**42028 Deep Learning and Convolutional Neural Networks Assignment 1**

**Handwritten digit classification**

**Keyu Chen**

**13261021**

Table of Content

[Introduction 2](#_Toc37795237)

[Dataset 2](#_Toc37795238)

[Experimental results and discussion 2](#_Toc37795239)

[KNN 2](#_Toc37795240)

[Experimental settings 2](#_Toc37795241)

[Evaluation on test set 5](#_Toc37795242)

[SVM 5](#_Toc37795243)

[Experimental settings 5](#_Toc37795244)

[Evaluation on test set 6](#_Toc37795245)

[ANN 6](#_Toc37795246)

[Experimental settings and evaluation 6](#_Toc37795247)

[Experimental Results 7](#_Toc37795248)

[Confusion Matrix 7](#_Toc37795249)

[Comparative study 8](#_Toc37795250)

[Discussion 8](#_Toc37795251)

[Conclusion 8](#_Toc37795252)

# Introduction

Handwritten digit problem is a classic classification topic in the revolution of deep learning, it challenges computers’ capability to recognize and interpret handwritten digits of various styles of writing. The aim of the experiment is to implement 3 powerful machine learning techniques of K-Nearest Neighbors, Support Vector Machine and Artificial Neural Network on MNISIT dataset within Local Binary (LBP) Patterns and Histogram of Oriented Gradients (HOG) feature extractions. The report will also explain how the adjustment of parameters in each algorithm effects performance and compare the results using different feature extraction techniques. The comparative study of each algorithm’s experimental result based on Confusion Matrix will also be demonstrated.

# Dataset

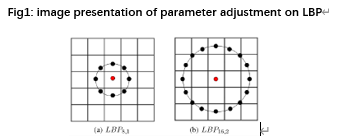
The dataset that we implement in this experiment is MNIST dataset which is developed by Yann LeCun. The dataset contains split training set and test set with 60000 examples and 10000 examples respectively with 784 pixels of each and two label values from 0 to 9. The digits have already been normalized and centered in a fixed-size image that it is handy for experimenting purposes and they are stored in 4 separate idx format which stores vectors and multidimensional matrices (LeCun et al., 1998a). Since both training set and test set contains handwritten digits of 50 percent students and 50 percent employees, the collected examples cover a range of unbiased writing styles.

# Experimental results and discussion

## KNN

### Experimental settings

#### Local Binary Pattern

LBP algorithm has parameters of numPoints P and radius R where P presents the number of points surrounding the central point to process binary computation and R presents radius from the central point to the neighboring cells that captures neighboring details in the spatial scale. There are different combinations of paring P and R, they are normally set to (P=8, R=1), (P=16, R=2), (P=24, R=3), and (P=32, R=4). The LBP value is then calculated for each block and stored into a new matrix, the values are calculated from pixel (2,2) to (27,27) in our reshaped image size of 28\*28. This process will create a matrix of size 26\*26 for model fitting.

4 combinations of P and R parameters have been performed with the training set in the experiment, the constructed histogram of training input using LBP algorithm with 4 different combinations of P and R are stored in 4 pairs of lists accordingly.

**Table1: different LBP parameters training results with KNN(k=3)**

|  |  |  |
| --- | --- | --- |
| **P (numPoints)** | **R (radius)** | **Training accuracy (k=3)** |
| 8 | 1 | 0.595 |
| 16 | 2 | 0.685 |
| 24 | 3 | 0.691 |
| 32 | 4 | 0.660 |

As table1 demonstrates, the best training accuracy obtained from 4 pairs of P and R is (P=24,R=3) with our default KNN classifier which is 0.691. Hence, parameters of LBP algorithm will be set to (P=24,R=3) in the following test evaluation.

#### KNN parameter setting on LBP feature extraction

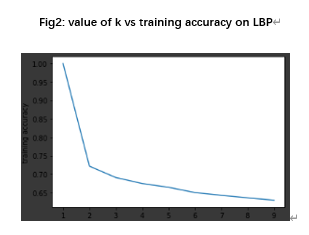
The idea of how KNN classifier works is that it searches for k numbers of observations in the input data that are closest to the measurements of the ground truth. Hence, it is essential for us to understand how the change of parameter k effects the training result and choose the appropriate value for k.

The experiment has performed KNN classifier on a range of k values from 1 to 9, training accuracy is calculated accordingly as shown below.

**Table2: training accuracy of LBP feature extraction with various k value**

|  |  |
| --- | --- |
| **Value of K** | **Training accuracy** |
| 1 | 1.0 |
| 2 | 0.7216 |
| 3 | 0.6908 |
| 4 | 0.6746 |
| 5 | 0.6643 |
| 6 | 0.65 |
| 7 | 0.6427 |
| 8 | 0.6358 |
| 9 | 0.6293 |

As presented in table 2, the highest accuracy obtained is when k=1 which is 100%. However, this result is unreliable as it tests on a single observation and gives wrong assumptions based on the calculation of distance, closeness and proximity between the two points when k=1, the algorithm will also give the same prediction as it has memorized the training data. As we will likely encounter problem of overfitting, the value of k is avoided to be set as 1 in the experiment.

Inversely, the reliability of the training result increases as the value of k increases, the predictions are more stable when majority voting is processed. As shown in the diagram below, despite the accuracy of 100% when k=1, the training accuracy gradually drops when the value of k increases. The best training accuracy obtained from KNN with LBP feature extraction is therefore determined as 0.72.

#### Median filtered Local Binary Pattern feature extraction

With the application of the robustness of Median Filtering technique, the performance of LBP can be improved. While median filtering technique can effectively reduce noise on an image, the more informative details can be extracted from Local Binary Pattern feature extraction. The training accuracy obtained from LBP feature extraction with median filtered images is 0.724 and it is the best training result with KNN classifier training on LBP feature extraction.

#### Histogram of Orientated Gradient

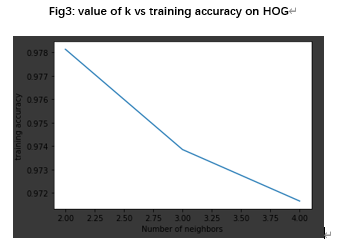
HOG is capable of detecting edge feature by extracting the gradient and orientation for localized portions of an image and generate histograms for each region. Parameters of *pixels\_per\_cell*, *cells\_per\_block* and *block\_norm* are adjusted. While cell size of [2,2] contains more shape information than size of [4,4] and [8,8], the training accuracy would be more stable intuitively, hence the cell size is set to [2,2] to obtain histograms. Two different parameter settings of cells\_per\_block is tested in the experiment with two different normalization method of ‘L2-hys’ and ‘L1-sqrt’.

**Table2: HOG parameter tuning with KNN(k=2)**

|  |  |  |
| --- | --- | --- |
| **Pixels\_per\_cell** | **Block\_norm** | **Training accuracy** |
| 10,10 | L2-Hys | 0.933 |
| 10,10 | L1-sqrt | 0.957 |
| 8,8 | L2-Hys | 0.978 |

As shown above, the best training result that we get is when *pixels\_per\_cell* set to [8,8], and block normalization method set to *L2-Hys.*

#### KNN parameter setting on HOG feature extraction

The value of k is tested from the range of 2 to 5 on extracted features. It can be evident that the training performance with HOG feature extraction is gradually dropping as the value of k increases.

As we can see from the figure above, the best training accuracy on HOG feature extraction is 0.978.

### Evaluation on test set

The evaluation on test is performed on the KNN classifiers with the best training accuracy for both LBP and HOG feature extraction.

**Table3: test results comparison**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature extraction technique** | **Parameter setting on feature extraction** | **Classifier parameter setting** | **Test result** |
| LBP | (p=24, R=3) | K = 2 | 0.453 |
| HOG | Orientations=9, pixels\_per\_cell=(8,8), cell\_size=(2,2), block\_norm=’L2\_Hys’ | K = 2 | 0.94 |

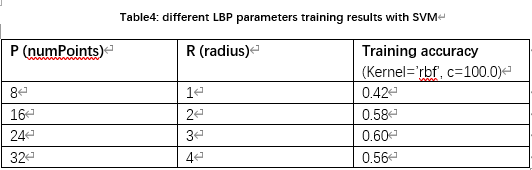
As presented above, HOG feature extraction technique outperforms LBP feature extraction technique with KNN classifier on MNIST dataset, the highest test accuracy obtained from this experiment is 0.94.

## SVM

### Experimental settings

#### Local Binary Pattern

The same procedure is done to compare the best parameter pair for LBP feature extraction on SVM model, the best result obtained from the 4 groups of P and R is (P=24,R=3), the training accuracy increases to a certain point where most valuable details are extracted from 24 neighboring points with radius of 3 from the central point. The feature extracted from LBP with (P=24,R=3) will be used for test evaluation.



#### Hyperparameter tuning

The experiment has adjusted the parameters of *C*, *Gamma* and *Kernel* to obtain the best training result. *Kernel* contains a function that takes two data points as input and returns a similarity score, the function of ‘rbf’ and ‘poly’ are experimented. The ‘rbf’ kernel allows us to build complex decision boundaries that helps us to define a hyperplane which separates our data neatly while the ‘poly’ kernel presents the similarity of datapoints in a feature space over polynomials of the original data points. The parameter C applies penalty on the error, it balances smoothing decision boundary and correctness of the classification. The parameter gamma represents the non-linearity of the feature plane, larger the value of gamma is, the more effect of individual support vector decreases. The SVM classifier has been trained with different parameter values on LBP feature extraction, the best performing set of parameters is when *C = 100, gamma = 100* and *kernel = ‘rbf’*, the highest training result is 0.657.

#### HOG feature extraction

The same procedure was done in the experiment to compare different parameter settings of HOG feature extraction with SVM classifier. The training accuracies obtained from different parameter settings are all approaching 1 in the experiment.

### Evaluation on test set

The test accuracy obtained from LBP and HOG feature extraction with optimal parameter settings are 0.615 and 0.971 respectively.

## ANN

### Experimental settings and evaluation

#### Raw pixel input

The ANN is trained and tested on raw pixel input with different hyperparameters. Different numbers of hidden layer, neurons in each layer and activation function are experimented. Each model is trained with 20 epochs, early stopping has been implemented to stop training once the training loss is below 0.05.

**Table5: ANN parameter tuning with raw input**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Number of hidden layers** | **Number of neurons** | **Activation function** | **Learning rate** | **Train accuracy** | **Test accuracy** |
| 1 | 128 | SGD | 0.1 | 0.987 | 0.9731 |
| 1 | 128 | SGD | 0.01 | 0.9635 | 0.958 |
| 1 | 256 | SGD | 0.1 | 0.9875 | 0.9764 |
| 2 | 128 each | SGD | 0.1 | 0.9874 | 0.9754 |
| 1 | 128 | Adam | 0.1 | 0.5432 | 0.5434 |
| 1 | 128 | Adam | 0.01 | 0.9848 | 0.9613 |
| 1 | 256 | Adam | 0.01 | 0.9858 | 0.9654 |
| 2 | 128 each | Adam | 0.01 | 0.9719 | 0.9644 |

As table above shows, with raw pixel input, ANN’s performance is quite good with minor change in parameters except for activation function of Adam with 0.1 learning rate. The reason behind this might be the large learning rate has caused overshooting and the function fails to find the local minima. In overall, both training and testing results are stable, SGD performs a bit better than Adam.

#### HOG feature extraction

Similarly, 4 different parameter settings are performed with ANN, parameters of *pixel\_per\_cell* and *block\_norm* are adjusted using, the shape of input data changes after the features are extracted. With *pixel\_per\_cell* set to (10,10), the shape of input becomes (40000,36) while the shape of input becomes (40000,36) when *pixcel\_per\_cell* is (8,8). The input neurons of 36 and 144 are initialized in our neural network respectively.

**Table6: HOG feature extraction parameter tuning on ANN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Pixels\_per\_cell** | **Block\_norm** | **Training accuracy** | **Test accuracy** |
| 10,10 | L2-Hys | 08827 | 0.8835 |
| 10,10 | L1-sqrt | 0.9024 | 0.9051 |
| 8,8 | L2-Hys | 0.9607 | 0.9551 |
| 8,8 | L1-sqrt | 0.9647 | 0.9374 |

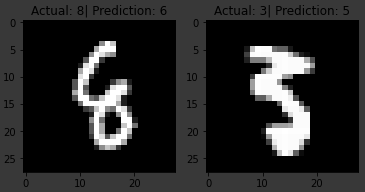
As demonstrated above, the test accuracy reaches the highest when *pixel\_per\_cell* is (8,8) with *block\_norm* set to ‘L2-Hys’.

Parameters on the neural network are also adjusted to compare losses and accuracies. It is evident in the experiment that adding another hidden layer with 128 neurons increases the model’s performance slightly, more details of the data are learned when the model becomes more complex.

# Experimental Results

## Confusion Matrix

The 3rd ANN model with raw pixel input shape of 28\*28, 1 hidden layer, 256 neurons in the hidden layer and SGD activation function with 0.1 learning rate achieves the highest accuracy out of 3 classifiers in the experiment. As confusion matrix shows in the notebook, the predictions on test labels of 5 and 8 are the lowest, the model captures some input images of 5/8 and returns predictions of 8/5. There are 236 wrong predictions out of 10000 test samples. **Fig4: wrong predictions visualization**

As presented below, the wrong predictions are reasonable as the handwritten digits are not normative and they do look like the predicted value.

## Comparative study

To compare the best performing result from 3 different classifiers, the highest accuracies obtained from the classifier using different features are presented below:

|  |  |  |
| --- | --- | --- |
| **Classifier/Results** | **Training result** | **Testing result** |
| KNN | 0.978 | 0.94 |
| SVM | Approaching 1 | 0.971 |
| ANN | 0.9875 | 0.9764 |

## Discussion

In overall, HOG feature extraction produces satisfying results on 3 different classifiers while LBP feature extraction works better with SVM than KNN. ANN has the best performance on both training and testing data with raw pixel input, while ANNs can learn complex patterns, having the input data’s feature extraction might reduce the model’s stability.

It is discovered that LBP feature extraction performs better on smooth image data than flat data. Since the variance of intensity within the region is very low on MNIST dataset, the image areas that contain valuable information are flat. In this case, LBP is not robust on extracting valuable features.

# Conclusion

The experiment has implemented 3 powerful machine learning techniques of KNN, SVM and ANN with different feature extraction techniques. It is essential to understand each model’s functionality and behaviors to produce the optimized results. The performance of the classifier and feature extraction techniques is strongly determined by the nature of the dataset. While LBP feature extraction is a powerful technique with image classifications, it is not the best choice with the nature of MNIST data.

The experiment has performed parameter tuning based on personal understandings of the model and its functionality instead of using powerful methods such as GridSearch, there’s still space of improvement.

# Reference

A.M. Nickfarjam, A. Pourshabanan Najafabadi & H. Ebrahimpour-komleh. 2014, ‘Multi0input 1-dimensional deep belief network: action and activity recognition as case study’, *Iranian Conference on Electrical Engineering*, vol.78, pp. 17739

Fraj. B. M, 2017 *In Depth: Parameter tuning for KNN,* Median, viewed 3 April 2020, <<https://medium.com/@mohtedibf/in-depth-parameter-tuning-for-knn-4c0de485baf6>>

Hafiane, A., Seetharaman, G & Zavidovique, B. 2007, ‘Median Binary Pattern for Textures Classification’ *Springer-Verlag Berlin Heidelberg*, vol.4633

Lecun, Y., Cortes, C & Burges, J.C. 1998, ‘The MNIST Database of Handwritten digits’, *Proceedings of the IEEE*, vol.86, no.11, pp.2278-2324

Jain. R, 2017 *Simple Tutorial on SVM and Parameter Tuning in Python and R*, Hackerearth, viewed 3 April 2020, <<https://www.hackerearth.com/blog/developers/simple-tutorial-svm-parameter-tuning-python-r/>>

Sharma, P. 2018 *Improving Neural Networks – Hyperparameter Tuning, Regularization, and More,* Analytics Vidhya, viewed 6 April 2020, <<https://www.analyticsvidhya.com/blog/2018/11/neural-networks-hyperparameter-tuning-regularization-deeplearning/>>

Signh, A. 2019, *Feature Engineering for images: A valuable Introduction to the HOG Feature Descriptor*, Analytics Vidhya, viewed 4 April 2020, <<https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/>>

Stewar. M, 2019 *Simple Guide to Hyperparameter Tuning in Neural Networks,* Towards Data Science, viewed 6 April 2020, <<https://towardsdatascience.com/simple-guide-to-hyperparameter-tuning-in-neural-networks-3fe03dad8594>>