Convolution Neural Network for Handwriting Recognition

Harsh Dhillon
Western University, Canada
hdhill48@uwo.ca

Pankaj Kumar
Western University, Canada
pkumar@uwo.ca

Abstract

Our project aims to create a functional neural network architecture for accomplishing the task of handwriting recognition. We will design a Multilayer Perceptron (MLP) and convolution neural net (CNN) in Keras with TensorFlow in the backend. We will use MNIST database for training our models. Further we will test the designed deep learning model and assess its efficiency using multiple evaluation criteria and visualizations.

Acronyms

- OCR Optical character recognition
- MLP Multi-Layered Perceptron
- CNN Convolution Neural Network
- SVM Support Vector Machine

Introduction

Handwriting recognition and conversion into digital text form has been a key area of interest in the field of computer science and machine learning. Banking industry and postal services work with handwritten text daily in the form of bank cheques, signature, written postal address, application forms etc. To automate this system for accommodating a bigger number of clients and their data, there is a crucial requirement of a software system that can decipher handwritten text and convert it into a digitalized text format without any human assistance. Digital text has a lot of advantages in terms of storage, accessibility and distribution as compared to the traditional handwritten text. We aim to provide a framework using machine learning concepts and deep learning methods to convert handwritten text into a digital format with a high accuracy rate than the traditional OCR frameworks. We will also explore past research and implementation methods to understand this area of study. Furthermore, we will base our applied solutions on the current methods and up-to-date research in the field of image recognition, computer vision and deep learning.

Background

MNIST is the most widely used benchmark for isolated handwriting recognition. More than a decade ago, Artificial neural networks called Multilayer Perceptron or MLP were the first architecture solution for classifying digits using the MNIST dataset. Most had only few units or few neurons per layer. This was still one of the biggest feasible MLP possible at the time, given that the CPU cores were many times slower compared to the modern higher processing power cores available today. However, few complex architectures still outperformed MLP. As listed on the MNIST database website, most efficient architectures leaned towards complex variants of Support Vector Machines (SVM) and combinations of Perceptron and SVM. Yann Lecun applied his Convolution Neural Network architecture called LeNet to this problem, which became the most efficient neural network architecture for this task of digit recognition till date with an error rate of 0.35%. That was also the very first application of CNN's to any practical problem. Many variants of LeNet and hybrid CNN architectures exist today. Recent widely successful architecture been the AlexNet, which is written in CUDA language and uses GPU processing power to run and solve classic problems in record breaking time, with extreme low error rates. We are focussing on handwriting recognition because it's an excellent problem for learning neural networks in general. Furthermore, it's a great way to develop more advanced techniques and get familiar with deep learning methods such as Convolution Neural Networks. We aim implement more complex architectures which can recognize complete words and sentences in future and work on audio data to decipher speech and delve into speech related interesting problems that exist in field of deep learning.

Related Work

Ray Kurzweil invented the first noteworthy optical character recognition (OCR) software in 1947[2]. His architecture used an advanced matrix method for pattern matching, which involves comparing an image to a pre-stored glyph on a pixel-by-pixel basis and choose the nearest match for prediction outcome.

Yann LeCun et. al used gradient based learning methods in association with multi-layered machine learning framework, a predecessor to modern deep learning models, to implement the handwriting recognition system in 1998[3]. His approach was widely adopted by US postal service and banking industry. Today around 10% checks in America are still processed using LeNet, so it had a wide practical worth. His formulated architecture integrated various processes such as segmentation, feature extraction, classification, contextual post processing and language modeling into a single machine learning framework, which was also trained in an end to end fashion.

Andrew Ng et. al applied CNN to the problem of text recognition and expanded it into a wider context of signs, license plates, handwritten notes etc. His framework deciphered the text in the images by using a sliding window algorithm [4]. The final approach for architecture used a CNN with two convolution layers, two average pooling layers and a fully connected layer to classify each character in the image. Various newer models such as RNN and LSTM are also being used today to solve this classic problem.

We aim to use the past research and its rich information history to create our own handwriting recognition system first using a Multilayer Perceptron and then building CNN model LeNet, both architectures will be trained using the MNIST database. We will develop and optimize our architectures in a such a way that it provides a modest accuracy with low error rates, which further will be visualised properly using plotting functions provided by Keras.

Database

MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for image processing and computer vision systems[5]. Database is also widely used for training and testing in the field of machine learning[6]. MNIST consists of two datasets, one for training which contains 60,000 images and one for testing which contains 10,000 images for validation. The original images from NIST database were size normalized and processed with an anti aliasing algorithm. Each image in the MNIST database is 28×28 pixels and is centered by computing center of mass of the pixels. Images also contain grayscale levels, providing each pixel in the image with a value from 0 to 255. Each image has a total of 784 pixels, so the input layer in our handwriting recognition architecture has this number of input units.

Figure 01. MNIST Sample Images

Analysis and Visualization

To analyse and solve this task, we are using python programming language. We are also using modern neural network and machine learning libraries to hand code and design Multilayer Perceptron and Convolution Neural Network architectures. We will use the following toolsets to solve our problem –

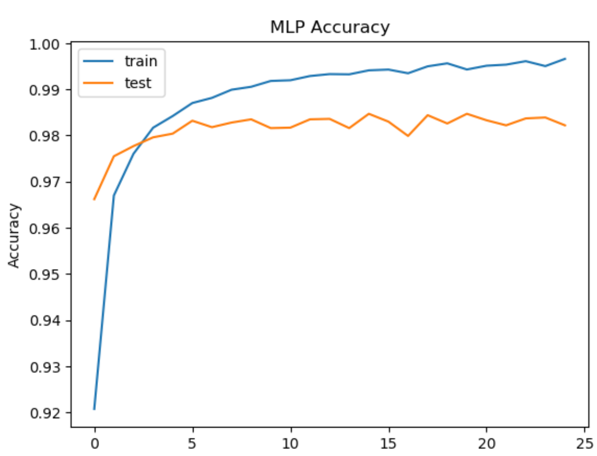
- Keras is an open source neural network library written in python. It is quite modular and very user friendly. It contains numerous implementations of commonly used neural network building blocks which enables quick experimentation and development of diverse neural net architectures for performance evaluation[7].

- TensorFlow is an open source machine learning framework. It works in symbiosis with Keras to practically build neural networks using a line by line approach[8]. It provides a stable API for the python language used in our project.

Multilayer perceptron is built using a 2-layer architecture. Input layer has 28×28 = 784 input units, a variable number of hidden units and 10 output units each representing a number from the range 0-9. Because CNN's tend to work more efficiently with raw pixel data rather than features or parts of an Image, we will use complete digit images[9].

We also experimented with different hyperparameters present in our architecture and fine tuned them by selecting appropriate size of hidden units and trialling with drop out value to find optimal settings for both of our models. In order to save time in training and testing also did kept the number of epochs in our architecture to a minimum[10]. Furthermore, we also experimented with the batch size of our dataset and used the cross-validation approach to substantiate the stability of our design.

We visualize our results first using model accuracy plot, which plots the accuracy on Training and Testing datasets. From the



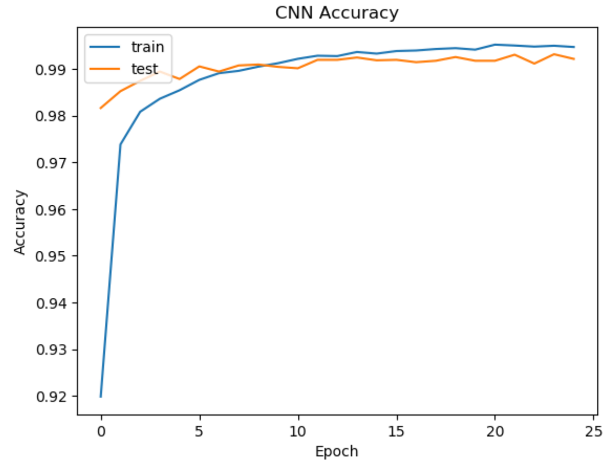


Figure 02. Accuracy Plots of MPL and CNN Models

accuracy plot we can infer that MLP and CNN has almost a same accuracy curve with the training dataset. But when we test the models on the testing dataset, CNN's accuracy is much higher than MLP. CNN also requires less epochs to converge to such high level of accuracy as compared to the MLP model. This shows the processing power saving capabilities of using CNN model over a tradition MLP.

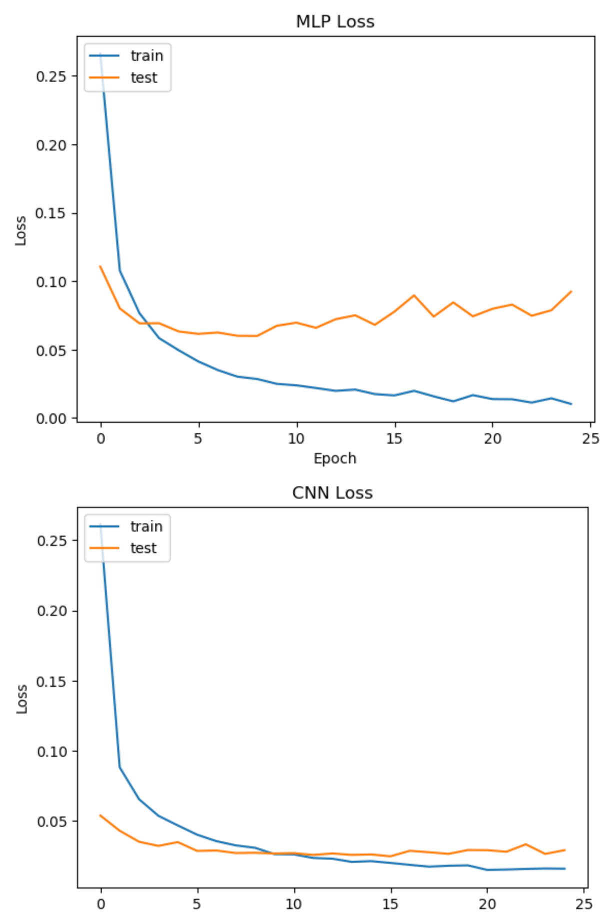


Figure 03. Loss Plots of MLP and CNN models

We also visualized the loss value for both models using loss plots. A loss function is a measure of how good a prediction model does in terms of being able to predict the expected outcome. Ideally, a good prediction model will have a reducing loss value over the epoch cycles.

We are using cross-entropy loss for our project. Cross-entropy loss, or log loss, measure the performance of a classification model whose output is a probability value between 0 and 1[11]. The cross-entropy loss is computed as follows:

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = -\frac{1}{N}\sum_{i}^{N}[y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)]$$

From the plotting of loss function, we can deduce that comparable performance in terms of better prediction outcomes of our model on both training and testing datasets. The result shows that MLP and CNN have almost a very similar loss curve on the training dataset, but the CNN model is able to converge to a minimum in fewer epochs as compared to the MLP on the testing dataset. This demonstrates that CNN models have a higher prediction accuracy.

To further compare both models visually, we also plot confusion matrix for each model. A confusion matrix or an error matrix[12] is a specific table layout that allows visualization of the performance of our models. Each row of matrix represents the instances in a predicted class while each column represents the instances in an actual class.

	predicted 0	predicted 1	predicted 2	predicted 3	predicted 4	predicted 5	predicted 6	predicted 7	predicted 8	predicted 9	recall
actual 0	954	0	0	7	1	10	6	3	7	3	96%
actual 1	0	1031	4	3	1	4	1	2	16	2	97%
actual 2	12	21	852	18	11	8	14	20	29	5	86%
actual 3	2	5	9	899	1	71	0	12	23	7	87%
actual 4	2	8	2	2	861	7	7	1	4	89	88%
actual 5	7	5	9	24	3	833	12	8	12	2	91%
actual 6	11	6	2	0	6	31	902	0	8	1	93%
actual 7	3	10	5	3	7	7	1	1041	0	14	95%
actual 8	2	28	4	29	2	31	1	9	882	21	87%
actual 9	7	3	1	7	10	11	1	44	4	873	91%
precision	95%	92%	96%	91%	95%	82%	95%	91%	90%	86%	accuracy 96%

Fig 04. Confusion Matrix – Multi Layered Perceptron

	predicted 0	predicted 1	predicted 2	predicted 3	predicted 4	predicted 5	predicted 6	predicted 7	predicted 8	predicted 9	recall
actual 0	988	0	0	0	0	0	1	1	1	0	100%
actual 1	0	1059	2	0	0	0	1	1	1	0	100%
actual 2	6	6	949	5	1	0	0	19	2	2	96%
actual 3	0	2	0	1015	0	0	0	3	2	7	99%
actual 4	1	7	0	0	959	0	1	4	0	11	98%
actual 5	5	1	0	12	3	868	15	2	4	5	95%
actual 6	0	1	0	1	1	1	963	0	1	0	99%
actual 7	0	3	1	0	0	0	0	1083	0	3	99%
actual 8	8	2	1	2	1	2	9	6	970	8	96%
actual 9	7	2	0	4	5	0	0	14	2	927	96%
precision	97%	98%	100%	98%	99%	100%	97%	96%	99%	96%	accuracy 98%

Fig 05. Confusion Matrix – Conv. Neural Network

We can deduce from both confusion matrix plots that CNN model has a higher accuracy rate of 98% as compared to MLP model, which has an accuracy rate of 96%. CNN correctly predicted majority of digit classes, whereas we can infer that MLP had certain digit classes such as Number 3 and 9 which gave high rate of prediction inaccuracies. The confusion matrix plotting demonstrates that CNN is a much efficient and reliable model of choice to get accurate class predictions for the task of digit recognition.

Conclusion

We presented the applied problem of handwritten digit recognition, which is one of the key areas of interest in the field of computer science. We implemented the traditional Multi-Layered Perceptron model and Convolutional Neural Network architecture as our data science and machine learning methods to solve it. We used MNIST database, which gave us large handwritten digit data resource to work with. Further, we also compared both of these models and visualized their performance accuracy and loss value to understand how each model performs for this particular problem. To identify the accuracy for each digit class, we also implemented and visualized confusion matrix. We gained 98% accuracy rate for CNN model and 96% accuracy rate for MLP model. In future, we aim to code and implement deeper CNN models and study RNN, LSTM models to perform single word and sentence recognition.

References

[1] U. Marti and H. Bunke. A full English sentence database for off-line handwriting recognition. In Proc. of the 5th Int. Conf. on Document Analysis and Recognition, pages 705 - 708, 1999.

[2] Kurzweil Computer Products. http://www.kurzweiltech.com/kcp.html

[3] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner: Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, 86(11):2278-2324, November 1998

[4] Ng, A.Y., et al. "End-to-end text recognition with convolutional neural networks." *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012.

[5] THE MNIST DATABASE. (n.d.). Retrieved from http://yann.lecun.com/exdb/mnist/

[6] Ciresan, D., et al. “Multi-Column Deep Neural Networks for Image Classification.” 2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012, doi:10.1109/cvpr.2012.6248110.

[7] Chollet, François. "Keras: The python deep learning library." Astrophysics Source Code Library (2018).

[8] Abadi, Martín, et al. "Tensorflow: a system for large-scale machine learning." OSDI. Vol. 16. 2016.

[9] Niu, Xiao-Xiao, and Ching Y. Suen. "A novel hybrid CNN–SVM classifier for recognizing handwritten digits." Pattern Recognition 45.4 (2012): 1318-1325.

[10] Jozefowicz, Rafal, Wojciech Zaremba, and Ilya Sutskever. "An empirical exploration of recurrent network architectures." International Conference on Machine Learning. 2015

[11] Boer, P. D., Kroese, D. P., Mannor, S., & Rubinstein, R. Y. (2005). A Tutorial on the Cross-Entropy Method. Annals of Operations Research, 134(1), 19-67. doi:10.1007/s10479-005-5724

[12] Fawcett, T. (2006). An introduction to ROC analysis. Pattern Recognition Letters, 27(8), 861-874. doi:10.1016/j.patrec.2005.10.010

Appendix

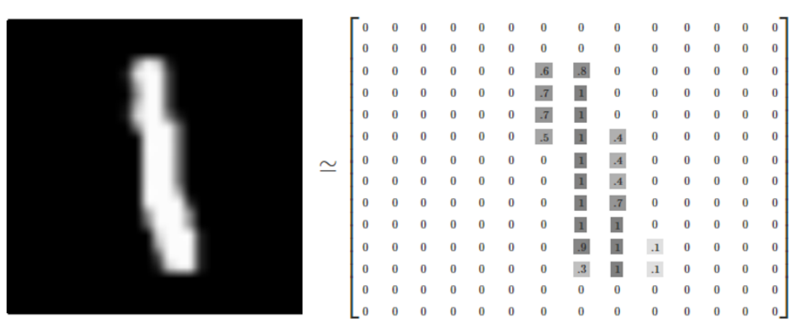


Fig 06. MNIST Data Example and Grayscale Value Matrix

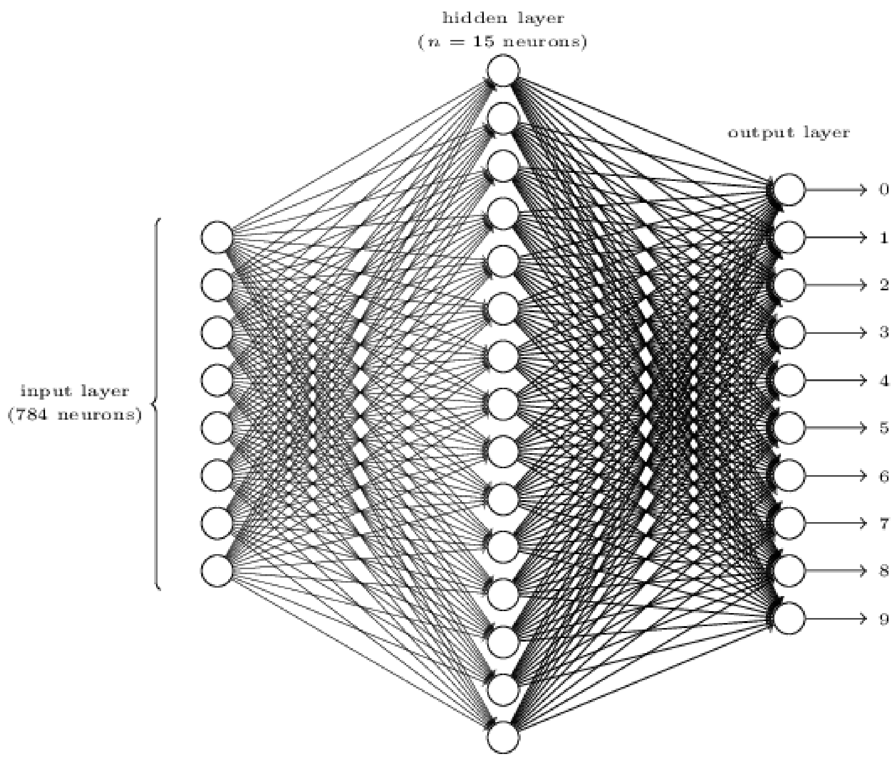


Fig 07. Visual Representation of Multi-Layered Perceptron

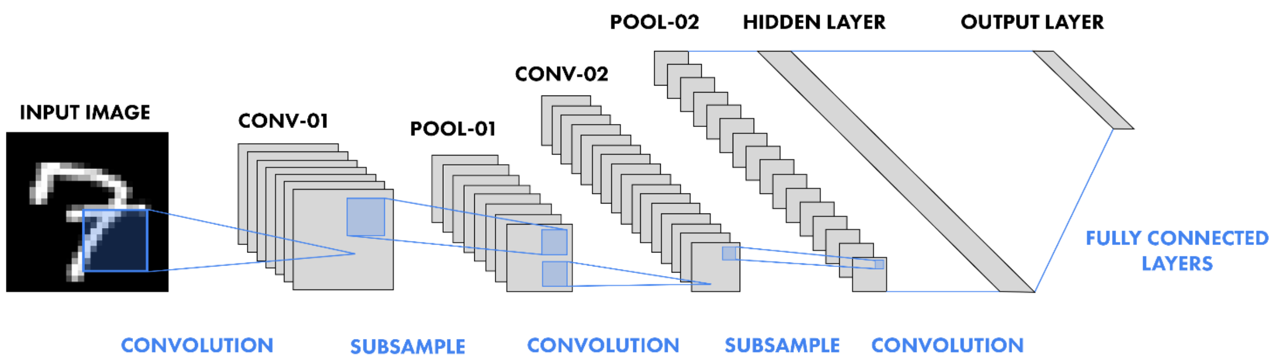


Fig 08. Visual Representation of CNN