Handwritten Digits Recognition using HOG features and SVM

Imaan Saleem bcsf20m523
Tuba Mushtaq bcsf20m550

Faculty of Computing and Information Technology

PUCIT
Punjab Univeristy
College of Information Technology

Introduction:

- Handwriting recognition is the ability of a computer to receive and interpret intelligible handwritten input from sources such as paper documents, photographs, touch-screens, and other devices. There are many techniques to that have been developed to recognize the handwriting. One of them is Optical character recognition.
- Algorithms: Machine learning establishes classification methods are K-Nearest Neighbor (KNN), Decision Tree (DT), and Neural Networks (NN). In this experiment, we will discuss the handwritten digits classification using Histogram of Oriented Gradient (HOG) features and a Support Vector Machine (SVM) analyzed on MNIST dataset.

HOG and SVM

- Histogram of Oriented Gradients (HOG):The histogram of oriented gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection.
- Support vector machines (SVM):Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection.

MINIST dataset:

- MNIST is a widely used dataset for the hand-written digit classification task. It consists of 70,000 labeled 28x28 pixel grayscale images of hand-written digits. MNIST is one of the most famous and popular used database for handwritten digits recognition which contains 70,000 samples included two parts of 60,000 and 10,000 samples corresponding to training and test data.From the total numbers of instances, Some of the samples are shown in figure 1

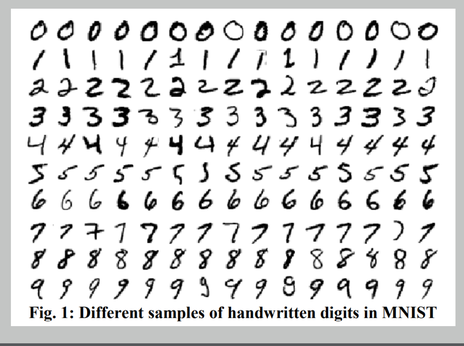

Fig. 1: Different samples of handwritten digits in MNIST

Procedure

- There are 10 classes corresponding to the handwritten digits from '0' to '9' which are very depend on the handwritten. The main difficulty in the handwritten digit's recognition is different handwritten style which is a very personal behaviour where there are a lot of models for numbers based on the angles, length of the segments, stress on some parts of numbers. Figure 1 shows 15 different handwritten digits related to these issues. However, recognizing numbers is clear for human but it is not very easy for machines especially when there are some ambiguities on different classes (e.g., '1' and '7').
- When collecting a data set we will split it into two parts: one for training and the other for testing.

Training and Testing

- There are different methods for splitting the dataset, the most common following the Pareto ratio of 80:20 or 70:30.
- Training data is a set of samples (such as a collection of photos or a set of texts) with assigned relevant and comprehensive labels (classes or tags) used to fit the parameters (weights) of a machine learning model with the goal of training it.We will use 80% of data for training.
- After we have trained the machine learning model, we will use the 20% of saved from the training data set to test it.

Description on whole model:

- Our model works in three steps:
  1) Preprocessing,
  2) HOG features extraction and
  3) Support vector machines classification.
- In the preprocessing, we have some basic image processing to separate numbers from real samples or preparing data from dataset (which is reshaped from images to the vectors)
- In the second part, we extract HOG features which is very distinguishable descriptor for digits recognition where we divide an input image into 9×9 cells and compute then the histogram of gradient orientations thereby we represent each digit with a vector of 81 features.
- In the third stage, a linear multiclass support vector machine has been employed to classify digits.
- In general, the main contribution of our model is employing HOG features with SVM. HOG can perform distinguishable features. And SVM can be useful to classify HOG features.

Overall Structure

- The overall view of proposed approach has been illustrated in Figure 2.

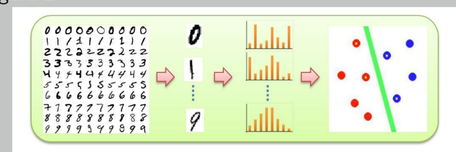

Fig. 2: The overall structure of our model, we first separate digits and then extract HOG features, finally, features are classified by multiclass SVM classification

- The total numbers of instances are shown in Table 1 ordered by classes.

Table 1: MNIST Handwritten Digits test and train info

Class	Number of samples		
	Train Data	Test Data	Total
'0'	5923	980	6903
'1'	6742	1135	7877
'2'	5958	1032	6990
'3'	6131	1010	7141
'4'	5842	982	6824
'5'	5421	892	6313
'6'	5918	958	6876
'7'	6265	1028	7293
'8'	5851	974	6825
'9'	5949	1009	6958
All	60,000	10,000	70,000

- Applying testing digit to the trained model

Conclusion

- In general, proposed approach used HOG which is a very efficient appearance-based descriptor with linear multiclass SVM classification for handwritten digits recognition. We have successfully done digit recognition using this method. The most important benefit of above structure is that our model is fast and useful for real-time applications.

Matlab Code

- Training Code

```
clc;
clear all;
close all;
warning off;
imds=imageDatastore('database','IncludeSubFolders',true,
'LabelSource','foldernames');
trainingFeatures=[];
trainingLabels=imds.Labels;
for i = 1:numel(imds.Files)
img = readimage(imds,i);
trainingFeatures(i,:) = extractHOGFeatures(img,'CellSize'
,[8 8]);
end
Classifier =fitcecoc(trainingFeatures,trainingLabels);
save Classifier Classifier
```

- Test Code

```
clc;
clear all;
close all;
warning off;
load Classifier;
[filename, pathname] = uigetfile('*.*', 'Pick an Image');
filename=strcat(pathname,filename);
imga=imread(filename);
img=imresize(imga,[45 24]);
[Features] = extractHOGFeatures(img,'CellSize',[8 8]);
PredictedClass=predict(Classifier,Features);
PredictedClass=char(PredictedClass);
figure,
imshow(imga),title(PredictedClass);
disp(PredictedClass);
```

References

- International Journal of Computer Applications (0975 – 8887) Volume 104 – No.9, October 2014
- https://medium.com/@basu369victor/handwritten-digits-recognition-d3d383431845
- https://labelyourdata.com/articles/machine-learning-and-training-data
- https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/