



 slington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CU6051NI - Artificial Intelligence

Assessment Weightage & Type

75% Individual Coursework

Year and Semester

Student Name:

London Met ID:

College ID:

Assignment Due Date:

Assignment Submission Date:

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Acknowledgement:

I would like to express my deep gratitude to our respected module leader Mr. Sunil Raut Kshetri for guiding us throughout Artificial Intelligence module. His way of teaching was a key factor for us to understand the technically tough topic of machine learning and intelligence. I was always enthusiastic to learn how an artificially intelligent program is written and what are the necessary pre-requisites which our tutor taught us in the first semester. In the assigned coursework we are asked to develop any artificially intelligent program solving real-life problem.

Furthermore, I would like to express my sincere appreciation to our tutor Mr. Bibek Khanal for guiding us throughout the lab classes teaching us different machine learning algorithm and way to code them. The completion of the coursework would be strenuously difficult without the help of Mr. Bibek Khanal sir's guidance. I am extremely grateful to our instructors for how they have guided us to be a better, keen and vivacious student. Similarly, I would like to extend my thank and deep respect to our Islington college's administration team for making sound teaching-learning experience.

Finally, my profound appreciation to my classmates who have been helpful during the discussion of the coursework and have clarified me about several problems whenever I couldn't understand it.

Abstract:

Human effort can be lowered in recognizing, learning, predicting, and many other tasks with the application of different machine learning models. This project is about doing research on a handwritten digit recognition system and algorithms that is to be used while developing the solution. After studying about several algorithm that could be used for digit classification task, a static Artificial Neural Network model known as Multi-layer perceptron or Deep Feedforward Neural Network is finalized to solve the problem domain. Artificial Neural Networks (ANNs) are widely used today for variety of machine learning applications ranging classification, regression, clustering etc. The code for developing such Artificial intelligent model is described in the report in detail.

Table of Contents

1. Introduction:	1
1.1. Introduction to Artificial Intelligence and Machine Learning:	1
1.2. Explanation of the AI concept used:	4
1.3. Introduction to the Chosen Problem Domain:	6
2. Background:	7
2.1. Research on Chosen Topic/ Problem Domain:	7
2.2. Review and Analysis of Existing Work in the Problem Domain:	11
2.2.1. Solution 1: Handwritten digits recognition with decision tree classification: a machine learning approach	11
2.2.2. Solution 2: Handwritten Digit Recognition Application Based on Improved Naïve Bayes Method	12
2.2.3. Solution 3: Handwritten Digit Recognition Using K-Nearest Neighbour (KNN) Classifier	13
3. Solution:	15
3.1. Explanation of proposed solution:	15
3.2. Explanation of AI Algorithm Used:	17
3.3. Pseudocode of the Solution:	23
3.4. Digit Recognition System Flowchart:	25
3.5. Explanation of the development process:	27
3.6. Achieved Results:	31
4. Conclusion:	36
4.1. Analysis of the work done:	36
4.2. How the solution addresses real world problems:	36
4.3. Further work:	37
4.4. Limitation of the work:	37
5. References:	38
6. Appendix (Plagiarism Report):	42

Table of Figures:

Figure 1 Machine Learning Algorithms (Taffese, 2020).....	3
Figure 2 Biological Neuron Vs Artificial Neuron (Mwandau & Nyanchama, 2018)	4
Figure 3 Comparing different algorithms for digit classification (Shamim, et al., 2018) ...	9
Figure 4 Decision Tree Accuracy Table (Assegie & Nair, 2019)	11
Figure 5 Accuracy for KNN based handwritten digit classifier (Babu, et al., 2014).....	13
Figure 6 A single perceptron	17
Figure 7 A sigmoid function.....	18
Figure 8 A Multi-layer Neural Network.....	19
Figure 9 Flow chart of the solution	25
Figure 10 Importing modules.....	27
Figure 11 Splitting training-testing data	27
Figure 12 Developing the Neural Network Model	28
Figure 13 Training and testing the model	28
Figure 14 Training accuracy	31
Figure 15 Testing accuracy	32
Figure 16 Classification 1	32
Figure 17 Classification 2	33
Figure 18 Classification 3	34
Figure 19 Confusion matrix of the classification	35
Figure 20 Originality Report	42

1. Introduction:

Artificial intelligence (AI) is a branch of study that investigates how to create intelligent programs and machines that can solve problems intuitively without human interference. It is a greater challenge in the present context to develop a purely intelligent computer program that make its own decision without ordering explicitly. In the following coursework, I have taken the challenge to develop a fully intelligent system that, after training on prior data, can make its own decision with greater accuracy. There were several fields of machine learning that could shown such task but I chose the task of automating the handwritten digit recognition task which has a wide range of real-life usage.

1.1. Introduction to Artificial Intelligence and Machine Learning:

Technically, an artificially intelligent agent is a system that can act and think humanly as well as rationally (Norvig & Stuart, 2020). Artificial intelligence as a field of study has a long history, but the first extensive research on the field stands back to 1955 when the world's top mathematician and computer scientists John McCarthy, Claude Shannon, Marvin Minsky and Nathaniel Rochester submitted a proposal for, "*Dartmouth Summer Research Project on Artificial Intelligence*" (McCarthy, et al., 1955). Another Pioneer of Artificial intelligence world is, Alan Turing, who established the Turing test to distinguish humans from machines, highlighted the potential of machines being able to replicate human behaviour and really think (Mintz & Brodie, 2019). For taking an early initiation in development of Artificial Intelligence as a field of study, Marvin Minsky and John McCarthy separately won prestigious Turing award separately in 1969 (Minsky, 1970) and 1971 (McCarthy, 1987) respectively. Artificial Intelligence (AI) itself has several subfields including the machine learning.

The term “Machine Learning” was coined by Arthur Samuel in 1959 (Samuel, 1959). Machine learning, a subfield of artificial intelligence, is a field of study that gives computers the ability to learn without being explicitly programmed. Technically, a machine learning system is defined as, “*A computer program is said to learn from experience E with respect to some task T and some performance measure P , if its performance on T , as measured by P , improves with experience E* ” (Mitchell, 1997).

- **Task (T):** The task T describes how a machine learning should process. Some of the popular machine learning tasks are;
 - Classification,
 - Regression,
 - Transcription,
 - Machine translation,
 - Probability density estimation, etc.
- **Performance measure (P):** The performance measure P is the measure on how accurately the system performs the given task T . We use measures to calculate performance. Some of them are listed below;
 - Accuracy,
 - Error rate,
 - Average log-probability, etc.
- **Experience (E):** The experience E is the learning of skill for the system. For example, in a supervised learning, experience is a labelled dataset on which the system trains.

There are several machine learning algorithms in use which are depicted below;

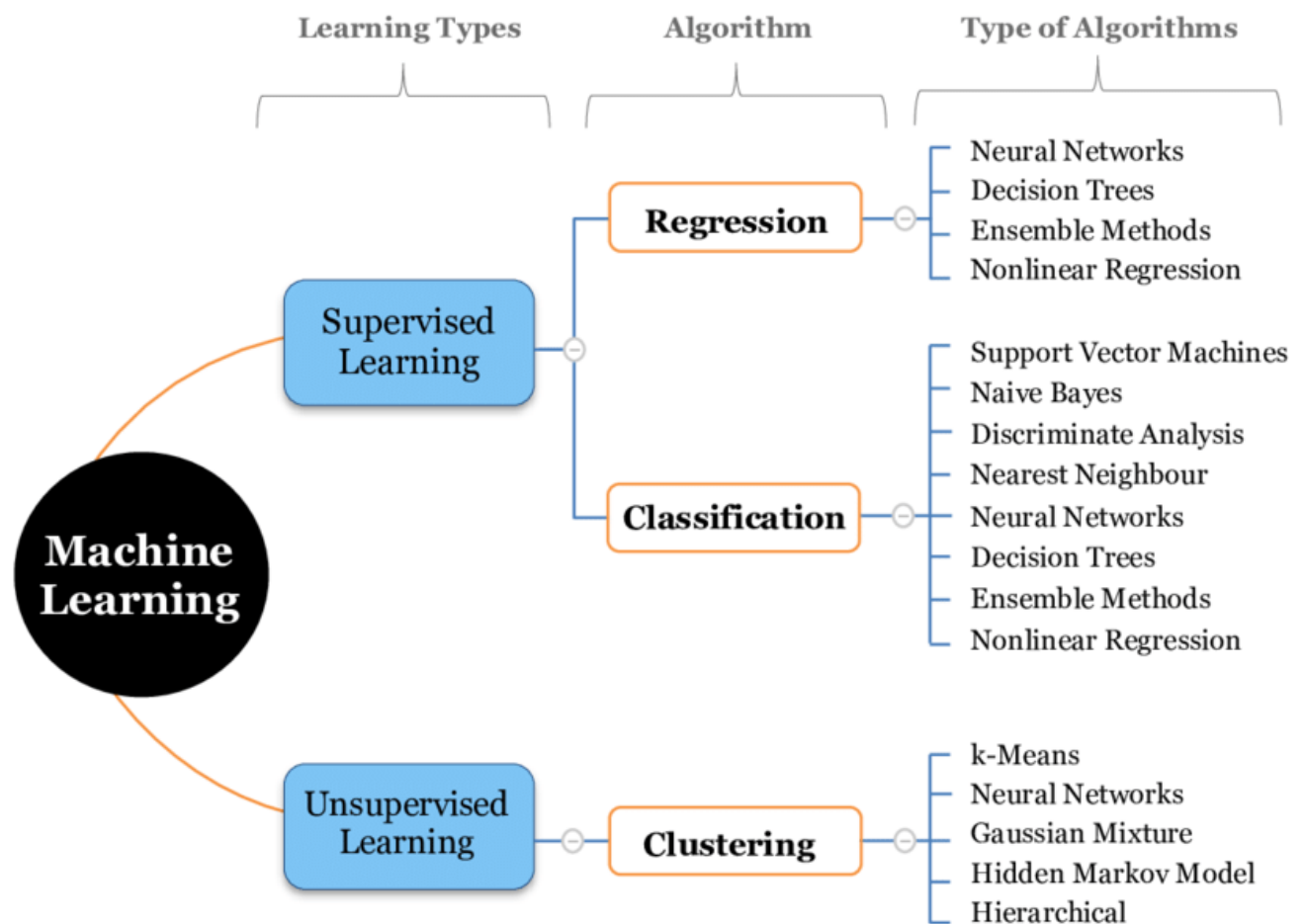


Figure 1 Machine Learning Algorithms (Taffese, 2020)

1.2. Explanation of the AI concept used:

Artificial Neural Network (NN):

Artificial Neural Networks (ANNs) are biologically inspired computational networks that allow learning by example from representative data that describes a physical phenomenon or a decision process. In 1943, Warren McCulloch and Walter Pitts pioneered artificial neural networks by developing a computational model for neural networks based on threshold logic methods (McCulloch & Pitts, 1943). In 1958, Frank Rosenblatt discovered perceptron as an algorithm for pattern recognition which was also a breakthrough in the field of Artificial Intelligence (Rosenblatt, 1958). The real transition and likening to Neural Network were triggered with the development of backpropagation algorithm by Paul John Werbos in 1975 (Werbos, 1975).

An ANN model is inspired from the neuronal connections in the human brain. A typical nerve cell comprises parts like dendrite which receives signal from the other neuron, synapse which is a connection point, axon from which neuronal signal traverses through (if threshold is met) and so on. The following figure compares the biological neuron with an artificial neural network.

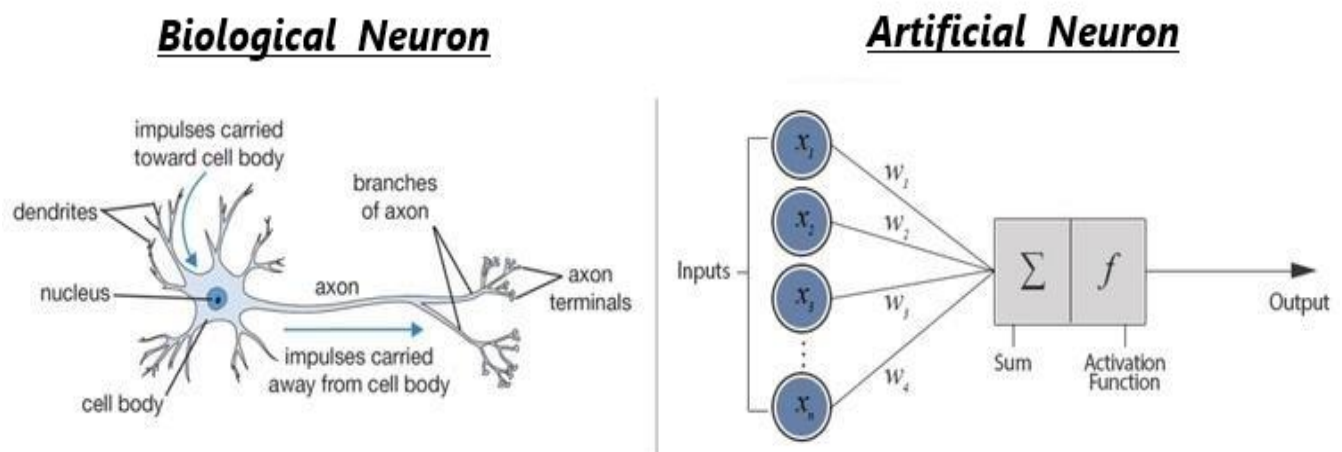


Figure 2 Biological Neuron Vs Artificial Neuron (Mwandau & Nyanchama, 2018)

Neuron is a single entity. When multiple neurons are connected to perform a specific task, they are called a neural network. Similar to interconnected neurons performing specific tasks in the human nervous system, an ANN model also has a network of artificial neurons designed to be trained to learn specific tasks. There are mainly three types of artificial neural networks (ANNs) (Malekian & Chitsaz, 2021),

- **Static ANN:** Models like Multi-layer perceptron (MLP) are static.
- **Dynamic ANN:** Dynamic ANNs includes models like tapped delay lines and Recurrent Neural Network (RNN).
- **Statistical ANN:** Neural Network Models such as radial bias function model and generalized regression neural network model are statistical ANNs.

In the coursework, a static Artificial Neural Network model known as Multi-layer perceptron or Deep Feedforward Neural Network is used to solve the problem domain. Neural networks are among the most effective learning methods now available for specific sorts of challenges, such as learning to comprehend complex real-world sensor data (Mitchell, 1997). All Real-valued, discrete-valued, and vector-valued target functions can all be approximated using neural network learning methods. Thus, supervised, unsupervised and reinforcement learning are all possible using neural network.

1.3. Introduction to the Chosen Problem Domain:

Handwritten Digit Recognition system:

An automated system capable to detecting human handwritten digits is the handwritten digit recognition system. The problem can be written in the language of machine learning as;

- **Task T:** To recognize and classify handwritten digits within the images
- **Performance measure P:** Percentage of digits correctly classified using the algorithm.
- **Experience E:** A database of handwritten digits with classification labels.

Since handwritten digits are not precise and may be generated with a variety of styles, it is a challenging task to detect it. The answer to this problem is a handwritten digit recognition system, which utilizes a picture of a digit to recognize the digit contained in the image. The system uses but machines can do it persistently without fatigue with almost similar certainty every time. Since this is the classification task several classification algorithms such as K-Nearest Neighbour (KNN), Gaussian Naïve Bayes, Decision Tree, Neural Network (NN) can be used. Out of these algorithms, this coursework will use a Neural Network based classification algorithm to automatize the recognition of handwritten digits. The task of recognizing handwritten digits can be extended to be used in numerous other applications such as;

- Number plate recognition system,
- Zip code recognition,
- Bank account number recognition, etc.

2. Background:

2.1. Research on Chosen Topic/ Problem Domain:

Extensive research was done to solve the problem of automatic digit recognition system. First thing that is required to develop a supervised classification algorithm like digit recognition is the availability of dataset that could be used to train the model.

2.1.1. Research No. 1) Searching the ideal dataset:

During research it was found that the dataset provided by Modified National Institute of Standards and Technology (MNIST) was best for training the machine learning algorithm for automated classification tasks after training. The MNIST dataset includes 60,000 training images of handwritten digits ranging from zero to nine, as well as 10,000 test images. As a result, the MNIST dataset comprises ten distinct classes where the handwritten digits are represented as a 28 by 28 matrix with grayscale pixel values in each cell (LeCun, et al., 1998).

2.1.2. Research No. 2) Researching about random forest algorithm:

Multiple researches were done to select one of the above-mentioned methods to solve the above problem. The research was done on using random forest for digit classification tasks (Bernard & Heutte, 2007). This research was done by the scientist of University of Rouen in France in 2007. The scientists have used random forest algorithm on MNIST dataset to make a handwritten digit classifier.

2.1.3. Research No. 3) Understanding about multilayer perceptron:

Similarly, several other algorithms were researched and one of the main algorithms on which extensive research was done is the multi-layer perceptron. Several research papers were read and several other resources were explored to understand the working of the algorithm. The popular machine learning text book written by Tom Mitchell was used to study about the Artificial Neural Network (Mitchell, 1997). Different other books and web-articles were used to further understand about the core principle behind the Artificial Neural Network. Kim Scott's lecture in MIT in 2016, was used to study about the connection between biological and artificial neural networks (Scott, 2016). The lecture connects the neurological vocabular to artificial neural network terminologies where it compares neuron as ANN's node, synapse as the connection, synaptic strength with weight, firing frequency with unit output etc. Another greater resource was of Gill, where the Artificial Neural Network is described in as detail as now where else to be found (Gill, 2021). The article, co-written by Navdeep Singh, depicts in-detail explanation of the Neural Network. Another most important paper that was taken in consideration was the paper authored by Rosenblatt who explains for the first time about the way we could develop an Artificial Neural Network to work like a Biological Neuron (Rosenblatt, 1958).

2.1.4. Research No. 4) Research on AI application:

After researching about the AI algorithm, I dug deeper into its real-life application. One of the papers I found was authored by Gopalakrishnan and Abhirami of Kings College of Engineering in 2021 where they use a handwritten digit recognition system for banking (Gopalakrishnan, et al., 2021).

2.1.5. Research No. 5) Research on Choosing AI algorithm:

Since there are a lot of classification algorithms that could be used to classify the given digits. Different algorithms produce different accuracy and time taken to classify also varies as per algorithms used to train the dataset on. The following research was useful in tentative comparisons of several algorithms that could be used for digits classification;

Table 1: Simulation result based on accuracy and time consumption

Name of Algorithms	Correctly Classified Instances % (value)	Incorrectly Classified Instances % (Value)	Time Taken (seconds)	Kappa Statistic
Multilayer Perceptron	90.37	9.63	3.15	0.893
Support Vector Machine	87.97	12.03	0.56	0.8664
Random Forest	85.75	14.25	0.44	0.8416
Bayes Net	84.35	15.65	0.86	0.8262
Naive Bayes	81.85	18.15	3.45	0.7983
J48	79.51	20.49	0.53	0.7722
Random Tree	85.6	24.94	0.55	0.7228

Table 2: Simulation result based on different error

Name of Algorithms	Mean Absolute Error	Root Mean Squared Error	Relative Absolute Error (%)	Root Relative Squared Error (%)
Multilayer Perceptron	0.023	0.1231	12.78	41.04
Support Vector Machine	0.1611	0.2734	89.49	91.15
Random Forest	0.0593	0.1532	32.97	51.06
Bayes Net	0.0312	0.1745	17.36	58.15
Naive Bayes	0.0361	0.1879	20.06	62.65
J48	0.0444	0.1957	24.66	65.25
Random Tree	0.0499	0.2234	27.72	74.45

Figure 3 Comparing different algorithms for digit classification (Shamim, et al., 2018)

From the above comparisons, Multilayer perceptron is the better for digit recognition task.

There are multiple benefits of selecting ANN for digit recognition tasks. The **advantages** of using Neural Network based classification algorithm are;

- Neural networks can learn on their own and produce output that isn't constrained by the input they receive.
- Instead of using a database, the input data is saved in its own networks. As a result, data loss has no impact on how it functions.
- The neural network will learn from previous instances and adjust when a similar event occurs, allowing them to perform in real-time during an event.
- The network can detect the defect and generate the output even if the neuron does not respond or information is lost.
- Multiple tasks can be carried out in parallel by neural networks without affecting the system's performance.

Also doing the further research there were several factors that could make the ANN not a good machine learning algorithm. The **disadvantages** of using Neural Network based classification algorithm are;

- Neural networks require much more computing power/resources than the other machine learning algorithms.
- Neural networks require a huge dataset to train on.
- One of the major drawbacks of using neural networks is its black -box nature.

Although neural networks have few disadvantages their usefulness overshadows the drawbacks. Thus, in the coursework an Artificial Neural Network (ANN) is used over other machine learning algorithms for digit recognition tasks. To achieve that task, a dataset provided by MNIST is used to train and test the model which is built on python language using different machine learning libraries.

2.2. Review and Analysis of Existing Work in the Problem Domain:

There were several other researchers who used different classification algorithms to classify the handwritten digits with different accuracy. Let's observe existing work that was done in the field of digit recognition;

2.2.1. Solution 1: Handwritten digits recognition with decision tree classification: a machine learning approach

Author: Tsehay Admassu Assegie and Pramod Sekharan Nair

Description: Handwritten digits recognition is a form of machine learning in which a computer is taught to recognize handwritten numbers. A decision tree classification model is one way to accomplish this. A decision tree classification is a machine learning technique that employs established labels from previously known data sets to determine or predict the classes of future data sets with unknown labels. In this research, we employed a decision tree classification strategy to recognize handwritten digits using the standard Kaggle digits dataset which produced about 83.4 % accuracy (Assegie & Nair, 2019). The result of the proposed solution is below;

HandWritten Digits(0-9)	Accuracy of the Classifier (100%)
0	83.56
1	93.73
2	83.69
3	83.73
4	83.81
5	83.65
6	83.47
7	83.81
8	84.12
9	83.75

Figure 4 Decision Tree Accuracy Table (Assegie & Nair, 2019)

2.2.2. Solution 2: Handwritten Digit Recognition Application Based on Improved Naïve Bayes Method.

Author: Dianwei Chi

Project Description: There is a wide range of application scenarios of handwritten digit recognition systems. As handwritten digits have the characteristics of randomness and variability, it becomes difficult to obtain high accuracy in digit classification. The paper proposes a Bayesian Classification algorithm's improvement in digit recognition tasks. Though the paper primarily focuses on improving a specific Naïve Bayes Classification algorithm, its results could be used to compare with the proposed solution. The algorithm uses the MNIST dataset where it trains two thirds of the dataset and trains on the remaining one third. The proposed method gives at most classification accuracy to be 84.17 % at 278.59 seconds of prediction cost (Chi, 2020).

Prediction Cost	278.59 seconds (fastest)
Accuracy	84.17 % (best)

2.2.3. Solution 3: Handwritten Digit Recognition Using K-Nearest Neighbour (KNN) Classifier

Author: Ravi Babu, Y. Venkateswarlu and Aneel Kumar Chintha

Project Description: This work describes a new approach to off-line handwritten digit detection based on structural features that does not require thinning or size normalization. The major goal of this study is to present strategies for recognizing handwritten digits that are both efficient and trustworthy. To discover minimum distances, a Euclidean minimum distance criterion is utilized, and the digits are classified using a k-nearest neighbour classifier. The system is trained and tested using the MNIST database. A total of 5000 numerical images were used to test the suggested approach, with a recognition rate of 96.94 % (Babu, et al., 2014).

Digit	Test Images	Correctly not Classified	Correctly Classified	% Accuracy
0	425	15	410	96.47
1	635	13	622	97.95
2	585	16	569	97.26
3	532	13	519	97.56
4	550	15	535	97.27
5	434	11	423	97.47
6	435	17	418	96.09
7	491	21	470	95.72
8	443	19	424	95.71
9	470	13	457	97.23
Total	5000	153	4847	96.94

Figure 5 Accuracy for KNN based handwritten digit classifier (Babu, et al., 2014)

Comparison of the above pre-existing solutions with my solution i.e., a handwritten digit classifier using Multi-layer perceptron.

Features	Solution 1	Solution 2	Solution 3	My Solution
Algorithm Used	Decision Tree	Naïve Bayes	K-Nearest Neighbour	Multi-layer Perceptron
Classification Accuracy	83.4 %	84.17 %	96.94 %	98.04 %
Classification Time	Not Stated	278.59s	Not Stated	50s
User Experience	New classification only from the dataset.	New classification only from the dataset.	New classification only from the dataset.	A better user experience by coding in google collab.

3. Solution:

3.1. Explanation of proposed solution:

Handwritten Digit Recognition System using ANN:

Handwritten Digit recognition system is one of the most important fields to be explored in the field of machine learning. Automating the task of digit recognition has a lot of real-life application such as zip code detection (Cun, et al., 1990), number plate recognition (Prabhakar, et al., 2014), account number detection in banking (Srivastava, et al., 2018), numerical data recognition (e.g., price) from handwritten paper bill etc.

All of the above real-life problems can be solved fully or partially using a handwritten digit recognition system that is to be built in this project. Many different machine learning algorithms could be used for developing a classifier for handwritten digit recognition systems such as K-Nearest Neighbours (KNN), Random Forest, Naïve Bayes, Decision Tree, Multi-layer Perceptron etc. Among the different available classifiers, in the coursework, an artificial neural network called Multi-layer perceptron was used because of various reasons such as better accuracy, extension to other classification tasks, better at omitting out noise etc.

Thus, to solve the problem of recognizing handwritten digits, an artificial neural network called Multi-layer perceptron was used. Whenever there is a use of supervised classification algorithm, such as neural networks, availability of the labelled data is the most. Also, for the better accuracy, the bigger the dataset, the better the neural network learns.

After selecting which algorithm to use, a labelled dataset of numerical digits was searched and fortunately a pre labelled dataset called MNIST dataset was found. The MNIST dataset includes 60,000 training images of handwritten digits ranging from zero to nine, as well as 10,000 test images. As a result, the MNIST dataset comprises ten distinct classes where the handwritten digits are represented as a 28 by 28 matrix with grayscale pixel values in each cell (LeCun, et al., 1998).

After the dataset was found, the dataset was converted into a suitable format for easing the training task. As data is converted to suitable format, a neural network classifier was coded on which the training dataset was passed as an input and a classification model was obtained. The classification model's accuracy was tested using a different test dataset. Hence, the digit recognition system is built. The entire approach can be summarized as below:

1. **Collection of Dataset:** A pre labelled dataset called MNIST dataset was used.
2. **Conversion of Data into suitable format:** To ease the training, the dataset was converted to suitable format.
3. **Developing the Neural Network Classifier:** A neural network classifier was developed.
4. **Training the classifier using train dataset:** The classifier was trained using the train dataset.
5. **Testing the classifier using test dataset:** The accuracy of the classifier was checked using the test dataset.
6. **Optimizing the parameter:** Several parameters of the classifier were tuned for achieving better accuracy.
7. **Recognizing the handwritten digit:** The trained model is now used for detecting the handwritten digit.

3.2. Explanation of AI Algorithm Used:

Artificial Neural Networks (ANNs) are widely used today for verity of machine learning applications ranging classification, regression, clustering etc. The core of ANN is perceptron, which was developed for the first time by Frank Rosenblatt in 1958 (Rosenblatt, 1958) taking inspiration from the work of Warrant McCulloch and Walter Pitts (McCulloch & Pitts, 1943). A single perceptron or an artificial neuron takes several binary inputs x_1, x_2, \dots, x_n and produces a single binary output y .

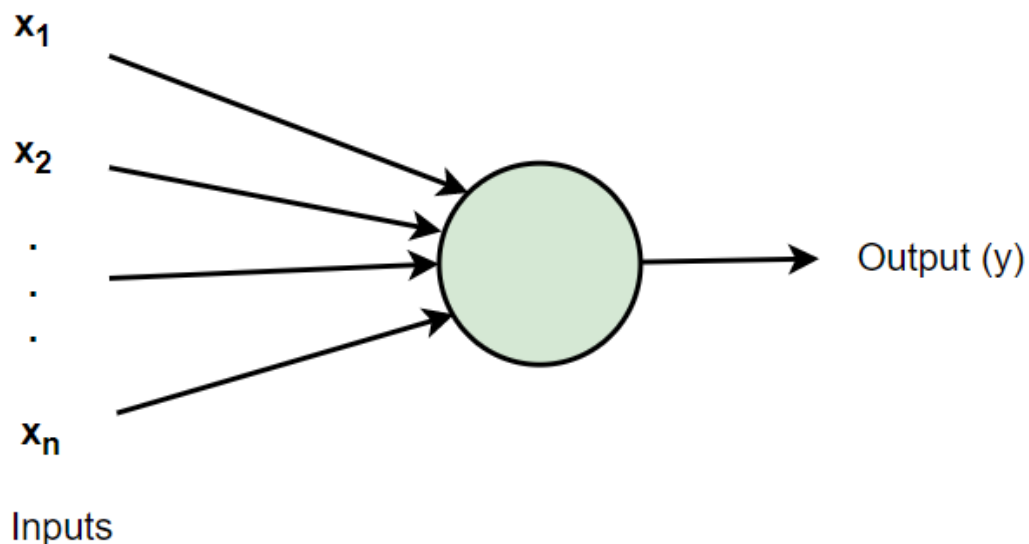


Figure 6 A single perceptron

In the single perceptron shown above Rosenblatt proposed a simple rule to compute the output. A real numbered value w_1, w_2, \dots, w_n called weight was introduced. Similar to biological neurons, the output of the perceptron is always binary i.e., 0 or 1. The output of the neuron is determined by what the weighted sum (sum of weighted inputs) is compared to the threshold value. The output is labelled '0' if the weighted sum is smaller than the threshold value and output is labelled '1' if the weighted sum is greater than the threshold value.

Mathematically, the threshold and weighted sum is depicted as;

$$\text{output}(y) = \begin{cases} 1 & \text{if } \left(\sum_{i=1}^n w_i x_i \right) \geq \text{threshold} \\ 0 & \text{if } \left(\sum_{i=1}^n w_i x_i \right) \leq \text{threshold} \end{cases}$$

Simplifying the above formula, we can write $\sum_{j=1}^n w_j x_j$ as the dot product of weight (w) and input space(x) i.e., $w \cdot x$ and we can also shift the threshold to the other side of the inequality and replace it with bias(b) which is equivalent to negative threshold.

$$\text{output}(y) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{if } w \cdot x + b \leq 0 \end{cases}$$

Giving output as 0 and 1 is not the best solution as a small change in a weight (or bias) causes a small change in output, then we could use this fact to modify the weights and biases to get our network to behave more in the manner we want. We solve that problem using an activated neuron where the output is not a 0 or 1 but activation function times the sum of weighted inputs a bias. Multiple activation functions can be used such as sigmoid, tanh or Rectified Linear Unit (ReLU) etc., The selection of activation function is one of the ways to tune the accuracy of the classifier. For simplicity let's take an example of the output of a sigmoid neuron;

$$\text{output}(y) = \frac{1}{1 + e^{-(w \cdot x + b)}}$$

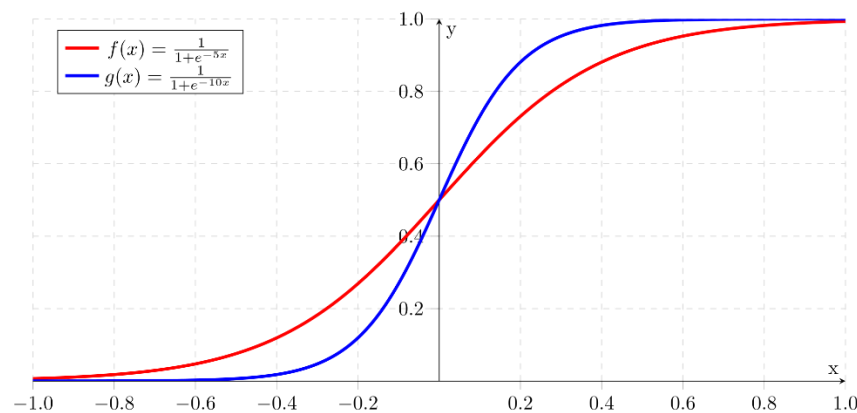


Figure 7 A sigmoid function

There are multiple neurons or perceptrons also called as nodes in multi-layer perceptrons but each node works similar to the single perceptron as described above. A combination of multiple nodes stacked into different layers is capable of making quite a subtle decision (Nielson, 2019). The first layer is the input layers, the last layer is the output layer and the layer between these two layers are hidden layers. Increasing the depth of the layers helps to improve the learning ability of the neural network. A simple pictorial representation of a multilayer neural network is below;

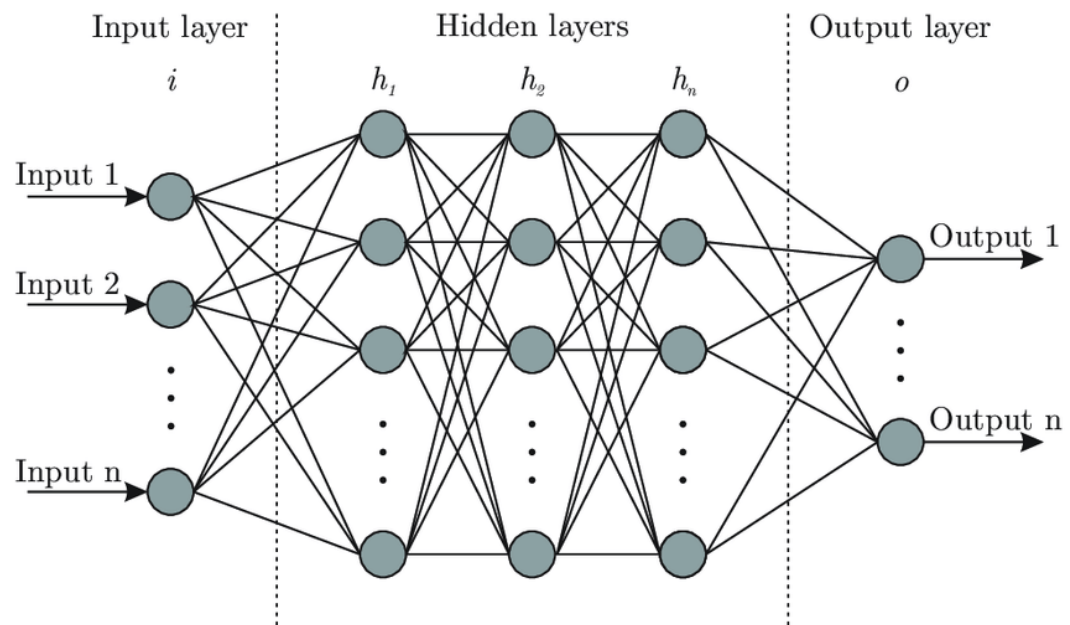


Figure 8 A Multi-layer Neural Network

With increasing layers, the number of nodes also increases. We don't want to compute the weights and bias manually so we have to automate it. We use a gradient descent algorithm to find the weight and bias that makes the cost function $C(w, b)$ as small as possible. Then the gradient descent algorithm picks out randomly chosen minibatch of training inputs and trans with those.

The stochastic gradient descent algorithm is represented mathematically as;

$$w_k \rightarrow w'_k = w_k - \frac{\eta}{m} \sum_i \frac{\partial C_{x_i}}{\partial w_k}$$

$$b_l \rightarrow b'_l = b_l - \frac{\eta}{m} \sum_i \frac{\partial C_{x_i}}{\partial b_l}$$

Where,

w_k and b_l denote the weights and bias in neural network,

w'_k and b'_l denote updated weights and bias

m is the size of the minibatch

η is the learning rate.

$\frac{\partial C_{x_i}}{\partial w_k}$ is the partial derivative of cost function C_{x_i} with respect to weights (w_k).

$\frac{\partial C_{x_i}}{\partial b_l}$ is the partial derivative of cost function C_{x_i} with respect to any weight (b_l).

The gradient descent approach can be used to train the weights and biases of neural networks, but we need a fast algorithm that can compute the cost function's gradients. Backpropagation is a learning algorithm used to compute a gradient descent with respect to weights. Desired outputs are compared to actual system outputs, and the systems are fine-tuned by altering connection weights to close the gap as much as possible (Vaughan, 2021). The weights are updated backwards, from output to input, which gives the algorithm its name i.e., back propagation. The backpropagation algorithm was first presented in the 1970s, but its significance wasn't fully realized until David Rumelhart, Geoffrey Hinton, and Ronald Williams published a seminal work in 1986 (Rumelhart, et al., 1989).

To compute $\partial C / \partial w_{ik}^l$, which means partial derivative of the cost with respect to w_{ik}^l i.e., weights connecting to the l^{th} layer of neurons, that is, the entry in the i^{th} row and k^{th} column, we need to introduce an intermediate quantity, δ_i^l i.e., error in the i^{th} neuron in the l^{th} layer. Similar is the case for bias. Thus, error of neuron i in layer l can be computed as;

$$\delta_i^l = \frac{\partial C}{\partial z_i^l}$$

Where, z_i^l is the weighted input to the activation function for neuron i in layer l , $z_i^l = w_{ik}^l \cdot a_k^{l-1} + b_i^l$ and a_k^{l-1} is the activation of neuron k in layer $l-1$.

There are four backpropagation equations that we make use in different layers of the neural network.

1. An equation for computing error in the outer layer L :

$$\delta_i^L = \frac{\partial C}{\partial a_i^L} \sigma'(z_i^L)$$

2. An equation for computing error in terms of error in the subsequent layer $l + 1$:

$$\delta^l = ((w^{l+1})^T \delta^{l+1} \odot \sigma'(z^l))$$

3. An equation for computing rate of change of the cost w.r.t any bias in the neural network:

$$\delta_i^l = \frac{\partial C}{\partial b_i^l}$$

4. An equation for computing rate of change of the cost w.r.t any weights in the neural network:

$$a_k^{l-1} \delta_i^l = \frac{\partial C}{\partial w_{ik}^l}$$

Now, let's wrap all the mathematical notation and write the algorithm of the neural network classifier that was used in the classification task in the next page.

Step 1: Start

Step 2: Input csv dataset file for training

Step 3: Assign random weights to all the neural connection

Step 4: Calculate the activation rate of neurons on hidden layers using input data and neural connections

Step 5: Find the activation rate of the output neurons using the activation rates of hidden neuron and connections to output

Step 6: Find the error rate at the output neuron and recalibrate all the connections between hidden neurons and output neuron

Step 7: Propagate down the error to hidden neuron using the weights and errors found at the output neuron

Step 8: Recalibrate the weights between the hidden and input neurons

Step 9: Loop on the process till the desired convergence is obtained

Step 10: Score the activation rate of the output neurons using the final connection weights

Step 11: Test the classifier

Step 12: Tune the parameters

Step 13: Develop the final model

Step 14: End

3.3. Pseudocode of the Solution:

The pseudocode of the desired solution is given below (Nelson, 2019):

IMPORT necessary libraries

INPUT a set of training examples x

ASSIGN random weight and bias

REPEAT

SET corresponding input activation $a^{x,1}$ *i.e. for first layer*

FOR ALL training example x

FOR EACH $l = 2, 3, \dots, L$

CALCULATE $z^{x,l} = w^l a^{x,l-1}$ and $a^{x,l} = \sigma(z^{x,l})$

END FOR

CALCULATE $\delta^{x,l} = \nabla_a C_x \odot \sigma'(z^{x,L})$

FOR EACH $l = L - 1, L - 2, \dots, 2$

CALCULATE $\delta^{x,l} = ((w^{l+1})^T \delta^{x,l+1} \odot \sigma'(z^{x,l}))$

END FOR

END FOR

FOR EACH $l = L, L - 1, \dots, 2$

UPDATE $w^l \rightarrow w^l - \frac{\eta}{m} \sum_x \delta^{x,l} (a^{x,l-1})^T$

UPDATE $b^l \rightarrow b^l - \frac{\eta}{m} \sum_x \delta^{x,l}$

END FOR

UNTIL achieving convergence

GET testing dataset

PREDICT the input of test data

CALCULATE accuracy of prediction

IF accuracy is less

TUNE parameters

END IF

ELSE

SAVE weights and bias for the model

DELIVER recognition system

END

3.4. Digit Recognition System Flowchart:

The flowchart of the digit recognition system that is to be developed is depicted below as;

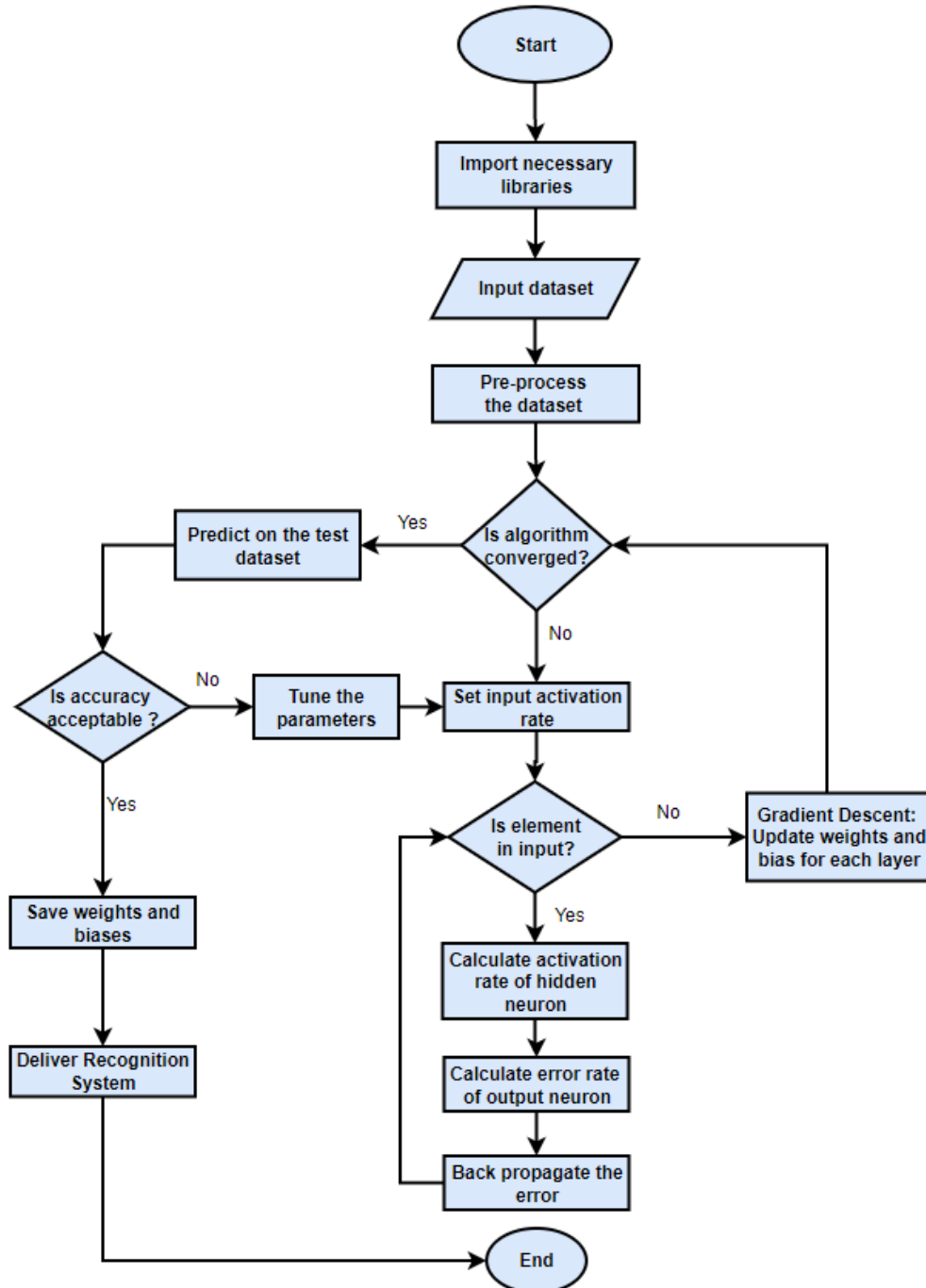


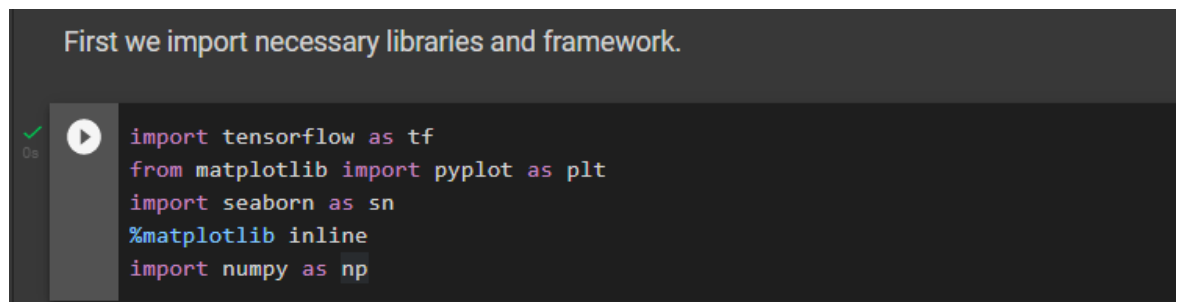
Figure 9 Flow chart of the solution

Description of flowchart: The above flowchart is the flow diagram of a multilayer perceptron with backpropagation and gradient descent algorithm in use. First, the system imports necessary libraries that are to be used in the system. After that dataset is given to the program as an input and the pre-processing task is done. Until the algorithm is converged, for each of the neurons, activation rate is calculated and specific for hidden and outer neurons and error is calculated and with the use of Adam optimizer (an updated gradient descent algorithm) algorithm, weights and biases are recalibrated. After convergence is achieved, the test dataset is used to calculate classification error and if accuracy is acceptable weights and bias is saved for future handwritten digit prediction otherwise, the tuning of parameters is performed and the weights are recalculated.

3.5. Explanation of the development process:

In the application, a handwritten digit classifier is developed using an artificial neural network. The most important task in developing a Neural Network based classifier is the availability of huge dataset to train the model. MNIST dataset which has 70000 labelled data was used. The MNIST dataset includes 60,000 training images of handwritten digits ranging from zero to nine, as well as 10,000 test images. After data is obtained, a neural network classifier was coded on which the training dataset was passed as an input and a classification model was obtained. The classification model's accuracy was tested using a different test dataset. Hence, the digit recognition system is built. The development process is described step-by step below:

Step1) Importing necessary modules and frameworks: This is one of the first thing we have to do to develop the system.

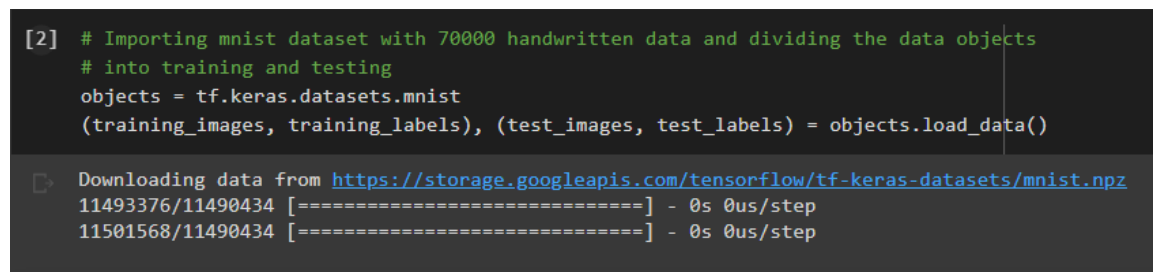


```
First we import necessary libraries and framework.

import tensorflow as tf
from matplotlib import pyplot as plt
import seaborn as sn
%matplotlib inline
import numpy as np
```

Figure 10 Importing modules

Step2) After importing the necessary modules, we have to import the dataset and then split the dataset into the training and testing. This is done automatically by keras.



```
[2] # Importing mnist dataset with 70000 handwritten data and dividing the data objects
# into training and testing
objects = tf.keras.datasets.mnist
(training_images, training_labels), (test_images, test_labels) = objects.load_data()
```

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz>
11493376/11490434 [=====] - 0s 0us/step
11501568/11490434 [=====] - 0s 0us/step

Figure 11 Splitting training-testing data

Step3) After getting data, we are going to normalize the data. After that we create the neural network model using TensorFlow. The first layer has 786 neurons, there are 128 hidden layers and we are using the Rectified Linear Unit (ReLU) activation function. The selected optimizer is Adam and we use sparse categorical cross entropy as the loss function.

```
# Getting the value from (0-255) to (0-1) :- Normalization
training_images = training_images / 255.0
test_images = test_images / 255.0

# Now lets create a Neural Nwtwork model
model = tf.keras.models.Sequential([tf.keras.layers.Flatten(input_shape=(28,28, 1)), # Fl
                                     tf.keras.layers.Dense(128, activation=tf.nn.relu), # 
                                     tf.keras.layers.Dense(10, activation=tf.nn.softmax)])

# Using optimization and loss function
# Adamoptimizer is modfied version of gradient descent algorithm.
# Used the crossentropy loss
# Used accuracy as model's measurement metric
model.compile(optimizer=tf.optimizers.Adam(),
              loss = 'sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

Figure 12 Developing the Neural Network Model

Step4) After building the model, the model is trained with training data that was already split and then the model's accuracy is tested by using the testing data.

```
# Training the model (42 s)
model.fit(training_images, training_labels, epochs=5)

# Testing the model
print(model.evaluate(test_images, test_labels))
```

Figure 13 Training and testing the model

To achieve the development task the following tools and technologies were used.

Tools and Technologies:

- **Google Colab:** Colab is a free cloud-based Jupyter notebook environment that is free to use. Most importantly, it doesn't require any setup, and the notebooks you create can be modified simultaneously by your team members, much like Google Docs projects. Many common machine learning libraries are supported by Colab and can be quickly loaded into your notebook. This is the product from Google Inc (Inc, 2022). Neural Networks models are CPU intensive to train, using Colab only requires us to have a good internet connection without worrying about the CPU usage.
- **TensorFlow:** TensorFlow is a machine learning and artificial intelligence software library that is free and open-source. It can be used for a variety of applications, but it focuses on deep neural network training and inference. The Google Brain team created TensorFlow for internal Google use in research and production (Metz, 2015). The underlying source code of the library is C++ but it can be integrated to be used with many other programming languages like python. IN the application, I have used TensorFlow to obtain the dataset, split it and then train the ANN model and test the accuracy.
- **NumPy:** NumPy is the most important Python package for scientific computing. Jim Hugunin created Numeric, the forerunner of NumPy. Numarray, a new package with some extra features, was also created. Travis Oliphant created the NumPy package in 2005 by combining the functionality of Numarray with the Numeric package (Peter, 2020). This open-source project has a large number of contributors. NumPy is written in python and C-language at its core and it is one of the most popular python libraries. In the application, I have used NumPy to operate with the array values of the output layer.

- **Matplotlib:** Matplotlib is a Python package that allows you to create static, animated, and interactive visualizations (matplotlib, 2021). It is one of the most popular visualization libraries in Python and is completely free to use. John D. Hunter was the creator of Matplotlib. It's had a thriving development community since then, and it's distributed under a BSD-style license. In the application, I have used Matplotlib to visualize the number.
- **Seaborn:** Seaborn is a popular statistical graph developing library in python (Seaborn, 2021). It is a Python data visualization package based on matplotlib that is tightly connected with pandas data structures. The core component of Seaborn is visualization, which aids in data exploration and comprehension. In the application, I have used seaborn to develop the confusion matrix of the result of the classification of the test data as predicted by the model and its actual level.

3.6. Achieved Results:

The Neural Network model is first trained using different parameters and once the model starts to give better result with selected parameter, we set the parameter to be final. The model's accuracy is measured from the already available labelled dataset. Weights and biases are continuously updated while training with 60,000 labelled digits. Once the value converges, we test its accuracy with test dataset. Then the model is saved and future classification is done using that model. The result of the model from training and testing accuracy to classification demo is shown below;

- **Training Accuracy:** The model bears different accuracy while training on the data. Weights and biases are continuously updated while training with 60,000 labelled digits. The following accuracy was observed in the application while training:

```
# Training the model (42 s)
model.fit(training_images, training_labels, epochs=5) # Achieved 99% accuracy on training example

Epoch 1/5
1875/1875 [=====] - 7s 4ms/step - loss: 0.0120 - accuracy: 0.9962
Epoch 2/5
1875/1875 [=====] - 5s 2ms/step - loss: 0.0116 - accuracy: 0.9962
Epoch 3/5
1875/1875 [=====] - 5s 2ms/step - loss: 0.0092 - accuracy: 0.9974
Epoch 4/5
1875/1875 [=====] - 4s 2ms/step - loss: 0.0087 - accuracy: 0.9974
Epoch 5/5
1875/1875 [=====] - 5s 2ms/step - loss: 0.0089 - accuracy: 0.9969
<keras.callbacks.History at 0x7fc6c2d50bd0>
```

Figure 14 Training accuracy

- **Testing Accuracy:** The real accuracy of the model is measured with the testing dataset. Already from the dataset 10000 labelled digit at random are selected as the test data. The accuracy of 98.04 % is obtained while testing the neural network model.

```
# Let's evaluate the model that is trained to measure the accuracy for the test example
print(model.evaluate(test_images, test_labels)) # Achieved 98% accuracy on testing example

313/313 [=====] - 0s 2ms/step - loss: 0.0813 - accuracy: 0.9804
[0.0812532901763916, 0.980400025844574]
```

Figure 15 Testing accuracy

- **Example Classifications:** Let's take some examples by taking random digit from the testing space and see what the model predicts the digit to be.
 - Classifying 10th data point from the test sample space: As shown in the image, the 10th data is the number '9' which is correctly classified by the model.



Figure 16 Classification 1

- Classifying 10000th data point from the test sample space: As shown in the image, the 10000th data is the number '6' which is correctly classified by the model.



Figure 17 Classification 2

- Classifying 5556th data point from the test sample space: As shown in the image, the 5556th data is the number '3' which is correctly classified by the model.



Figure 18 Classification 3

Now, let's observe how correctly the model predicts all the data points from the test sample by observing the result in the confusion matrix drawn from the given model. The x-axis shows the predicted number and y-axis shows the true label and the highlighted are the correct classifications and other are wrong ones.

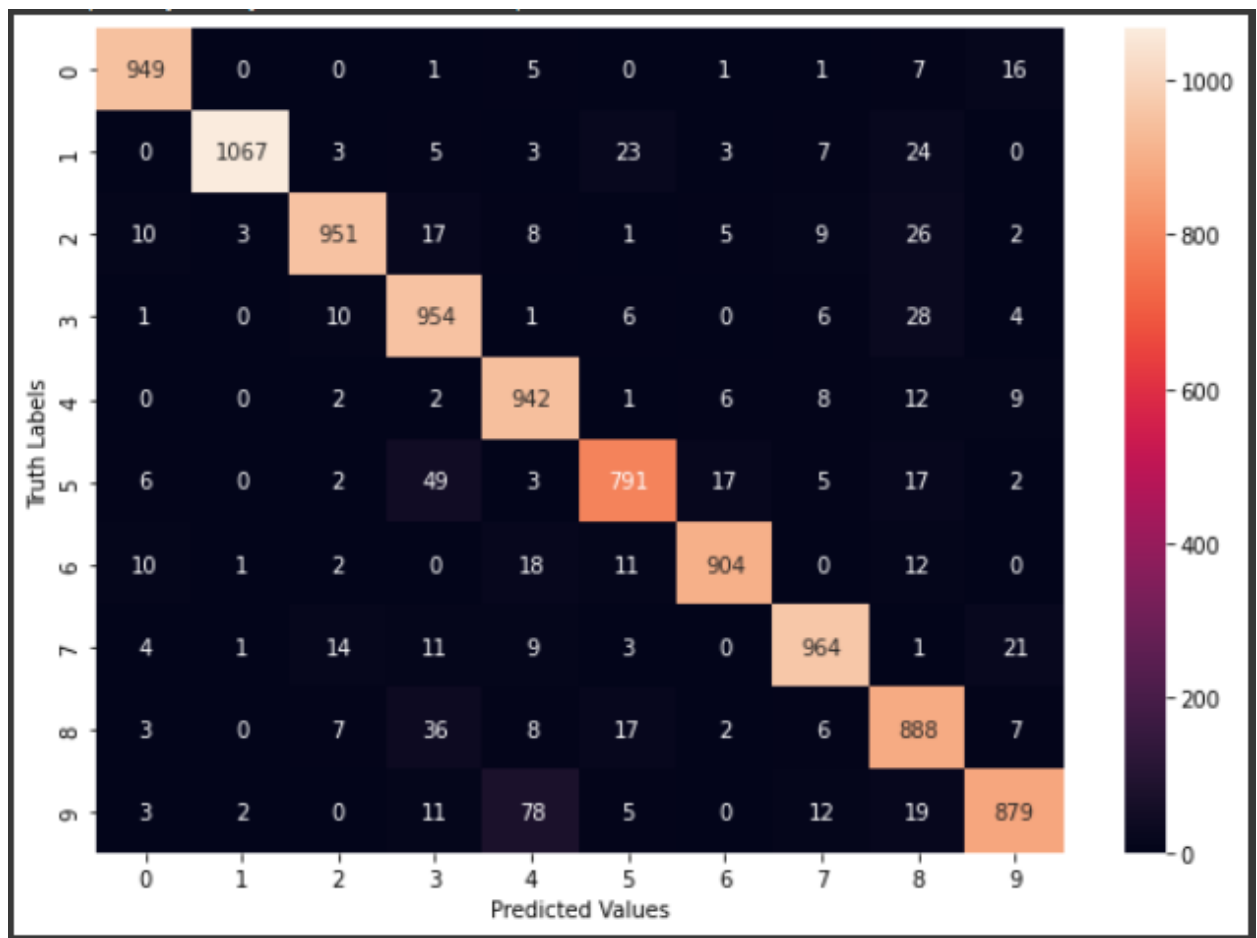


Figure 19 Confusion matrix of the classification

4. Conclusion:

4.1. Analysis of the work done:

A task was assigned to develop an Artificial Intelligent System by doing extensive research in that field. For the project, I develop an artificially intelligent handwritten digit recognition system using the Artificial Neural Network algorithm. The research work started with searching for a problem which is to be solved using Artificial Intelligence. From the research I found that handwritten digit recognition systems have a lot of real-life use cases and the algorithm developed for digit recognition could be extended to be used for text recognition too. For that I selected one of the popular classification algorithms called neural networks. Several other classification algorithms such as Naïve Bayes, K-Nearest Neighbour, Random Forest, Decision Tree etc. were in consideration but the neural network algorithm was finalized because it gets better with increasing dataset and is capable of ignoring noise better than other algorithms. After selecting the algorithm, an approval was sent for choosing the algorithm to out tutor Mr. Bibek Khanal and it was approved. After that, pseudocode and flowchart of the algorithm was developed to ease the coding part in future.

4.2. How the solution addresses real world problems:

Handwritten Digit recognition system is one of the most important fields to be explored in the field of machine learning. Automating the task of digit recognition has a lot of real-life application such as;

- Zip code detection can be done using the developed algorithm.
- With extended features, number plate recognition can also be performed.
- Account number detection in banking,
- Numerical data recognition (e.g., price) from handwritten paper bill etc.,
- Automating task of recognizing money by camera
- Automating task of numeric entries from form filled up by hand

4.3. Further work:

After developing the first report on what algorithm to use and what problem domain to solve, in this coursework, a model was developed using python and the classification accuracy was tested. The model that was developed was able to detect the handwritten digit with very high accuracy though it was not the best. The first task for future is to try to improve the accuracy of the model even more. And another task is to defend the development task that was done in this coursework in the form of PowerPoint presentation.

4.4. Limitation of the work:

Although the model works with high accuracy and there are many real-life use-cases of the algorithm, limitations still exist. The limitations come from the model itself as well as my own work too. Neural networks require much more computing power/resources than the other machine learning algorithms. Despite the best effort, these are the limitations of my work. Neural networks require a huge dataset to train on. These are the major limitations due to the selected models. No matter how strong the algorithm is and no matter how well the parameters are selected, the system is still prone to error specially when it is used with real-life data not the one from the dataset.

5. References:

Assegie, T. A. & Nair, P. S., 2019. Handwritten digits recognition with decision tree classification: a machine learning approach. *International Journal of Electrical and Computer Engineering (IJECE)*, 9(5), pp. 4446-4451.

Babu, R., Venkateswarlu, Y. & Chintha, A. K., 2014. Handwritten Digit Recognition Using K-Nearest Neighbour Classifier. *Proceedings - 2014 World Congress on Computing and Communication Technologies, WCCCT 2014*, Febraury, pp. 60-65.

Beniwal, H., 2018. *Handwritten Digit Recognition using Machine Learning*. [Online] Available at: <https://medium.com/@himanshubeniwal/handwritten-digit-recognition-using-machine-learning-ad30562a9b64> [Accessed 20 12 2021].

Bernard, S. A. S. & Heutte, L., 2007. *Using Random Forests for Handwritten Digit Recognition*. s.l., Ninth International Conference on Document Analysis and Recognition.

Chi, D., 2020. Handwritten Digit Recognition Application Based on Improved Naïve Bayes Method. *International Conference on Computer Vision, Image and Deep Learning (CVDIL)*, pp. 622-624.

Cun, Y. L. et al., 1990. *Handwritten zip code recognition with multilayer networks*. Atlantic City, Institute of Electrical and Electronics Engineers (IEEE).

Gill, N. S., 2021. *Artificial Neural Networks Applications and Algorithms*. [Online] Available at: <https://www.xenonstack.com/blog/artificial-neural-network-applications> [Accessed 20 12 2021].

Gopalakrishnan, V., Arun, R. & Abhirami, M. K., 2021. Handwritten Digit Recognition for Banking System. *International Journal of Engineering Research & Technology (IJERT)*, 9(5), pp. 313-314.

Inc, G., 2022. *Welcome To Colaboratory*. [Online] Available at: https://colab.research.google.com/?utm_source=scs-index [Accessed 25 01 2022].

LeCun, Y., Corinan, C. & Burges, J., 1998. *THE MNIST DATABASE of handwritten digits*. [Online]

Available at: <http://yann.lecun.com/exdb/mnist/>
[Accessed 20 12 2021].

Malekian, A. & Chitsaz, N., 2021. Concepts, procedures, and applications of artificial neural network models in streamflow forecasting. In: P. Sharma & D. Machiwal, eds. *Advances in Streamflow Forecasting - From Traditional to Modern Approaches*. s.l.:Elsevier Inc., pp. 115-147.

matplotlib, 2021. *Matplotlib: Visualization with Python*. [Online]
Available at: <https://matplotlib.org/>
[Accessed 25 12 2022].

McCarthy, J., 1987. Generality in artificial intelligence. *Communications of the ACM*, 30(12), pp. 1030-1035.

McCarthy, J., Minsky, M., Rochester, N. & Shannon, C., 1955. *A PROPOSAL FOR THE DARTMOUTH SUMMER RESEARCH PROJECT ON ARTIFICIAL INTELLIGENCE*, New Hampshire: Dartmouth.

McCulloch & Pitts, 1943. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Volume 5, pp. 115-133.

Metz, C., 2015. *Google Just Open Sourced TensorFlow, Its Artificial Intelligence Engine*. [Online]

Available at: <https://www.wired.com/2015/11/google-open-sources-its-artificial-intelligence-engine/>
[Accessed 25 12 2022].

Minsky, M., 1970. Form and Content in Computer Science. *Journal of the ACM*, 17(2), pp. 197-215.

Mintz, Y. & Brodie, R., 2019. Introduction to artificial intelligence in medicine. *Minimally Invasive Therapy & Allied Technologies*, 27 02, 28(2), pp. 73-81.

Mitchell, T., 1997. 2nd ed. New York: McGraw Hill Science.

Mitchell, T., 1997. Artificial Neural Network. In: T. Mitchell, ed. *Machine Learning*. New York: McGraw-Hill Science, pp. 81-127.

Mwandau, B. & Nyanchama, M., 2018. *Investigating Keystroke Dynamics as a Two-Factor Biometric Security*, Nairobi: Faculty of Information Technology Strathmore .

Nelson, M., 2019. *How the backpropagation algorithm works*. [Online] Available at: <http://neuralnetworksanddeeplearning.com/chap2.html> [Accessed 21 12 2021].

Nielson, M., 2019. *Using neural nets to recognize handwritten digits*. [Online] Available at: <http://neuralnetworksanddeeplearning.com/chap1.html> [Accessed 20 12 2021].

Norvig, P. & Stuart, R., 2020. *Artificial Intelligence a modern approach*. 4th ed. s.l.:Pearson.

Orysix, 2020. *Artificial intelligence in neurology: promising research and proven applications*. [Online] Available at: <https://www.quantib.com/blog/artificial-intelligence-neurology-promising-research-and-proven-application> [Accessed 19 12 2021].

Peter, M., 2020. *The Library of Numerical Python*. [Online] Available at: <https://blog.devgenius.io/the-library-of-numerical-python-3a4a7a352cfc> [Accessed 25 12 2022].

Prabhakar, P., Anupama, P. & Resmi, S. R., 2014. *Automatic vehicle number plate detection and recognition*. Kanyakumari, Institute of Electrical and Electronics Engineers (IEEE).

Rosenblatt, F., 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), pp. 386-408.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J., 1989. Learning representations by back-propagating errors. *Nature*, Volume 323, pp. 533-536.

Samuel, A., 1959. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, july, 3(3), pp. 210-229.

Scott, K., 2016. *Kim Scotts Slides*. [Online] Available at: https://www.mit.edu/~kimscott/slides/ArtificialNeuralNetworks_LEAD2011.pdf [Accessed 20 12 2021].

Seaborn, 2021. *An introduction to seaborn*. [Online] Available at: <https://seaborn.pydata.org/introduction.html> [Accessed 25 12 2022].

Shamim, S. M. et al., 2018. Handwritten Digit Recognition using Machine Learning Algorithms. *Global Journal of Computer Science and Technology: D*, 18(1).

Srivastava, S. et al., 2018. Optical Character Recognition on Bank Cheques Using 2D Convolution Neural Network. *Applications of Artificial Intelligence Techniques in Engineering*, Volume 697, pp. 589-596.

Taffese, W., 2020. *Data-Driven Method for Enhanced Corrosion Assessment of Reinforced Concrete Structures*. s.l.:s.n.

Vaughan, J., 2021. *backpropagation algorithm*. [Online] Available at: <https://www.techtarget.com/searchenterpriseai/definition/backpropagation-algorithm> [Accessed 20 12 2021].

Werbos, P. J., 1975. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. s.l.:Harvard University.

6. Appendix (Plagiarism Report):

The plagiarism report, after testing in the google classroom has been reported below with the screen shot and other statistics;

Flagged Content: 2%, Cited or Quoted content: 7%

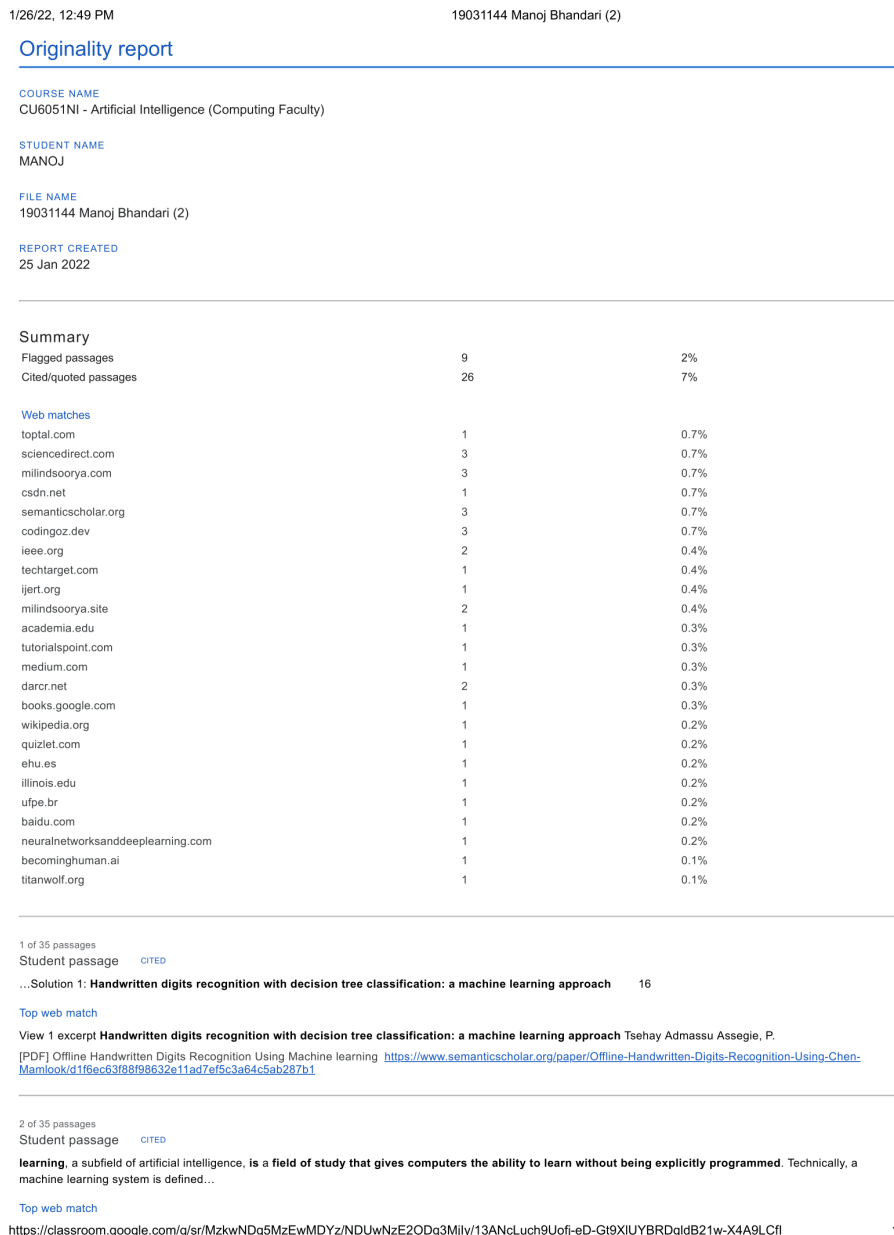


Figure 20 Originality Report

1/26/22, 12:49 PM

19031144 Manoj Bhandari (2)

There are some basic common threads, however, and the overarching theme is best summed up by this oft-quoted statement made by Arthur Samuel way back in 1959: "(Machine Learning is the) field of study..."

A Machine Learning Tutorial with Examples | Toptal <https://www.toptal.com/machine-learning/machine-learning-theory-an-introductory-primer>

3 of 35 passages

Student passage CITED

...Artificial Neural Networks (ANNs) are biologically **inspired** computational networks **that allow learning by example from representative data that describes a physical phenomenon or a decision process**

Top web match

ANN is a modeling technique **inspired** by the human nervous system **that allows learning by example from representative data that describes a physical phenomenon or a decision process**.

Artificial Neural Network - an overview | ScienceDirect Topics <https://www.sciencedirect.com/topics/earth-and-planetary-sciences/artificial-neural-network>

4 of 35 passages

Student passage CITED

Warren McCulloch and Walter Pitts pioneered artificial neural networks by developing a **computational model for neural networks based on threshold logic**

Top web match

The history of artificial neural networks (ANN) began with **Warren McCulloch and Walter Pitts** (1943) who created a **computational model for neural networks based on algorithms called threshold logic**.

History of artificial neural networks - Wikipedia https://en.wikipedia.org/wiki/History_of_artificial_neural_networks

5 of 35 passages

Student passage QUOTED

...(if threshold is met) and so on. The following **figure** compares **the biological neuron with an artificial neural network**.

Top web match

Figure 1 below shows the analogy between **the human biological neuron and an artificial neural network**. Figure 1.

From Perceptron to Deep Neural Nets | by Adi Chris - Becoming ... <https://becominghuman.ai/from-perceptron-to-deep-neural-nets-504b8ff616e>

6 of 35 passages

Student passage FLAGGED

...like tapped delay lines and Recurrent Neural Network (RNN). **Statistical ANN: Neural Network Models such as radial bias function model and generalized regression neural network model**

Top web match

Static ANN model is known as a multilayer perceptron neural network model, dynamic neural network models such as tapped delay lines and recurrent neural network models, and **statistical neural network**...

Artificial Neural Network - an overview | ScienceDirect Topics <https://www.sciencedirect.com/topics/earth-and-planetary-sciences/artificial-neural-network>

7 of 35 passages

Student passage CITED

...Neural Network is used to solve the problem domain. **Neural networks are among the most effective learning methods** now available for specific sorts of challenges, such as...

Top web match

For certain types of problems, such as learning to interpret complex real-world sensor data, artificial **neural networks are among the most effective learning methods** currently known.

Machine Learning - CIn UFPE <https://www.cin.ufpe.br/~cavmj/Machine%20-%20Learning%20-%20Tom%20Mitchell.pdf>

8 of 35 passages

Student passage CITED

The MNIST dataset includes **60,000 training images of handwritten digits ranging from zero to nine**, as well as **10,000 test images**. As a result, the MNIST dataset comprises ten distinct...

Top web match

The MNIST dataset contains **60,000 training images of handwritten digits from zero to nine** and **10,000 images** for testing. So, the MNIST dataset has 10 different classes.

<https://classroom.google.com/g/sr/MzkwNDg5MzEwMDYz/NDUwNzE2ODg3MjIy/13ANcLuch9Uofj-eD-Gt9XIUYBRDgldB21w-X4A9LCfi>

2/7

1/26/22, 12:49 PM

19031144 Manoj Bhandari (2)

Mnist handwritten digit classification using tensorflow - Milind Soorya <https://www.milindsoorya.com/blog/handwritten-digits-classification>

9 of 35 passages

Student passage CITED

...result, the MNIST dataset comprises ten distinct classes where **the handwritten digits are represented as a 28 by 28 matrix** with **grayscale pixel** values in each cell (LeCun, et al., 1998).

[Top web match](#)

So, the MNIST dataset has 10 different classes. **The handwritten digits images are represented as a 28×28 matrix** where each cell contains **grayscale pixel** value.

Mnist handwritten digit classification using tensorflow - Milind Soorya <https://milindsoorya.site/blog/handwritten-digits-classification>

10 of 35 passages

Student passage FLAGGED

Solution 1: Handwritten digits recognition with decision tree classification: a machine learning approach

[Top web match](#)

View 1 excerpt **Handwritten digits recognition with decision tree classification: a machine learning approach** Tsehay Admassu Assegie, P.

[PDF] Offline Handwritten Digits Recognition Using Machine learning <https://www.semanticscholar.org/paper/Offline-Handwritten-Digits-Recognition-Using-Chen-Mamlook/d1f6ec63f88f98632e11ad7ef5c3a64c5ab287b1>

11 of 35 passages

Student passage CITED

A decision tree classification is a machine learning technique that employs established labels from previously known data sets to determine or predict the classes of future data sets with unknown

[Top web match](#)

A decision tree classification is Revised Apr 22, 2019 **a machine learning approach that uses the predefined labels from the past** Accepted May 4, 2019 **known sets to determine or predict the classes of...**

(PDF) Handwritten digits recognition with decision tree classification https://www.academia.edu/42976792/Handwritten_digits_recognition_with_decision_tree_classification_a_machine_learning_approach

12 of 35 passages

Student passage FLAGGED

Solution 2: Handwritten Digit Recognition Application Based on Improved Naïve Bayes Method.

[Top web match](#)

Handwritten digit recognition application based on improved Naive Bayes method ieee conference publication ieee xplore Handwritten digit recognition application based on improved Naive Bayes method

Handwritten Digit Recognition Application Based on Improved Naive ... <https://ieeexplore.ieee.org/document/9270535>

13 of 35 passages

Student passage CITED

...of application scenarios of handwritten digit recognition systems. As **handwritten digits have the characteristics of randomness and variability**, it becomes **difficult to obtain high accuracy**

[Top web match](#)

Abstract: Handwritten digit recognition has a wide range of application scenarios, but because **handwritten digits have the characteristics of randomness and great variability**, it is often **difficult to...**

Handwritten Digit Recognition Application Based on Improved Naive ... <https://ieeexplore.ieee.org/document/9270535>

14 of 35 passages

Student passage CITED

Project Description: This work describes **a new approach to off-line handwritten digit detection based on structural features** that does **not** require **thinning** or **size normalization**

[Top web match](#)

A new approach to off-line handwritten digit recognition based on structural features which is **not** required **thinning** operation and **size normalization** technique is presented, to provide efficient and...

[PDF] Offline Handwritten Digits Recognition Using Machine learning <https://www.semanticscholar.org/paper/Offline-Handwritten-Digits-Recognition-Using-Chen-Mamlook/d1f6ec63f88f98632e11ad7ef5c3a64c5ab287b1>

<https://classroom.google.com/g/sr/MzkwNDg5MzEwMDYz/NDUwNzE2ODg3Mjly/13ANcLuch9Uofj-eD-Gt9XIUYBRDgldB21w-X4A9LCfl>

3/7

1/26/22, 12:49 PM

19031144 Manoj Bhandari (2)

15 of 35 passages

Student passage CITED

...a pre labelled dataset called MNIST dataset was found. **The MNIST dataset includes 60,000 training images of handwritten digits ranging from zero to nine, as well as 10,000 test images**

Top web match

The MNIST dataset contains 60,000 training images of handwritten digits from zero to nine and 10,000 images for testing. So, the MNIST dataset has 10 different classes.

Mnist handwritten digit classification using tensorflow - Milind Soorya <https://www.milindsoorya.com/blog/handwritten-digits-classification>

16 of 35 passages

Student passage CITED

...result, the MNIST dataset comprises ten distinct classes where **the handwritten digits are represented as a 28 by 28 matrix with grayscale pixel values in each cell** (LeCun, et al., 1998).

Top web match

So, the MNIST dataset has 10 different classes. **The handwritten digits images are represented as a 28×28 matrix** where each cell contains **grayscale pixel** value.

Mnist handwritten digit classification using tensorflow - Milind Soorya <https://milindsoorya.site/blog/handwritten-digits-classification>

17 of 35 passages

Student passage CITED

...McCulloch and Walter Pitts (McCulloch & Pitts, 1943). **A single perceptron or an artificial neuron takes several binary inputs and produces a single binary output.**

Top web match

How do perceptrons work? **A perceptron takes several binary inputs and produces a single binary output.** Rosenblatt proposed a simple rule to compute the output.

NN&DL - Using neural nets to recognize handwritten digits ... - Quizlet <https://quizlet.com/207083071/nndl-using-neural-nets-to-recognize-handwritten-digits-flash-cards/>

18 of 35 passages

Student passage QUOTED

is labelled '0' if the weighted sum is smaller than the threshold value and output is labelled '1' if the weighted sum is greater than the threshold value.

Top web match

A perceptron is a simple classifier that takes **the weighted sum of the D input feature values (along with an additional constant input value) and outputs + 1 for yes if the result of the weighted sum...**

Perceptron - an overview | ScienceDirect Topics <https://www.sciencedirect.com/topics/mathematics/perceptron>

19 of 35 passages

Student passage QUOTED

dot product of weight (w) and input space(x) i.e., and we can also shift the threshold to the other side of the inequality and replace it with bias(b) which is equivalent to negative threshold. ...

Top web match

Let **dot product** $w \cdot x = \sum w_j x_j$, and move **the threshold to the other side of the inequality, and to replace it by what's known as the perceptron's bias, $b = -\text{threshold}$** , the perceptron rule can be...

Using neural nets to recognize handwritten digits - CSDN博客 https://blog.csdn.net/wej_chao_cheng/article/details/69257943

20 of 35 passages

Student passage CITED

...capable of making quite a subtle decision (Nielson, 2019). **The first layer is the input layers, the last layer is the output layer and the layer between these two layers are hidden layers**

Top web match

The first layer represents the input layer, the last layer represents the output layer, and the layers between these two are the hidden layers.

FUNDAMENTAL OF SOFT COMPUTING: Theory, Concepts and Methods of ... <https://books.google.com/books?id=A41jDwAAQBAJ&pg=PA30&lpg=PA30&dq=the+first+layer+is+the+input+layers+the+last+layer+is+the+output+layer+and+the+layer+between+these+two+layers+are+hidden&hl=en>

<https://classroom.google.com/g/sr/MzkwNDg5MzEwMDYz/NDUwNzE2ODg3Mjly/13ANcLuch9Uofj-eD-Gt9XIUYBRDgldB21w-X4A9LCfl>

4/7

1/26/22, 12:49 PM

19031144 Manoj Bhandari (2)

21 of 35 passages

Student passage QUOTED

...C (w, b) as small as possible. Then the **gradient descent** algorithm picks **out randomly chosen minibatch of training inputs and** trans **with** those.

[Top web match](#)

Stochastic **gradient descent** in neural networks: Pick **out a randomly chosen minibatch of training inputs and** train **with** them; then pick out another minibatch, until inputs exhausted—complete an epoch...

Not Bayesian networks - Mining Latent Entity Structures http://hanj.cs.illinois.edu/cs412/bk3_slides/09ClassAdvanced.pptx

22 of 35 passages

Student passage CITED

is the **partial derivative of cost function** with respect to weights ().

[Top web match](#)

Derivation of **partial derivative of cost function with respect to weights** in backpropagation algorithm ... I am studying Machine Learning from Andrew Ng's Machine ...

Hi Anonymous - TitanWolf <https://www.titanwolf.org/Network/q/fe18d02-373b-4bcc-973d-9cd745c2169e/x>

23 of 35 passages

Student passage CITED

is the **partial derivative of cost function** with respect to any weight ().

[Top web match](#)

At the heart of backpropagation is an expression for the **partial derivative** $\partial C / \partial w$ of the **cost function C with respect to any weight w** (or bias b) in the network.

Neural network and deep learning-chapter-2 - 百度文库 <https://wenku.baidu.com/view/7200dcf9a21614791611282c.html?re=view>

24 of 35 passages

Student passage CITED

Backpropagation is a **learning algorithm** used to compute a **gradient descent with respect to weights**. Desired outputs are compared to actual **system outputs**, and the **systems** are fine-tuned by altering ...

[Top web match](#)

Artificial neural networks use **backpropagation** as a **learning algorithm** to compute a **gradient descent with respect to weights**. Desired outputs are compared to achieved **system outputs**, and then the...

What is backpropagation algorithm? - Definition from WhatIs.com <https://www.techtarget.com/searchenterpriseai/definition/backpropagation-algorithm>

25 of 35 passages

Student passage CITED

The **backpropagation algorithm** was first presented in the **1970s**, but its significance wasn't fully realized until David Rumelhart, Geoffrey Hinton, and Ronald Williams

[Top web match](#)

The **backpropagation algorithm** was originally introduced in the **1970s**, but its importance wasn't fully appreciated until a famous 1986 paper by David Rumelhart, Geoffrey Hinton, and Ronald Williams.

Function of the neuron - Medium <https://medium.com/coinmonks/function-of-the-neuron-1f34af053e1d>

26 of 35 passages

Student passage QUOTED

To compute $\frac{\partial C}{\partial w_{lk}}$, which means partial derivative of the cost with respect to i.e., **weights connecting to the lth layer of neurons, that is, the entry in the lth row and kth column**

[Top web match](#)

The entries of the weight matrix are just the **weights connecting to the lth layer of neurons, that is, the entry in the jth row and kth column** is w_{lk} .

Image-based Family Verification in the wild - ADDI [https://addi.ehu.es/bitstream/handle/10810/22629/image-based-family%20\(23\).pdf?sequence=1](https://addi.ehu.es/bitstream/handle/10810/22629/image-based-family%20(23).pdf?sequence=1)

27 of 35 passages

Student passage QUOTED

<https://classroom.google.com/g/sr/MzkwNDg5MzEwMDYz/NDUwNzE2ODg3Mjly/13ANcLuch9Uofj-eD-Gt9XIUYBRDgldB21w-X4A9LCfI>

5/7

1/26/22, 12:49 PM

19031144 Manoj Bhandari (2)

Where, z_i is the weighted input to the activation function for neuron i in layer l , and a_i is the activation of neuron i ...

[Top web match](#)

It's also worth noting that z_i has components $z_i = \sum_k w_{ki} a_k + b_i$, that is, z_i is just the weighted input to the activation function for neuron i in layer l .

How the backpropagation algorithm works - Neural networks and ... <http://neuralnetworksanddeeplearning.com/chap2.html>

28 of 35 passages

Student passage FLAGGED

An equation for computing rate of change of the cost w.r.t any bias in the neural network

[Top web match](#)

An equation for the rate of change of the cost with respect to any bias in the network: In particular: $\partial C / \partial b_{lj} = \delta_{lj}$.

Dissecting Backpropagation – Welcome to DARCR <http://darcr.net/index.php/2017/11/22/backpropagation-demystified/>

29 of 35 passages

Student passage FLAGGED

An equation for computing rate of change of the cost w.r.t any weights in the neural network

[Top web match](#)

An equation for the rate of change of the cost with respect to any bias in the network: In particular: $\partial C / \partial b_{lj} = \delta_{lj}$.

Dissecting Backpropagation – Welcome to DARCR <http://darcr.net/index.php/2017/11/22/backpropagation-demystified/>

30 of 35 passages

Student passage QUOTED

...MNIST dataset which has 70000 labelled data was used. **The MNIST dataset includes 60,000 training images of handwritten digits ranging from zero to nine, as well as 10,000 test images**

[Top web match](#)

The MNIST dataset contains 60,000 training images of handwritten digits from zero to nine and 10,000 images for testing. So, the MNIST dataset has 10 different classes.

Mnist handwritten digit classification using tensorflow - Milind Soorya <https://www.milindsoorya.com/blog/handwritten-digits-classification>

31 of 35 passages

Student passage FLAGGED

Most importantly, it doesn't require any setup, and the notebooks you create can be modified simultaneously by your team members, much like Google Docs

[Top web match](#)

Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by your team members - just the way you edit documents in Google Docs. Colab supports many...

What is Google Colab? - Tutorialspoint https://www.tutorialspoint.com/google_colab/what_is_google_colab.htm

32 of 35 passages

Student passage FLAGGED

...is the most important Python package for scientific computing. **Jim Hugunin created Numeric, the forerunner of NumPy. Numarray, a new package with some extra features, was also created**

[Top web match](#)

Jim Hugunin created Numeric, the forerunner of NumPy. Numarray, a new package with some extra features, was also created.

What is Numpy in Python | Python Numpy in Data Science <https://codingoz.dev/what-is-numpy-in-python-python-numpy-in-data-science/>

33 of 35 passages

Student passage CITED

...new package with some extra features, was also created. **Travis Oliphant created the NumPy package in 2005 by combining the functionality of Numarray with the Numeric**

[Top web match](#)

Numarray, a new package with some extra features, was also created. **Travis Oliphant created the NumPy package in 2005 by combining the functionality of Numarray with the Numeric package.**

<https://classroom.google.com/g/sr/MzkwNDg5MzEwMDYz/NDUwNzE2ODg3MjIy/13ANcLuch9Uofj-eD-Gt9XIUYBRDgldB21w-X4A9LCfI>

6/7

1/26/22, 12:49 PM

19031144 Manoj Bhandari (2)

What is Numpy in Python | Python Numpy in Data Science <https://codingoz.dev/what-is-numpy-in-python-python-numpy-in-data-science/>

34 of 35 passages

Student passage FLAGGED

...functionality of Numarray with the Numeric package (Peter, 2020). **This open-source project has a large number of contributors.** NumPy is written in python and C —language at its...

[Top web match](#)

Travis Oliphant created the NumPy package in 2005 by combining the functionality of Numarray with the Numeric package. **This open-source project has a large number of contributors.**

What is Numpy in Python | Python Numpy in Data Science <https://codingoz.dev/what-is-numpy-in-python-python-numpy-in-data-science/>

35 of 35 passages

Student passage FLAGGED

is a Python data visualization package based on matplotlib that is tightly connected with pandas data structures. The core component of Seaborn is visualization, which aids in data exploration and...

[Top web match](#)

Seaborn: Seaborn **is a Python data visualisation package based on matplotlib that is tightly connected with pandas data structures. The core component of Seaborn is visualisation, which aids in data...**

Data Analysis using Python - IJERT <https://www.ijert.org/data-analysis-using-python>
