INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 6.078
*(Volume 6, Issue 3)*
Available online at: www.ijariit.com

# Handwritten digit classification using Convolutional Neural Networks

*Shivam Srivastava*
*percyj456@gmail.com*
*Babu Banarasi Das National Institute of Technology and Management, Lucknow, Uttar Pradesh*

*Richa Sharma*
*richa.sharma.ar@gmail.com*
*Babu Banarasi Das National Institute of Technology and Management, Lucknow, Uttar Pradesh*

*Pratyaksha Jindal*
*pratyakshajindal@gmail.com*
*Babu Banarasi Das National Institute of Technology and Management, Lucknow, Uttar Pradesh*

*Sikandar Singh Sandhu*
*sikandersngh199@gmail.com*
*Babu Banarasi Das National Institute of Technology and Management, Lucknow, Uttar Pradesh*

*Pratyush*
*pratyushtanwar1997@gmail.com*
*Babu Banarasi Das National Institute of Technology and Management, Lucknow, Uttar Pradesh*

*Atul Kumar*
*atul.003dx@gmail.com*
*Babu Banarasi Das National Institute of Technology and Management, Lucknow, Uttar Pradesh*

## ABSTRACT

***This research study throws light on one of the most common use-case of Handwritten Digit recognition which can be seen being implemented by using a particular Deep Learning technique for pattern recognition known as Convolutional Neural Networks which works similar to the functionality of neurons in a human brain. We have trained and tested the MNIST dataset using this technique and implemented a classifier to predict the pattern of the digits.***

***Keywords***— *Handwritten Digit Classifier, MNIST, Deep Learning, Convolutional Neural Networks, Machine Learning, PyTorch, Neural Networks*

## 1. INTRODUCTION

Machine learning is a sub-domain of artificial intelligence. The aim of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people.

As we all know machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computation approach, algorithms are set of explicitly programmed instructions used by computers to calculate or solve a problem. Machine learning algorithms instead allow for computers to train on data inputs along with its pattern and use statistical analysis techniques in order to output values that fall within a specific range. Because of this, machine learning gives computer a wider scope for building models from sample data in order to automate decision-making processes based on data inputs.

Machine learning is an emerging technology which enables computer to learn automatically from past data. Machine learning uses a number of algorithms for building mathematical models and making predictions using historical data or information. Currently, it is being implemented for various use-cases such as image recognition, speech recognition, email filtering, recommender system, and many more.

### 1.1 Importance of Deep Learning

To make machines more intelligent, the developers are diving into machine learning and deep learning techniques. A human learns to perform a task by practicing and repeating it again and again so that it memorizes how to perform the tasks. Then the neurons present in the human brain gets automatically triggered and they can quickly perform the task they have learned. Deep learning is also very similar to this. It uses different types of neural network architectures for different types of problems. For example, object recognition, image and sound classification, object detection, image segmentation, etc.

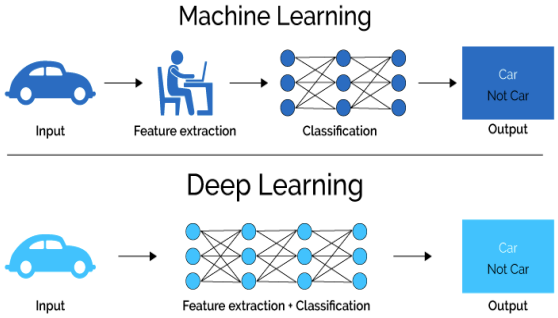


**Fig. 1: Machine Learning V/s Deep Learning**

### 1.2 What is Handwritten Digit Recognition?

The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes. The handwritten digit recognition system is a way to tackle this problem which uses the image of a digit and recognizes the digit present in the image.

 

In this study, we are going to implement a handwritten digit recognition app using the MNIST dataset. We will be using a special type of deep neural network that is Convolutional Neural Networks**.** In the end, we will be implementing an Android Application to serve the UI purpose of our trained classifier in which you can draw the digit and recognize it straight away.

MNIST Dataset**-:** The MNIST dataset contains handwritten digits which have a training set of 60,000 examples and a test set of 10,000 examples. It is a subgroup of a larger group available from NIST. The digits present in the dataset are already size-normalized and centred in an image having fixed-size. It is a recommended dataset for people who want to know about different pattern recognition techniques for real-world data while not caring about cleaning or pre-processing of the data.

### 1.3 PyTorch and its features

We have used PyTorch to implement our machine learning classifier which is an unique aspect of our project as there are very less implementations of deep learning technique through PyTorch present on the web so this gives us a fairer chance to opt for PyTorch instead of TensorFlow to implement our classifier.

PyTorch is a library based on python which is built to provide flexibility in the deep learning development environment. The workflow of PyTorch is similar to python's scientific computing library – numpy. Some advantages of PyTorch are listed below:

- Easy to use API: It is simple just like python.
- Python support: PyTorch smoothly integrates with the python data science stack. It is so similar to numpy that it is quite tough to notice the difference between the two.
- Dynamic computation graphs**:** Instead of pre-defined graphs which are of specific functionalities, PyTorch facilitates a framework for us to build computational graphs as we go, and which can be changed during runtime. This is valuable for the situations where we are unaware of how much memory is going to be required for implementing a neural network.

A few other advantages of using PyTorch are it's multi GPU support, custom data loaders and simplified pre-processors.

**Data Visualization on the dataset:** The MNIST dataset contains 60,000 different 28*28 images of hand written numerical characters from 0–9. This dataset is widely used for learning the basic of computer vision and image processing. After Reading the data in csv format using pandas, it will look as follows.

```
   label  pixel0  pixel1  pixel2  pixel3  pixel4  pixel5  pixel6  pixel7  \
0      1       0       0       0       0       0       0       0       0
1      0       0       0       0       0       0       0       0       0
2      1       0       0       0       0       0       0       0       0
3      4       0       0       0       0       0       0       0       0
4      0       0       0       0       0       0       0       0       0

   pixel8    ...     pixel774  pixel775  pixel776  pixel777  pixel778  \
0       0    ...            0         0         0         0         0
1       0    ...            0         0         0         0         0
2       0    ...            0         0         0         0         0
3       0    ...            0         0         0         0         0
4       0    ...            0         0         0         0         0

   pixel779  pixel780  pixel781  pixel782  pixel783
0         0         0         0         0         0
1         0         0         0         0         0
2         0         0         0         0         0
3         0         0         0         0         0
4         0         0         0         0         0

[5 rows x 785 columns]
```

**Fig. 2: Reading Data in csv format**

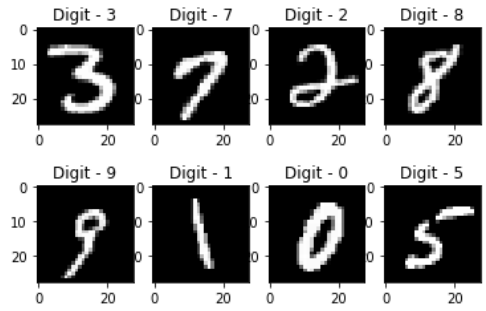


**Fig. 3: Sample digits from MNIST dataset**

PCA**:** Principal component analysis is one of the most fundamental and simple dimensionality reduction technique. It is also used as a clustering technique in case of un-supervised learning. It converts D-dimensional data to D`-dimensional data where D`≤D. For visualization using pair plot**, D`=2.** We`ll see how it works in 2d.

Suppose we have a dataset, with xi`s as the data points, as shown on the left and we want to project it on 1 dimension. It can be clearly seen that the spread or variance on the y axis in much more that the variance on x axis. Variance is a measure of information**.** So, if we project all the data points on the y-axis, we will conserve more variance as compared to the projection of data points on to the x axis. Which means we will be able to conserve more information which reducing its dimension from 2d to 1d.

Alternatively, preserving more information can be thought of as how easy is it to distinguish points in 1d. This can only be done if points are located in a wider spread and distinctly in 1d, which in turn means that the points should have a higher variance on the projected axis. Hence, we always try to preserve maximum variance possible. So, in this case, to visualize these points in 1-d we only need to project them on y axis and drop the x axis. Now for the case depicted in the above data points, we have our data set which is feature standardized. Standardization of the feature vector means to remove its mean (make it zero) and make it variance equal to unity. So the features are standardized independently.

$$x_{new} = \frac{x - \mu}{\sigma}$$

This is how standardization is done (here mu is mean and sigma is standard deviation).

In this case, the variance of the projected points on both of the axes will be the same as both axes have equal amount variance in 2d. In such case, which we typically see for this type of data, we will do something which is known as rotation of the axes. We will rotate the axes as:

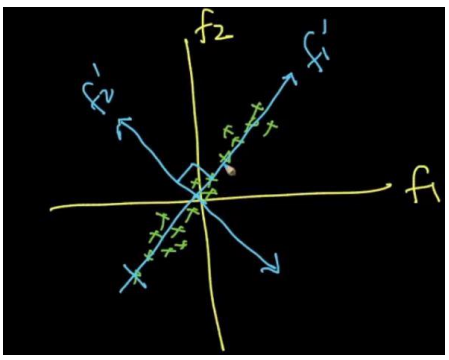


**Fig. 4: PCA implementation**

This is an optimizing problem given that u1 is a unit vector. It turns that the solution to our optimization problem is given by

 

eigen vectors of the co variance matrix of the data points or data matrix.

A co variance matrix is a matrix such that its value at $i^{th}$ row and $j^{th}$ column given the co variance between feature $i^{th}$ and feature $j^{th}$ of our data matrix(which has to be feature standardized)Eigen vectors are vector corresponding to a transformation matrix such that after the transformation they get scaled by a value know as its eigen value. All we need to do now is to take eigen vectors corresponding to the two largest eigen values (in case of reducing the dimension to 2) and project our data matrix onto these two vectors. Then our xi` s will become something like.

The visualization that we will get from PCA will be something like this.

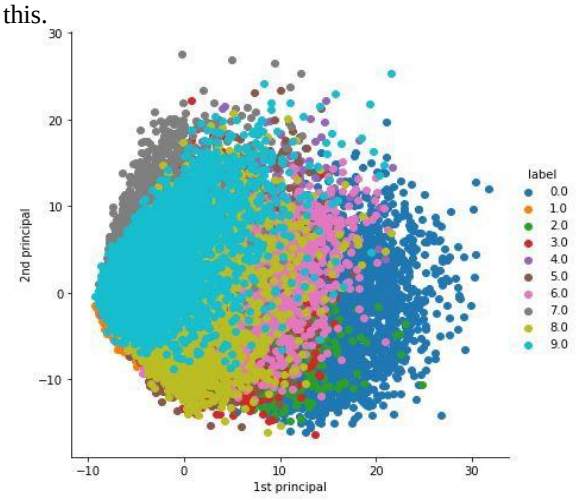


**Fig. 5: Visualization using PCA**

This might not be the best visualization but you can clearly see that it has clustered all the 0 s on one side and 1s and 9s on the other. For other number there is over huge lapping.

### 1.4 Algorithms taken into account

(a) KNN (K Nearest-Neighbours)**:** KNN is the non-parametric method or classifier used for classification as well as regression problems. This is the lazy or late learning classification algorithm where all of the computations are derived until the last stage of classification, as well as this, is the instance-based learning algorithms where the approximation takes place locally. Being the simplest and easiest to implement there is no explicit training phase initially and the algorithm does not perform any generalization of training data.

(b) SVM (Support Vector Machine): SVM falls into the category of supervised learning, and with the bonus of classification as well as regression problems. Generally, SVM draws an optimal hyperplane which classifies into different categories. In two dimensional space, To start with we plot the data points of the independent variable corresponding to the dependent variables. Then, begin the classification process from analysing the hyperplane or any linear or nonlinear plane differentiated the two-class at its best.

(c) NN (Neural networks): Neural Networks mimics the working of how our brain works. They have emerged a lot in the era of advancements in computational power. Deep learning is the application area for Neural Networks, the network connected with multilayers. The layers are composited from nodes. A node is just a perception which takes an input performs some computation and then passed through a node's activation function, to show that up to what context signal progress proceeds through the network to perform classification.

(d) CNN (Convolutional Neural Network): CNN has become famous among recent times. CNN is a part of deep, feed-forward artificial neural networks that can perform a various number of tasks with even better time and accuracy as compared to some other classifiers, in different applications regarding pattern analysis and recognition. Use of CNN has become wide in scope as Facebook uses neural nets for their automatic tagging algorithms, Google for photo search, Amazon for their product recommendations, Pinterest for their home feed personalization and Instagram for its search infrastructure within its ecosystem. Image classification or object recognition is a problem in passing an image as a parameter and predicting whether a condition is satisfied or not or the probability or most satisfying condition for an image. We are able to quickly recognize patterns, generalize from previous information and knowledge.

### 1.5 Why we opted CNN?

(CNN) has progressed rapidly, however, the real-world deployment of these models is often limited by computing resources and memory constraints. Another reason why CNN is hugely popular is because of its architecture- the best thing is there is no need for feature extraction. The system learns to do feature extraction and the core concept of CNN is, it uses convolution of image and filters to generate changeless features which are passed on to the next layer as an input. The features in next layer are convoluted with different filters to generate more changeless and abstract features and the process gets repeated till one gets final feature/output (let say the face of X) which is invariant to occlusions.

**1.5.1 Architecture of CNN:** CNN architecture is very much similar to the organization and functionality of the visual cortex and designed to mimic the connectivity pattern of neurons within the human brain. The neurons within a CNN architecture are split into a three-dimensional structure, with each set of neurons analysing a small region or feature of the image. In other words, each group of neurons specializes in identifying one part of the image. CNN's use the predictions from the layers to produce a final output that presents a vector of probability scores to represent the likelihood that a specific feature belongs to a certain class.

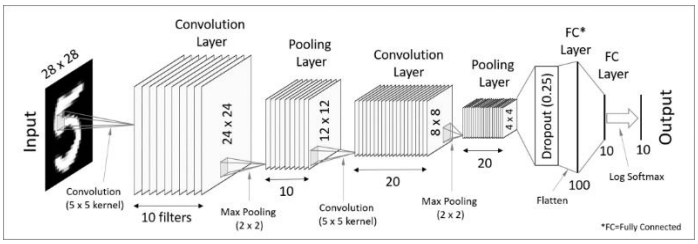


**Fig. 6: CNN-Architecture**

Algorithm used for the classifier is:

(a) Break the image into small image tiles. Similar to sliding window, we can pass sliding window over the entire large image and each result is saved as separate, as a segment of the large image as tiny picture tile.

(b) Feeding each tiny tile into the smaller size neural network, we rarely initialize the parameters with the same values and if not so, then we mark that tile as interesting.

(c) Store the results from each small tile into a new array, we would not like to misplace the index of the original file. So the results are placed in a grid having the same arrangement as an original image.

(d) Down-sampling, to reduce the size of a newer array, down-sampling is used by max-pooling.

 

**1.5.2 Layers of Convolution Neural Network:** The multiple occurring of these layers shows how deep our network is, and this formation is known as the deep neural network.

(a) Input: raw pixel values are provided as input.
(b) Convolutional layer: Input layers translates the results of neuron layer. There is a need to specify the filter to be used. Each filter can only be a 5*5 window that will slide over input data and get pixels with maximum intensities.
(c) Rectified linear unit [ReLU] layer: provided activation function on the data taken as an image. In the case of backpropagation, ReLU function is used which prevents the values of pixels form changing.
(d) Pooling layer: Performs a down-sampling operation in volume along the dimensions (width, height).
(e) Fully connected layer: score class is focused, and a maximum score of the input digits is found

## 2. RESULT

Applying the classifiers to real scenario problems for research purpose. Accuracy and speed of recognition are considered as the better measure of performance. Talking about the different classifiers one by one now.

### 2.1 Overall comparison results

To show several quantitative features like accuracy, time, specificity, sensitivity and other parameters comparison among different classifiers used in the training set.

| Parameters | KNN | SVM | NN | CNN |
|---|---|---|---|---|
| Accuracy (Test images) | 96.67 | 97.91 | 3.15 (Error) | 98.72 |
| Accuracy (Training images) | 97.88 | 99.91 | 2.17 (Error) | 99.78 |
| Time (Training) approx. | 25 min. | 19 min. | 35 min. | 70-90 min. |
| Time (Testing) approx. | 15 min. | 11 min. | 20 min. | 40 min. |

Classifier comparison on training and test data

Since our model is powered by PyTorch, which makes our model different from others which are present on the world wide web out of which most of them are powered by TensorFlow or Keras. Below is a brief comparison between PyTorch, TensorFlow & Keras to get a better glimpse of their functionalities and key features.

| S. No. | Comparison Factors | PyTorch | TensorFlow | Keras |
|---|---|---|---|---|
| 1 | Introduction | PyTorch is an open source **machine learning** library for Python, based on Torch. It is used for applications such as **natural language processing** and was developed by Facebook's AI research group. | TensorFlow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library that is used for **machine learning** applications like neural networks. | Keras is an open source **neural network** library written in Python. It is capable of running on top of TensorFlow. It is designed to enable fast experimentation with **deep neural networks**. |
| 2 | Level of API | PyTorch, on the other hand, is a **lower-level API** focused on direct work with array expressions. It has gained immense interest in the last year, becoming a preferred solution for academic research, and applications of deep learning requiring optimizing custom expressions. | TensorFlow is a framework that provides both **high and low level** APIs. | Keras is a **high-level API** capable of running on top of TensorFlow, CNTK and Theano. It has gained favour for its ease of use and syntactic simplicity, facilitating fast development. |
| 3 | Speed | PyTorch provide a similar pace which is fast and suitable for **high performance**. | TensorFlow provide a similar pace which is fast and suitable for **high performance**. | The performance is comparatively **slower** in **Keras** |
| 4 | Architecture | PyTorch has a **complex** architecture and the readability is less when compared to Keras. | Tensorflow on the other hand is not very easy to use even though it provides Keras as a framework that makes work easier. | **Keras** has a **simple** architecture. It is more readable and concise. |
| 5 | Debugging | **Pytorch** on the other hand has **better debugging capabilities** as compared to the other two. | But in case of Tensorflow, it is quite **difficult** to perform debugging. | In keras, there is usually very **less frequent** need to debug simple networks. |
| 6 | Datasets | PyTorch is used for **high performance** models and **large datasets** that require fast execution. | TensorFlow is also used for **high performance** models and **large datasets** that require fast execution. | Keras is usually used for **small datasets** as it is comparitively slower. |

### 2.2 Technological Stack for Application

The application has been developed with a wide set of tools to begin with. These tools have been handy in establishing the useful modules which, in turn have helped the whole team in the development phase of this application. The list of tools used in the implementation process of this application can be listed down as follows:

- Programming Language: The programming languages used in the creation of this application are Python 3, Kotlin & XML. With its massive support of open-sourced libraries and user-friendly command structure, Python 3 is perhaps one of the best platforms upon which the creation of newer software can be made possible. Our Machine Learning Model is trained on PyTorch framework which is also written in Python therefore, it made it easy to implement CNN architecture in our classifier.
- IDE: The IDE responsible for assisting in the creation process is Anaconda Prompt. The flexibility provided by the Anaconda prompt through its cloud infrastructure with most web browsers made this IDE the perfect tool for the assistance in the development phase of this application.
- Android Application Integration-: This has been done with the help of Android Studio. The Android Studio provides a list of useful tools and libraries which help the creator in deploying newer 6 methods, plus the creativity of the developer can also be assisted with the different modules present in the package. E.
- Testing Tool-: Deep learning technique has been inculcated in this project with the help of Pattern recognition & analysis. It has helped in the implementation of advanced filtering options that have enabled the application to deliver the suitable results to the user.

## 3. TOOLS USED IN WRITING RESEARCH STUDY

The research paper has been written with the help of various tools which include writing tools and as well as some research tools. Writing tools include MS Word 2013 which is a robust tool for text formatting and aligning. Research tools include Google scholars, Medium blogs and of course Google as search engine. These set of tools together have provided better flexibility to move to a possible approach while writing this research paper.

## 4. CONCLUSION

Handwritten digit recognition is the fundamental step to the vast field of Artificial Intelligence and Computer Vision. With the advance of technology, every day there are new algorithms coming up which can make computers do wonders. Machines can recognize images and understand handwriting of different people which humans themselves have failed to understand or comprehend from the image. As seen from the outcomes of the experiment, CNN proves to be far better than other classifiers.
The results can be made more accurate with more convolution layers and more number of hidden neurons. The target of the proposed work was to open the way to digitalization. Though the goal is to just recognize the digits but it can be extended to letters and then a person's handwriting.

This can completely abolish the need of typing. People can straight away write down in their own handwriting and then convert it in digital form by just clicking pictures of it. Digit recognition is an excellent prototype problem for learning about neural networks and it gives a great way to develop more advanced techniques of deep learning.

The broader aim in mind was to develop a M.L. model that could recognize people's handwriting. However, as we began

 

developing the model, we realized that the topic in hand was too tough and would require tremendous data to learn. Example to accurately classify a cursive handwriting will be very tough.

Thus, we settled on classifying a given handwritten digit image as the required digit using three different algorithms and consequently testing its accuracy. In future we are planning to further explore the topic to recognize people's handwriting.

## 5. REFERENCES


[1] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines, 2001. Software is available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[2] Hiroshi Sako Hiromichi Fujisawa Cheng-Lin Liu, Kazuki Nakashima. handwritten digit recognition: benchmarking of state-of-the-art techniques.

[3] Y. Le Cun et al. Comparison of learning algorithms for handwritten digit recognition. International Conference on Artificial Neural Networks, pages 53–60, 1995.

[4] John C.Platt Patrice Y.Simard, Dave Steinkraus. Best practices for convolutional neural networks applied to visual document analysis.

[5] David G.Stork Richard O.Duda, Peter E.Hart. Pattern Classification. John Wiley & Sons, Inc., 2 edition, 2001.

[6] Changsong Liu Xuewen Wang, Xiaoqing Ding. Gabor filter-based feature extraction for character recognition.

[7] H. P. Graf, W. Hubbard, L. D. Jackel and P. G. de Vegvar, "A CMOS Associative Memory Chip", Proc. IEEE 1st Int'l Conf. on Neural Networks, vol. III, pp. 461-468, 1987.

[8] D. H. Hubel and T. N. Wiesel, "Receptive Fields Binocular Interaction and Functional Architecture in the Cat's Visual Cortex", J. of Physiology, vol. 160, pp. 106-154, 1962.

[9] Y. Le Cun, Modèles Connexionnistes de l'Apprentissage, 19877.

[10] Y. Le Cun, "Generalization and Network Design Strategies" in Connectionism in Perspective, Switzerland, Zürich:North Holland, pp. 143-155, 1989.

[11] N. J. Naccache and R. Shingal, "SPTA: A Proposed Algorithm for Thinning Binary Patterns", IEEE Trans. Systems Man and Cybernetics, vol. SMC-14, pp. 409-418, 1984.

[12] W. C. Naylor, "Some Studies in the Interactive Design of Character Recognition Systems", IEEE Transaction on Computers, vol. 20, pp. 1,075-1,088, Sept. 1971.

[13] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning Internal Representations By Error Propagation" in Parallel Distributed Processing: Explorations in the Microstructure of Cognition, MA, Cambridge:Bradford Books, vol. I, pp. 318-362, 1986.


 