're trying to find. By finding the best parameters, for any given value of x, we can predict y. This is the same idea here, but much more complex, with the use of Convolutional Neural Networks.

A Convolutional Neural Network (CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases), and the image shape in 3dimensions or 2d- dimension, to various aspects in the image and be able to differentiate one from the other. The pre-processing required in a CNN layers is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, CNN Layers have the ability to learn these filters/characteristics.

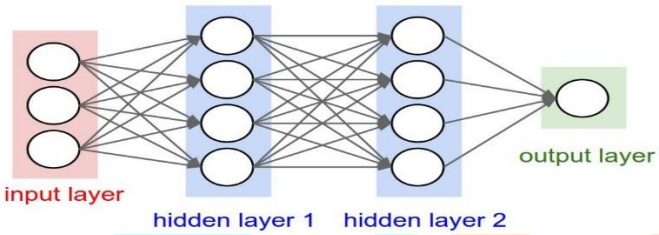


Figure 1 - Example of Convolutional Neural Network.

From Figure 1 and imagining the Linear Regression model equation that we talked about, we can imagine that the input layer is x and the output layer is y. The hidden layers vary from model to model, but they are used to "learn" the parameters for our model. Each one has a different function, but they work towards getting the best "slope and intercept".

Overview:

- Import what the need
- Creation of CNN
- Compiling and training model
- Saving model for later use

Convolutional neural networks (CNNs) are the current state-of-the-art model architecture for image classification tasks. CNNs apply a series of filters to the raw pixel data of an image to extract and learn higher-level features, which the model can then use for classification. CNNs contains three components:

Convolutional layers, which apply a specified number of convolution filters to the image. For each subregion, the layer performs a set of mathematical operations to produce a single value in the output feature map. Convolutional layers then typically apply a Padding, Stride, ReLU activation function to the output to introduce nonlinearities into the model.

Pooling layers, which down sample the image data extracted by the convolutional layers to reduce the dimensionality of the feature map in order to decrease processing time. A commonly used pooling algorithm is max pooling, which extracts subregions of the feature map (e.g., 2x2-pixel tiles), keeps their maximum value, and discards all other values.
```

Dense (fully connected) layers, which perform classification on the features extracted by the convolutional layers and downsampled by the pooling layers. In a dense layer, every node in the layer is connected to every node in the preceding layer.

Our Convolutional Neural Network consists of different layers that have different functions. As explained before, the Conv2D layer performs a 2-D convolutional operation, which can be basically interpreted as a mathematical operation to calculate weights inside the image. In order to maximize the network's performance, we need to understand the parameters required by them.

The first required by the Conv2D layer is the number of filters that the convolutional layer will learn. Layers early in the network architecture (closer to the actual input image) learn fewer convolutional filters while layers deeper in the network (closer to the output predictions) will learn more filters. This permits information to flow through the network without loss. These filters emulate edge detectors, blob detectors and other feature extraction methods. It is necessary to tune the values of the filters, but it is common practice to use powers of 2, starting with 32, 64, 128 and increasing to 256, 512, 1024, for example.

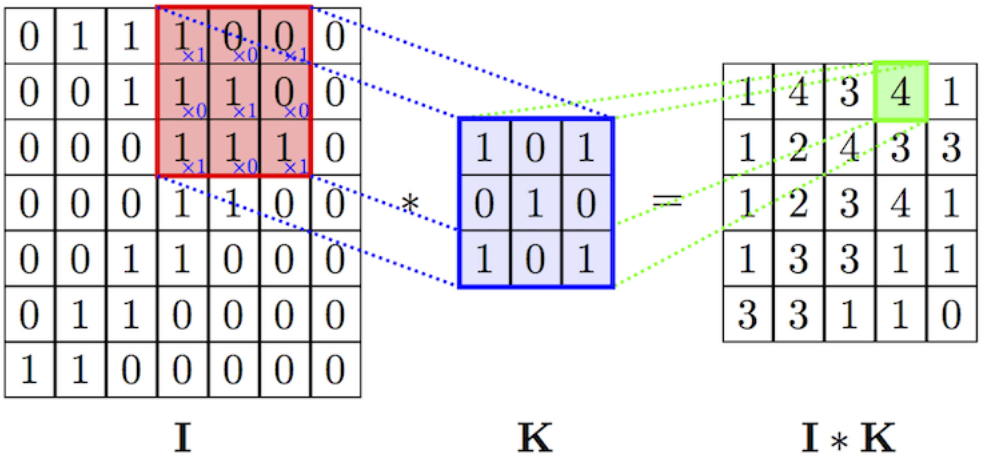


Figure 2 - Example of 2D convolution operation.

Another parameter required by the Conv2D layer is the kernel_size, a 2-tuple specifying the width and height of the 2D convolution window. The kernel_size must be an odd integer, with typical values of (1, 1) , (3, 3) , (5, 5) , (7, 7) . It's rare to see kernel sizes larger than 7×7. If the input images are greater than 128×128 it is recommended to test a kernel size > 3 to help learn larger spatial filters and to help reduce volume size.

Then, MaxPooling2D is used to reduce the spatial dimensions of the output volume. It reduces processing time and allows assumptions to be made about features contained in the sub-regions binned. It is possible to notice in this network that our output spatial volume is decreasing and our number of filters learned is increasing. This is a common practice in designing CNN architectures.

Finally, ReLU stands for rectified linear unit, and is a type of activation function. ReLU is the most commonly used activation function in neural networks, especially in CNNs. ReLU is linear (identity) for all positive values, and zero for all negative values. This means that it's cheap to compute as there is no complicated math. The model can therefore take less time to train or run. Also, it converges faster by applying non-linearities to the model, so there is no 'vanishing gradient problem' suffered by other activation functions like sigmoid or tanh.

In the end, there is a lot of trial and error to get the best parameters and network architecture. These are some common practices that help reach the best result faster.

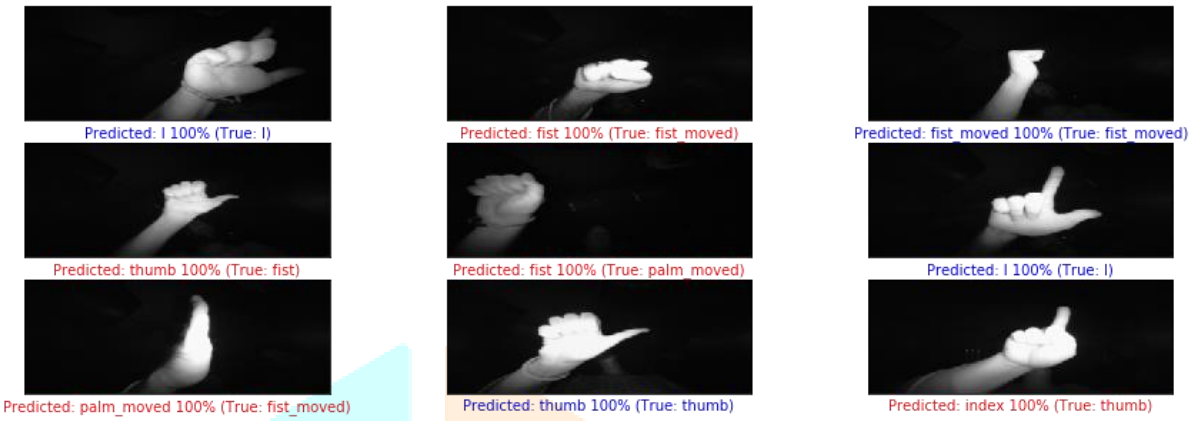
Testing Model¶

Now that we have the model compiled and trained, we need to check if it's good. First, we run model.evaluate to test the accuracy. Then, we make predictions and plot the images as long with the predicted labels and true labels to check everything.

Overview:

- Evaluate model
- Predictions
- Plot images with predictions
- Visualize model

Testing Results:



Model Architecture:

 # Construction of model

```
# Construction of model
model = Sequential()
model.add(Conv2D(32, (5, 5), activation='relu', input_shape=(120, 320, 1)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(10, activation='softmax'))
```

Drawbacks:

The drawbacks of some discussed methods are explained:

Convolutional Neural Network classifier has been applied for gestures classification but it is time consuming and when the number of training data increase, the time needed for classification are increased too. the NN required several hours for learning 42 characters and four days to learn ten words.

In the system is variation to environment lighting changes which produces erroneous segmentation of the hand region. HMM tools are perfect for recognition dynamic gestures, but it is computational consuming.

My model can detect 60% accurate results buy using CNN, Maxpooling, Fully connected layers, dense layers.

If increase model accuracy we need hight level of gpu system and we need to add more layers to get more accuracy.

Summary:

Here I have share my total model performance using confusion matrix from the below table you can observe how the model is performing for the different hand gestures.

	Predicted Thumb Down	Predicted Palm (H)	Predicted L	Predicted Fist (H)	Predicted Fist (V)	Predicted Thumbs up	Predicted Index	Predicted OK	Predicted Palm (V)	Predicted C
Actual Thumb Down	637	65	57	61	59	65	65	61	76	53
Actual Palm (H)	58	677	56	49	65	67	62	46	55	58
Actual L	69	59	700	54	65	65	52	49	51	59
Actual Fist (H)	63	68	54	693	63	67	69	61	57	47
Actual Fist (V)	61	59	63	61	636	58	54	60	64	69
Actual Thumbs up	64	57	67	67	64	677	61	53	71	62
Actual Index	63	50	55	65	59	54	668	64	49	55
Actual OK	63	55	61	66	57	54	60	628	62	68
Actual Palm (V)	66	65	55	75	58	59	64	52	650	50
Actual C	55	46	60	64	67	59	58	58	62	636

Conclusion:

Based on the results presented in the previous section, we can conclude that our algorithm successfully classifies different hand gestures images with enough confidence (>55%) based on a Deep Learning model.

References:

1. https://en.wikipedia.org/wiki/Gesture_recognition
2. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_table_of_contents_feature2d/py_table_of_contents_feature2d.html
3. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
4. https://missinglink.ai/guides/neural-network-concepts/backpropagation-neural-networks-process-examples-code-minus-math/
5. https://www.researchgate.net/publication/284626785_Hand_Gesture_Recognition_A_Literature_Review
6. https://towardsdatascience.com/image-pre-processing-c1aec0be3edf
7. https://www.deeplearning.ai/deep-learning-specialization/