ABSTRACT

Handwritten Digit Recognition has been a topic of research in the field of OCR since 1998, first performed by LeCun using MNIST dataset with 12% of error rate. Since then a large number of researchers have worked on the problem using different approaches, being able to achieve an astonishing state of art performance. This project aims to add one more stone to the pile. This authors intend to design a Feedforward Neural Network model from scratch, that will be able to recognize the handwritten digits. Different algorithms like Mini-Batch Gradient Descent, Back Propagation and Adam would be used to create the model. Optimization techniques like Batch normalization, Dropout regularization and hyper parameter tuning would be used to fine-tune the designed model. The authors also intend to implement all the concepts and mathematical models necessary for the process themselves and try to achieve the best result that is possible.

Keywords: ***Handwritten Digit Recognition, Feedforward Neural Network, Back Propagation, MNIST database, Deep Learning***

Table of Contents

CHAPTER 1: INTRODUCTION

1.1 Problem Statement

Humans' writings are often asymmetric with the varying form of size, width, orientation and distortion of the characters, digits being no exception. So when it comes to recognizing these characters, it becomes a great deal for a dumb machine like computer. Handwritten Digit Recognition is an approach of enabling computers to perform this difficult task through a complicated process of learning called Machine Learning (Deep Learning). This approach makes these computers intelligent enough to recognize the handwritten digits with a small error rate.

Several attempts have been made in the past to address this issue and a number of researchers have been able to achieve astonishing results. In recent days, a lot of people are working on the field using different frameworks and libraries. This has, however, limited the in-depth understanding of the process that the researchers used to achieve these results. This project aims to add one more stone to the pile but without over-running the actual process. This project intends to design a Feedforward Neural Network model from scratch, that will be able to recognize the handwritten digits. The authors intend to implement all the concepts and mathematical models necessary for the process simply by using the python programming language with its core packages used for scientific computing and try to achieve the best result that is possible.

1.2 Objective

- To design a Multi-Layer Feedforward Neural Network Model that recognizes Handwritten digits (one at a time) with high accuracy.
- To properly analyze and optimize the designed model using different model improvement techniques such as normalization, regularization, hyperparameter tuning, etc. and optimization algorithms like Adam and Minibatch Gradient Descent.
- To implement the components of the project from scratch so that the authors could have an in-depth understanding of Deep Learning.

CHAPTER 2: LITERATURE REVIEW

Handwritten Digit Recognition has been a topic of research in the field of OCR since 1998, first performed by LeCun using MNIST dataset with 12% of error rate. Since then a large number of researchers have worked on the problem using different approaches, being able to achieve an astonishing state of art performance. [1]

The first classifier, designed by LeCun, was a single layer linear model with no preprocessing of the dataset. It was then followed by a large number of classifiers built using different machine learning approaches such as KNN, SVM etc. Along with that, the Neural network based model kept on rising outperforming all other earlier models. Over the years LeCun himself experimented with a number of models with increased number of hidden layers and hidden units beating his own records. In 2003, a team of researchers led by Patrice Y. Simard were first to reduce the error rate below 1% using the Neural Network(NN) based model. They used 2-layer NN and 800 Hidden Units(HU) with elastic distortion. They used two simple cost function, MSE and Cross entropy each resulting in the error rate of 0.9% and 0.7% respectively. [2] Hinton followed shortly using a 3-layer NN with 500+300 HU, softmax function, cross entropy loss function and weight decay to achieve error rate of 1.53%. [3]

The next record breaking performance using simply MLP was achieved by a team of researchers led by Dan Claudiu Ciresan in 2010. His team used a 6-layer NN with elastic distortion and trained the model over GPU to achieve the error rate as low as 0.35%. [4]

Besides the use of MLP, various other Neural Network architectures, in addition with different optimization techniques, have been trained using MNIST dataset and used to classify the handwritten digits. They too have shown a significant level of performance as that achieved by the MLP. Among all CNN seems to be the most promising architecture that has been able to stay at the top of the MNIST handwritten digits Classification benchmarks. [9]

The table below gives a brief preview of the work done in classifying Handwritten digits using MNIST dataset. [10]

Classifier	Error Rate(%)	Reference
linear classifier (1-layer NN)	12	LeCun et al. 1998 [1]
K-NN with non-linear deformation (P2DHMDM)	0.52	Keysers et al. 2007 [5]
Virtual SVM, deg-9 poly, 2-pixel jittered	0.56	DeCoste and Scholkopf, 2002 [6]
2-layer NN, 800 HU, cross-entropy [elastic distortions]	0.7	Simard et al. 2003 [2]
6-layer NN 784-2500-2000-1500-1000-500-10 (on GPU) [elastic distortions]	0.35	Ciresan et al. 2010 [4]
Convolutional net Boosted LeNet-4, [distortions]	0.7	LeCun et al. 1998 [1]
Committee of 35 conv. net, 1-20-P-40-P-150-10 [elastic distortions]	0.23	Ciresan et al. CVPR 2012 [7]
Convolutional net With DropConnect	0.21	LeCun et al. ICML 2013 [8]

Table 1:Past Research Projects in Handwritten Digit Recognition

CHAPTER 3: METHODOLOGY

3.1 Resource Requirements

Various hardware and software resources required for the successful completion of this project are:

Hardware: Desktop PC/ Laptop with Multi core CPU

Software: Python Programming Language with Numpy, Matplotlib and Scipy as its core packages, Anaconda Navigator

3.2 Process Model

There are a huge number of process models that can be adopted to carry out a project. Yet, these existing models are either application development-centric or enterprise architecture focused or rooted in hardware or software development approaches. So they face a significant challenge when used with the unique lifecycle requirements of AI/ML projects. So there is a need of project management methodology that takes into account the various data-centric needs of AI while also keeping in mind the application-focused uses of the models and other artifacts produced during an AI lifecycle.

CPMAI: The Cognitive Project Management for AI Methodology, developed by Cognilytica with dozens of other organizations, is one of the best methodologies adopted for a large number of real-world Big data and AI projects. CPMAI extends the well-known CRISP-DM methodology with AI and ML specific documents, processes, and tasks. The CPMAI methodology also incorporates the latest practices in Agile Methodologies and adds additional DataOps activities that aim to make CPMAI data-first, AI-relevant, highly iterative, and focused on the right tasks for operational success. [11]

Since the methodology is for large organizations and is production oriented, many of its components are quite not feasible for a small project like this. So the methodology was adopted with necessary changes to best suit the needs of this project.

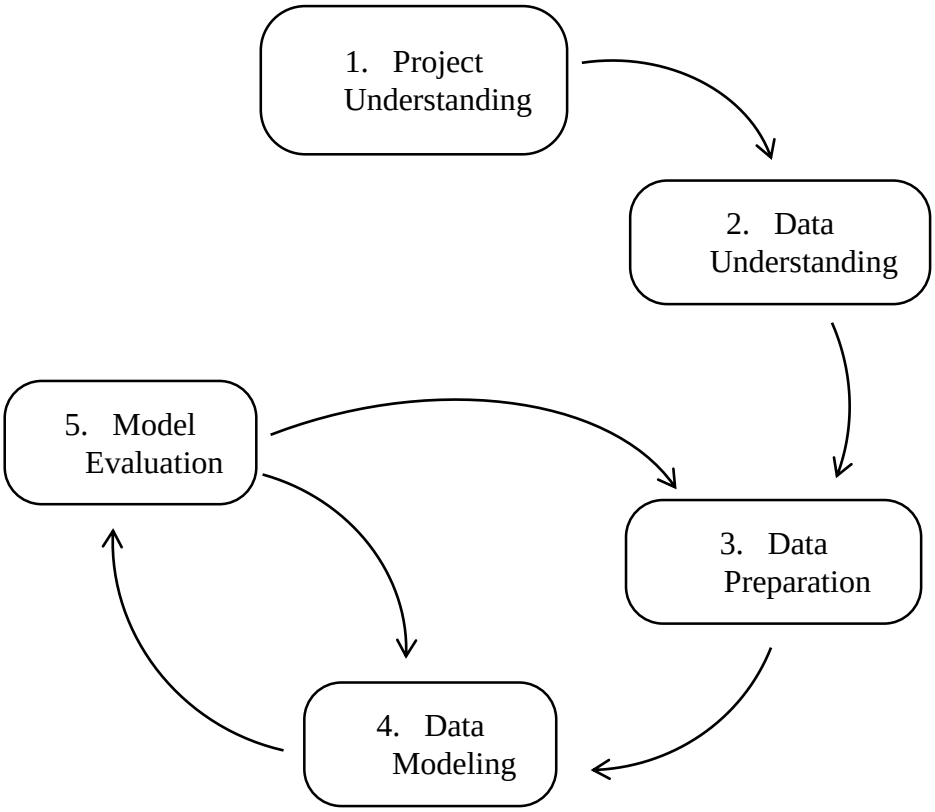


Figure 1: CPMAI Based Process Model

The methodology consists of 5 major phases:

- **Project Understanding:** This phase starts since the inception of the project idea. It then manifests to project selection followed by the understanding of the project characteristics and requirements. It then establishes the project among the *seven patterns of AI*. Along with the creation of the project plan, the scope and objective of the project is also defined in this phase.
- **Data Understanding:** In this phase the data requirements of the project are identified. The data are then looked for in different sources and checked for its authenticity. The data format is evaluated and the quality of the data is ensured. On completion, this phase is able to give a good description of the data to be used.
- **Data Preparation:** This phase includes the importing of the dataset and processing it so that the dataset becomes ready for training the model. Here, the data is cleaned, normalized and the label is encoded as necessary. Data Augmentation is carried out to upscale the data volume whenever required.

Besides, the main data is partitioned into train, dev and test set in a certain manner that are used to train the model.

- **Modeling:** This phase helps to select a modeling technique and any required algorithms. Here, we make all the assumptions that are required to build a model including its architecture, generate test designs and train the model. Besides this phase also includes different approaches to optimize, retrain and scale the model. This phase combined with the evaluation phase takes majority of the lifespan of the project.
- **Model Evaluation:** This is one of the most crucial phases for designing a better model. The evaluation of the model gives feedback about how well the model is learning and generalizing the data. Here, various outcomes and visualizations of the model like accuracy, learning curve, loss curve etc. helps to analyze the state of the model and gives intuitions about the what approach to select to better optimize the model. Once the problem is identified and a suitable approach is selected, this phase loops back to either data preparation phase for collection of more data or the modeling phase for redefining and retraining of the model with necessary changes.

3.3 Tools and Techniques:

3.3.1 Design Tools

Following are the design tools used by the authors in this project.

MS Visio: MS Visio is a diagram design tool used to create diagrams for varieties of field ranging from floor designs to software engineering designs and many more. In this project, the authors have used this tool to create DFD and the project Schedule (Gantt Chart).

MS Word: MS Word is a document design tool that provides an extended range of useful features. Various reports, including the proposal itself, are designed by the authors using this tool.

3.3.2 Implementation Tools

The implementation for this project is totally based on the Jupyter platform, an ecosystem that enables others to build tools on top of it. Jupyter is a collaboration tool for writing and sharing code and text within the context of a web. The code runs on a

server, that can be anywhere, and the results are turned into HTML incorporated into the page one is writing.

Jupyter is built from three parts:

1. **Jupyter Notebook- The Front End**: It is a web-based interactive computational environment for creating Jupyter notebook documents. It allows one to write code and edit and run them in the notebooks.
2. **The Jupyter server**: It is a relatively simple application that runs on your laptop, or a multi-user server like JupyterHub.
3. **The kernel protocol**: It allows the server to offload the task of running code to a language-specific kernel.

JupyterLab: It is a Jupyter based IDE that offers all the familiar building blocks of the classic Jupyter Notebook (notebook, terminal, text editor, file browser, rich outputs, etc.) in a flexible and powerful user interface. [12]

CHAPTER 4: SYSTEM ANALYSIS

4.1 Functional requirement

Functional requirements for a model describe what the model should and should not do. It also describes how the model should behave for a particular set of inputs. The functional requirements of the given model are as follows:

- The model should be able to classify any new image of a handwritten digit with maximum accuracy.
- The model should not classify any alphabets and characters as a digit.

4.2 Data Description

4.2.1 MNIST Database

MNIST Database is a large database of handwritten digits' images that is extensively used as a toy database for various image processing systems and machine learning projects in academics and industries. It was adopted from a larger database called NIST, with some manipulation on the images, by a team of researchers lead by Yann LeCun.

The database consists of four files:

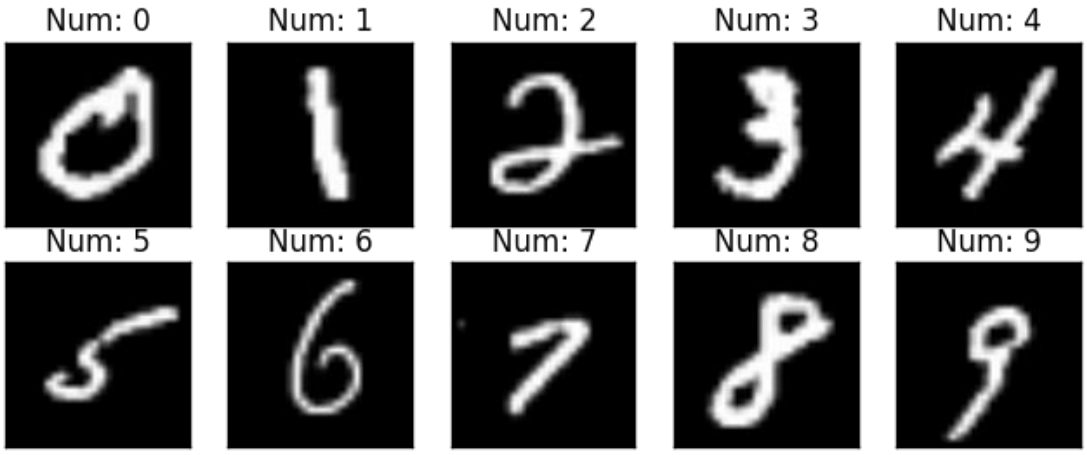
- Train-images-idx3-ubyte.gz: training set images (9912422 bytes)
- Train-labels-idx1-ubyte.gz: training set labels (28881 bytes)
- T10k-images-idx3-ubyte.gz: test set images (1648877 bytes)
- T10k-labels-idx1-ubyte.gz: test set labels (4542 bytes)

The data is stored in the idx file format designed for storing vectors and multidimensional matrices.

The training set consists of 60000 examples and the test set consists of 10000 examples. The first 5000 examples of the test set are taken from the original NIST training set. The last 5000 are taken from the original NIST test set. The first 5000 are cleaner and easier than the last 5000. [10]

4.2.2 Processed MNIST Dataset

The dataset when imported is converted into a set of n-dimensional arrays.



Source: learnmachinelearning.wordpress.com

Figure 2: Sample Images of MNIST Dataset

After importing, the test set is split into development/validation set and test set through random selection of the data resulting in Train/ dev and test set. Then the input image is flattened and normalized. The output labels are transformed into one-hot representation. Finally, the dataset is ready and has the following form

Dataset	**Processed Array size**	**Original Array Size**
Training Set Image	(784, 60000)	(60000, 28, 28)
Training Set Label	(11, 60000)	(1, 60000)
Dev Set Image	(784, 5000)	(5000, 28, 28)
Dev Set Label	(11, 5000)	(1, 10000)
Test Set Image	(784, 5000)	(5000, 28, 28)
Test Set Label	(11, 5000)	(1, 10000)

Table 2: Processed MNIST Dataset format

4.3 Process Modeling

Process modeling is a technique for organizing and documenting the structure and flow of data through a system's processes and/or the logic, policies, and procedures to be implemented by a system's processes.

Data flow diagram (DFD) is an analysis tool to represent the flow of data through an information system. The report presents the context diagram of proposed model as below:

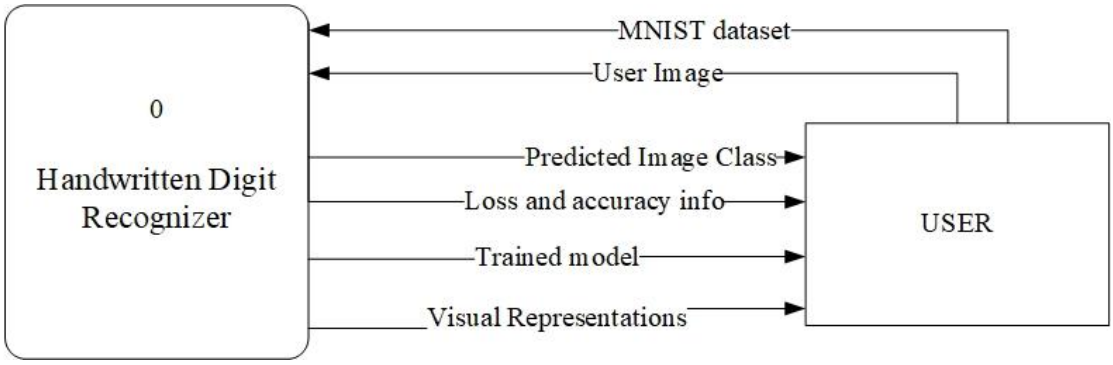


Figure 3: Context Diagram for Handwritten Digit Recognizer

CHAPTER 5: PROJECT SCHEDULE PLAN

This project begun from the mid of the February and is expected to last till the last week of May. Various activities of the project life cycle would be carried throughout this period as shown by the Gantt chart below. Besides, various reports prepared in between would be submitted to the college administration at the time of their respective defense proposal, mid and final defense.

ID	Task Name	Start	Finish	Duration
1	Project Selection	2/17/2020	2/23/2020	7d
2	Domain Study and Collection of Literature	2/24/2020	3/4/2020	10d
3	Proposal Preparation	3/5/2020	3/11/2020	7d
4	Data Collection and Processing	3/14/2020	3/20/2020	7d
5	Base Model Preparation	3/21/2020	4/4/2020	15d
6	Mid-term Report Preparation	4/5/2020	4/11/2020	7d
7	Model Analysis and Optimization	4/12/2020	5/12/2020	31d
8	Documentation	2/17/2020	5/10/2020	84d
9	Final Report Preparation	5/13/2020	5/22/2020	10d

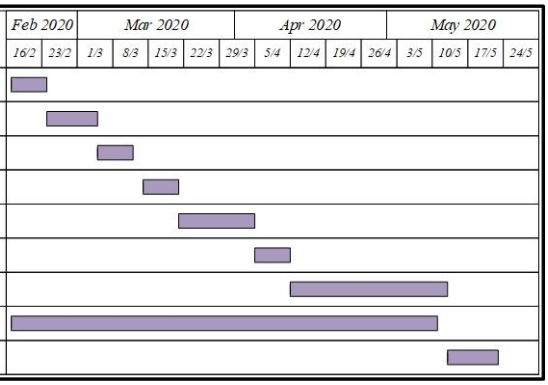
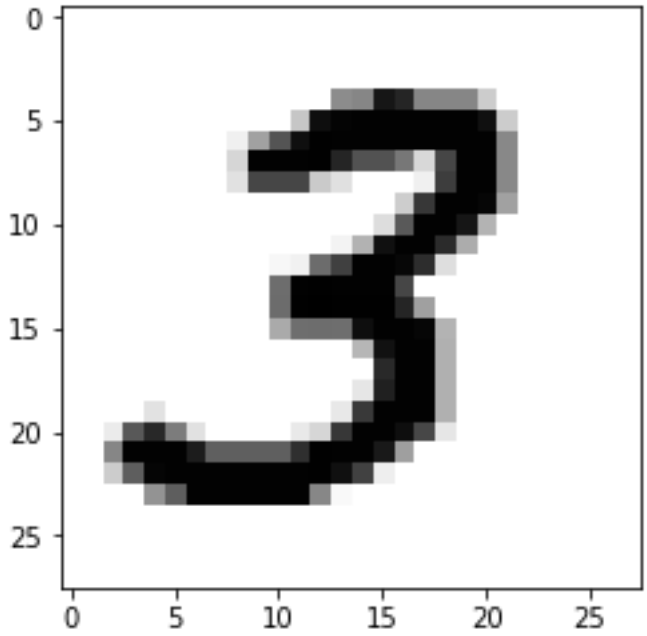


Figure 4: Gantt Chart for Project Scheduling of Handwritten Digit Recognizer

CHAPTER 6: EXPECTED OUTPUT

The final model should be able to classify any new image of a handwritten digit with maximum accuracy. If the input image is of 3, the model should be able predict that its 3 as shown below.



Actual Label: 3, Predicted Label: 3

Figure 5: Sample Image for Prediction

REFERENCES

Conference Paper:

[1] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.

[2] P. Y. Simard, D. Steinkraus and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, Edinburgh, UK, 2003, pp. 958-963

[3] G. E. Hinton, S. Osindero and Y. Teh, "A Fast Learning Algorithm for Deep Belief Nets," in *Neural Computation*, vol. 18, no. 7, pp. 1527-1554, July 2006.

[4] D. C. Ciresan, U. Meier, L. M. Gambardella, J. Schmidhuber, "Deep Big Simple Neural Nets Excel on Handwritten Digit Recognition." in *Neural Computation*, vol. 22, no. 12, pp. 3207–3220, Dec 2010.

[5] D. Keysers, T. Deselaers, C. Gollan and H. Ney, "Deformation Models for Image Recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 8, pp. 1422-1435, Aug. 2007.

[6] D. Decoste, B. Schölkopf, "Training Invariant Support Vector Machines," Machine Learning, **46,** 161–190, 2002.

[7] D. Ciresan, U. Meier and J. Schmidhuber, "Multi-column deep neural networks for image classification," 2012 *IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI*, 2012, pp. 3642-3649.

[8] L. Wan, M. D. Zeiler, S. Zhang, Y. LeCun, R. Fergus, "Regularization of Neural Networks using DropConnect," *ICML*, 2013.

Web Links:

[9] MNIST on Benchmark.AI, "MNIST Classify handwritten digits" , available on-line at https://benchmarks.ai/mnist, 2020 .

[10] MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges, "THE MNIST DATABASE of handwritten digits", available on-line at http://yann.lecun.com/exdb/mnist/.

[11] Cognilytica, “CPMAI: Methodology for Implementing AI Projects Successfully”, available on-line at https://www.cognilytica.com/cpmai-methodology/, 2020.

[12] Oreilly, “What is Jupyter?”, available on-line at https://www.oreilly.com/radar/what-is-jupyter/, June 20, 2017.

UNDERTAKING

This project is created by the authors with long and rigorous effort and not copied from any platform like github. The authors take full responsibility of any plagiarism of the project code. If some portion of their code has to be adopted, the authors would not fully replicate the code from the source but instead change it to fit their need with some improvement to the code and also cite the source honestly.

……..……..

Biswas Poudyal (10526/073)

……..

Krishna Khanal (10691/073)

…….……..

Sameer Kattel (10713/073)

…….……..

Subina Lama (10720/073)

SUPERVISOR / PROJECT COORDINATOR'S COMMENTS AND REMARKS

……

Er. Dhiraj Kumar Jha

Project Coordinator

Orchid International College

Bijayachowk, Gaushala, Kathmandu