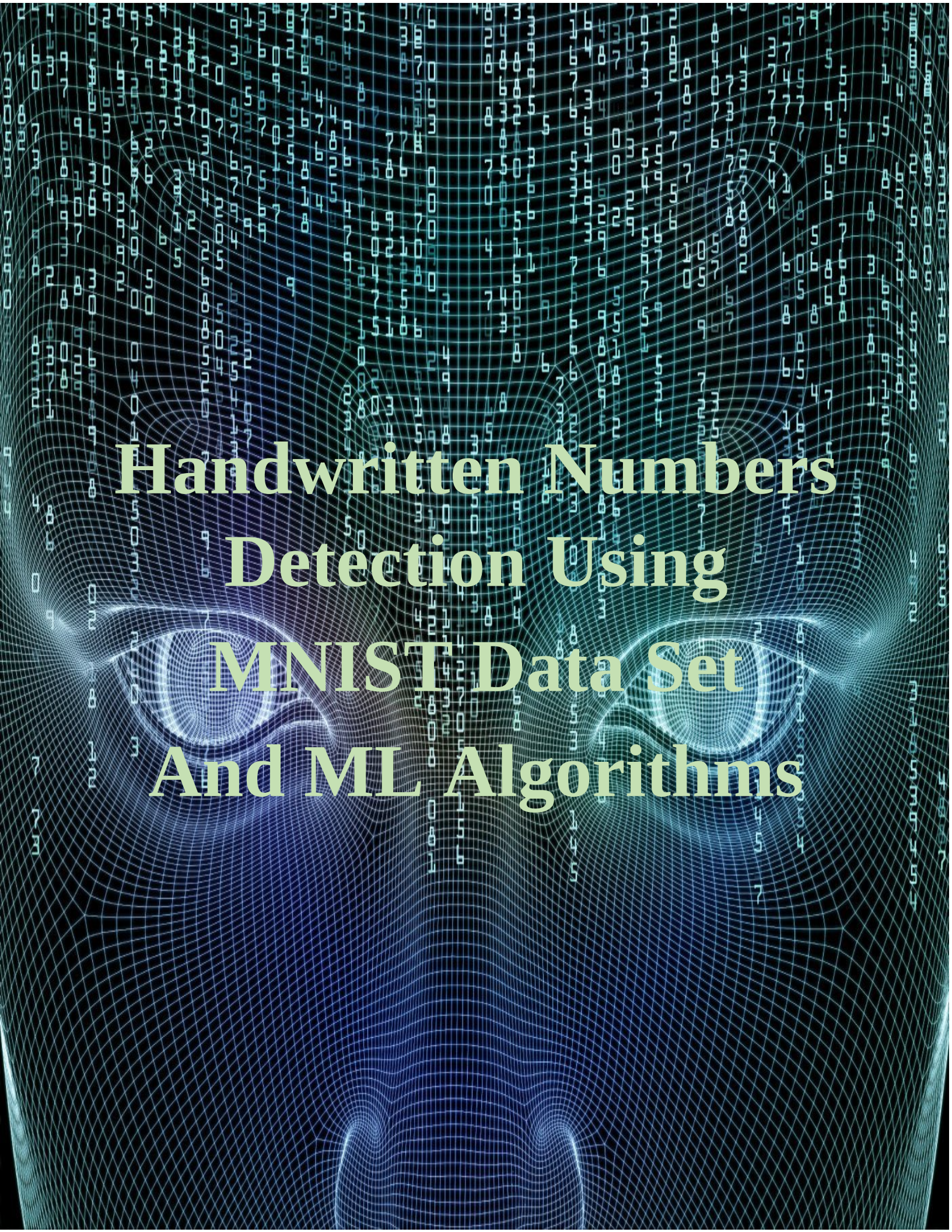

Handwritten Numbers
Detection Using
MNIST Data Set
And ML Algorithms

Handwritten Numbers Detection Using MNIST Data Set And ML Algorithms

Supervisor:

Dr. Moataz Elsaban

By:

Names	IDS
Youssef Sayed Ahmed	20180342
Ahmed Mostafa Abd El-Karim	20180031
Mahmoud Essam El-Din El-Masry	20180359

Table of Contents

Project Motivation

Handwriting recognition is the ability of a machine to receive and interpret handwritten input from multiple sources like paper documents, photographs, touch screen devices etc. Recognition of handwritten and machine characters is an emerging area of research and finds extensive applications in banks, offices and industries. The main aim of this project is to detect handwritten numbers using MNIST dataset as train data set.

The MNIST dataset is a large dataset of handwritten digits that is commonly used for training various image processing systems. The database is also widely used for training and testing in the field of machine learning.

This dataset contains 70000 images from different sources to a handwritten number.

So, this project is very useful for understanding machine learning classifiers for beginners.

Problem Definition

Till now we have 70000 of labeled handwritten images so we need to use it to classify unknown handwritten numbers from 0 to 9 based on this dataset.

Which we use to train some machine learning models like support vector machine, k-nearest neighbors, Artificial neural network and convolutional neural network.

After solving this problem, we can classify unknown images or patterns to identify the number inside picture.

Performance Measurement

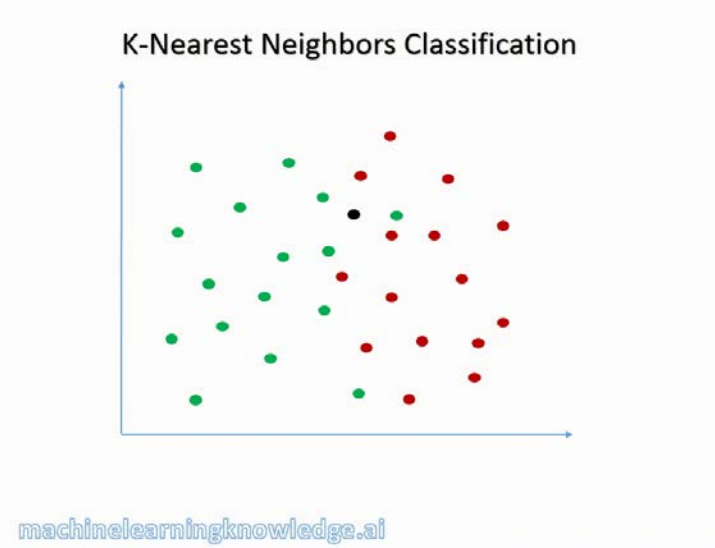
We are going to use MNIST dataset to make four models as we mentioned before and compare their accuracy to choose the best model which we can use its weights to identify unknown handwritten numbers in our applications. By splitting data into 60000 train data and 10000 test data then apply histogram of oriented gradients (HOG) to get best features from images to use in KNN, SVM and ANN.

First: k-nearest neighbor`s model

1. Model Definition

KNN: it is a non-parametric and lazy algorithm which predicts data by comparing distance between k nearest points by computing some mathematical functions as Euclidian distance rule and choose the maximum number of smallest distances to points of class and decides unknown point belongs to this class about these distances.

(Note: k is the number of nearest points to unknown point)



2. Model Refinement

We had changed parameter to improve our model accuracy. We had tried to change distance parameters between Euclidian distance and Manhattan distance. We got Euclidian is better than Manhattan so we decide to use Euclidian distance in our model.

We used some different k to get higher accuracy as k = 10, 15, 20 and set k as default. When k = 10 we got accuracy with 96.619%, when k = 15, accuracy = 96.47%, when k = 20, accuracy = 96.39% and when k was default, accuracy = 96.5%

3. Model Testing and Evaluation

We tested our model with different parameter as we mentioned before to get the best accuracy so we will see the confusion matrix of these different parameters to decide which parameters are the best

R2_Score

K = 10	K = 15	K = 20	K = default
92.56785449858823	91.84992260002439	91.5255414099756	92.24347330853945

1- K = 10

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	974	2	0	0	0	0	3	1	0	0
1	0	1129	3	0	0	0	2	1	0	0
2	15	3	991	5	0	0	1	13	4	0
3	2	2	5	975	1	6	0	5	12	2
4	4	13	2	0	927	0	7	1	1	27
5	5	1	2	12	1	851	9	1	8	2
6	6	6	1	0	0	1	941	0	3	0
7	0	11	4	1	2	0	0	982	2	26
8	15	4	4	6	1	6	4	4	916	14
9	4	6	1	7	2	0	0	8	5	976

2- K = 15

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	970	2	0	1	0	0	6	1	0	0
1	0	1130	3	0	0	0	2	0	0	0
2	13	5	981	6	0	0	2	18	4	0
3	2	3	4	975	0	5	0	5	12	4
4	3	14	0	0	921	0	7	1	1	35
5	4	2	1	14	2	850	10	0	6	3
6	5	6	0	0	0	1	944	0	2	0
7	0	14	3	1	3	0	0	976	0	31
8	16	3	4	7	1	5	3	4	919	12
9	4	7	2	6	2	0	0	5	5	978

3- K = 20

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	971	2	0	0	0	0	6	1	0	0
1	0	1129	3	0	0	0	3	0	0	0
2	14	6	976	6	0	0	2	18	9	1
3	2	3	1	979	0	5	0	5	11	4
4	3	14	0	0	916	0	8	2	1	38
5	4	2	1	15	1	848	10	1	7	3
6	5	6	0	0	0	1	944	0	2	0
7	0	13	3	0	2	0	0	977	1	32
8	18	2	3	7	1	5	5	3	920	10
9	4	7	1	6	2	0	0	6	4	979

4- K = default

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	974	2	0	0	0	0	4	3	0	0
1	0	1129	4	0	0	0	1	1	0	0
2	10	2	996	4	0	0	2	14	4	0
3	1	3	6	963	1	9	1	6	17	3
4	3	11	1	0	930	0	8	1	1	27
5	3	1	2	16	1	849	10	1	7	3
6	5	5	1	0	0	3	943	0	1	0
7	1	10	5	1	5	0	0	978	1	27
8	16	3	4	7	1	7	3	3	914	16
9	4	6	1	8	3	4	0	7	2	974

Precision & Recall

Classes	K = 10		K = 15		K = 20		K = default	
	precision	**Recall**	**precision**	**Recall**	**precision**	**Recall**	**precision**	**Recall**
0	95.02439024	99.3877551	95.37856441	98.97959	95.10284035	99.08163265	95.77187807	99.3877551
1	95.92183517	99.4713656	95.27824621	99.55947137	95.35472973	99.47136564	96.33105802	99.47136564
2	97.82823297	96.02713178	98.3016983	95.34883721	98.7854251	94.57364341	97.64705882	96.51162791
3	96.91848907	96.53465347	96.53465347	96.53465347	96.64363277	96.93069307	96.3963964	95.34653465
4	99.25053533	94.39918534	99.13885899	93.78818737	99.34924078	93.2790224	98.83103082	94.70468432
5	98.49537037	95.40358744	98.7224158	95.29147982	98.71944121	95.06726457	97.36238532	95.1793722
6	97.31127198	98.22546973	96.91991786	98.53862213	96.52351738	98.53862213	97.01646091	98.434238
7	96.65354331	95.52529183	96.63366337	94.94163424	96.44619941	95.03891051	96.73590504	95.13618677
8	96.31966351	94.04517454	96.83877766	94.35318275	96.33507853	94.45585216	96.61733615	93.83983573
9	93.21872015	96.72943508	92.00376294	96.92765114	91.75257732	97.02675917	92.76190476	96.53121903

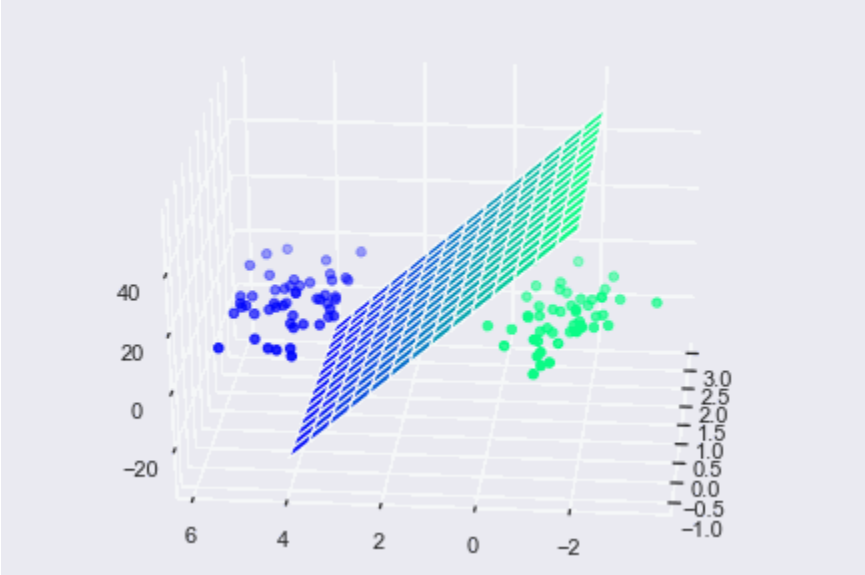
Due to confusion matrices, precision, recall, and r2_score we get the best accuracy, best matrix, precision, recall, and r2_score when k = 10 and distance is Euclidian are the best parameters.

Second: Support Vector Machine Model

1. Model Definition

SVM: it is a machine learning algorithm which can use in classifications and regression problems. It can use with continuous and discrete data.

It generates a hyperplane to separate different classes and makes some iterations to minimize errors and get the optimal hyperplane by maximize the distance between hyperplane and classes.



2. Model Refinement

There are some functions which are called kernels can be used to improve our model as Linear kernel, Polynomial Kernel and finally Radial Basis Kernel.

We decide to use one of these kernels by data shape.

In our model, the three kernels give same accuracy by 98.61% because our data is linear and these kernels try to shape data into linear shape.

3. Model Testing and Evaluation

We had tested the three kernels and find the three kernels gave same accuracy as we mentioned before and we will see the three-confusion matrix, precision, recall, and r2_score of three kernels.

R2_Score

K = Linear	K = Poly	K = RBF
96.93865251891464	96.93865251891464	96.93865251891464

1- K = Linear

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	978	0	0	0	0	0	1	0	1	0
1	0	1130	3	1	0	0	1	0	0	0
2	1	2	1020	0	1	0	0	6	2	0
3	0	1	2	994	0	5	0	2	5	1
4	0	0	1	0	970	0	0	0	3	8
5	2	0	0	7	0	876	3	0	3	1
6	3	2	0	0	1	2	947	0	3	0
7	0	4	6	2	4	0	0	1007	2	3
8	1	1	2	5	1	1	0	3	956	4
9	0	4	1	5	8	1	0	2	5	983

2- K = poly

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	978	0	0	0	0	0	1	0	1	0
1	0	1130	3	1	0	0	1	0	0	0
2	1	2	1020	0	1	0	0	6	2	0
3	0	1	2	994	0	5	0	2	5	1
4	0	0	1	0	970	0	0	0	3	8
5	2	0	0	7	0	876	3	0	3	1
6	3	2	0	0	1	2	947	0	3	0
7	0	4	6	2	4	0	0	1007	2	3
8	1	1	2	5	1	1	0	3	956	4
9	0	4	1	5	8	1	0	2	5	983

3- K = RBF

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	978	0	0	0	0	0	1	0	1	0
1	0	1130	3	1	0	0	1	0	0	0
2	1	2	1020	0	1	0	0	6	2	0
3	0	1	2	994	0	5	0	2	5	1
4	0	0	1	0	970	0	0	0	3	8
5	2	0	0	7	0	876	3	0	3	1
6	3	2	0	0	1	2	947	0	3	0
7	0	4	6	2	4	0	0	1007	2	3
8	1	1	2	5	1	1	0	3	956	4
9	0	4	1	5	8	1	0	2	5	983

Precision & Recall

Classes	K = Linear		K = Poly		K = RBF	
	precision	Recall	precision	Recall	precision	Recall
0	99.2893401	99.79591837	99.2893401	99.79591837	99.2893401	99.79591837
1	98.77622378	99.55947137	98.77622378	99.55947137	98.77622378	99.55947137
2	98.55072464	98.8372093	98.55072464	98.8372093	98.55072464	98.8372093
3	98.02761341	98.41584158	98.02761341	98.41584158	98.02761341	98.41584158
4	98.47715736	98.77800407	98.47715736	98.77800407	98.47715736	98.77800407
5	98.98305085	98.20627803	98.98305085	98.20627803	98.98305085	98.20627803
6	99.47478992	98.85177453	99.47478992	98.85177453	99.47478992	98.85177453
7	98.7254902	97.95719844	98.7254902	97.95719844	98.7254902	97.95719844
8	97.55102041	98.15195072	97.55102041	98.15195072	97.55102041	98.15195072
9	98.3	97.42319128	98.3	97.42319128	98.3	97.42319128

Due to confusion matrices, precision, recall, and r2_score we get they have same accuracy and confusion matrix there is no different between them.

Third: Artificial Neural Network

1. Model Definition

ANN: it is a machine learning algorithm designed to simulate human brain and let machine to learn and detect patterns.

It consists of many units called neurons and every set of these neurons consist a layer.

Layers are connected to each other and every neuron in layer connected to all neurons in other layer.

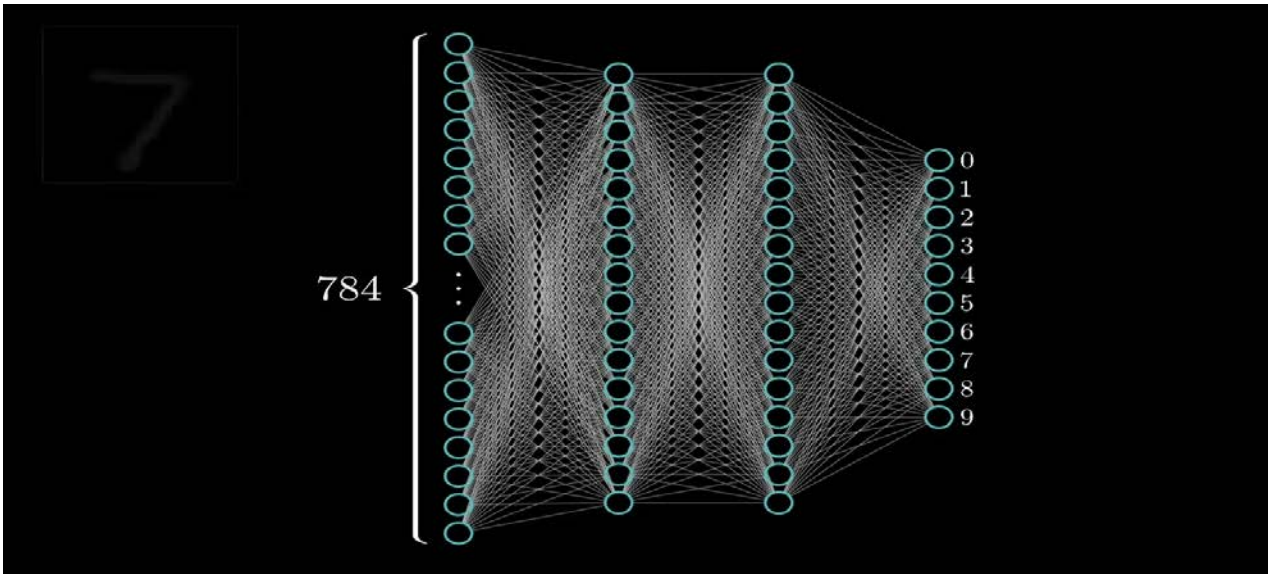
These neurons may be activated or fired according to inputs and this is decided by some functions called activation functions.

There are some types of activation functions as RELU, SOFTMAX, SIGMOID and TANH.

RELU is a non-linear activation function which used in multi-layer in neural network. The output of this function equal to input value when input greater than or equal zero and zero otherwise. It will activate neurons whose value greater than or equal zero and fire others.

SoftMax is a mathematical function which convert vector of number to vector of probabilities and every probability have a relation to its corresponding number. It will activate the neuron which has high probability and fire others.
(Note: SOFTMAX is chosen in output layer to activate one neuron only to detect the pattern)

ANN VISUALISATION



2- Model Refinement

The ANN is powerful model which generates high accuracy
So, we try to change some parameters to get higher accuracy
We add more layers sometimes or increase number of neurons in layers to get more accuracy.
We get some accuracy as 98.13%, 98.45%, 97.99%, 98.31%, 98.2 %, 98.25 %, 98.21 %, 98.16 %, 98.32, %...etc.

3- Model Testing and Evaluation

We tested different number of layers and neurons as we mentioned before and now, we will see confusion matrices, precision, recall, and r2_score of three highest accuracies and what is the number of layers and neurons in every one and compare them to get the best.

R2_Score

Accuracy = 98.32%	Accuracy = 98.31%	Accuracy = 98.45%
96.48905064888379	95.78304452936585	96.58922719286944

1- Layers = 6 layers with neurons = 512,256,128, 64,32,16
Accuracy = 98.45% iterations = 50

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	973	0	0	0	1	1	3	0	2	0
1	1	1126	1	2	0	1	2	1	1	0
2	1	3	1007	5	1	0	0	6	9	0
3	0	0	3	995	0	6	0	1	5	0
4	0	1	0	0	967	0	4	0	2	8
5	2	0	0	4	1	879	2	1	1	2
6	2	2	1	0	0	4	946	0	3	0
7	0	5	4	2	4	0	0	1004	2	7
8	2	0	1	4	1	2	1	0	960	3
9	0	4	0	4	4	2	1	3	3	988

2- Layers = 7 layers with neurons = 1024, 512,256,128, 64,32,16

Accuracy = 98.31% iterations = 50

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	978	0	0	0	0	0	1	0	1	0
1	0	1127	1	0	1	0	2	1	3	0
2	1	7	1013	1	0	0	0	6	4	0
3	0	1	4	992	0	7	0	1	4	1
4	0	4	1	0	961	0	2	1	3	10
5	2	0	0	4	1	880	3	0	2	0
6	8	3	0	0	1	3	939	0	4	0
7	0	12	6	2	4	1	0	996	2	5
8	2	1	3	3	1	0	0	0	957	7
9	2	6	1	3	4	0	0	1	4	988

3- Layers = 6 layers with neurons = 512,256,128, 64,32,16

Accuracy = 98.32% iterations = 100

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	972	0	0	0	1	1	2	1	1	0
1	0	1124	4	2	0	1	2	0	2	0
2	1	1	1022	1	1	0	0	5	1	0
3	0	1	1	998	0	3	0	5	2	0
4	0	1	3	0	967	0	0	1	3	7
5	2	0	0	12	0	872	2	1	2	1
6	1	1	3	0	3	3	941	0	6	0
7	0	2	10	2	1	0	0	1006	3	4
8	2	0	4	7	1	0	1	1	951	7
9	0	5	2	6	6	1	0	6	4	974

Precision & Recall

Classes	Accuracy = 98.45%		Accuracy = 98.31%		Accuracy = 98.32%	
	Precision	Recall	Precision	Recall	Precision	Recall
0	99.18450561	99.28571429	98.48942598	99.79591837	99.38650307	99.18367347
1	99.02826855	98.76651982	97.07149009	99.29515419	98.76977153	99.030837
2	99.01671583	99.2248062	98.44509232	98.15891473	97.42612011	99.03100775
3	97.93307087	96.73267327	98.70646766	98.51485149	97.08171206	98.81188119
4	98.77425945	96.94501018	98.76670092	98.47250509	98.77425945	98.47250509
5	98.2122905	98.0941704	98.7654321	98.5426009	98.9784336	97.75784753
6	98.64442127	99.0605428	99.15522703	98.7473904	99.36642027	98.22546973
7	98.81889764	98.44357977	99.00596421	97.66536965	97.95520935	97.85992218
8	97.1659919	98.15195072	97.25609756	98.56262834	97.53846154	97.6386037
9	96.8627451	97.91873142	97.72502473	97.91873142	98.09619238	97.02675917

Due to confusion matrices, precision, recall, and r2_score we get 6 layers with neurons 512,256,128, 64,32,16 and iterations = 50 are the best parameters.

Fourth: Convolution Neural Network

1. Model Definition

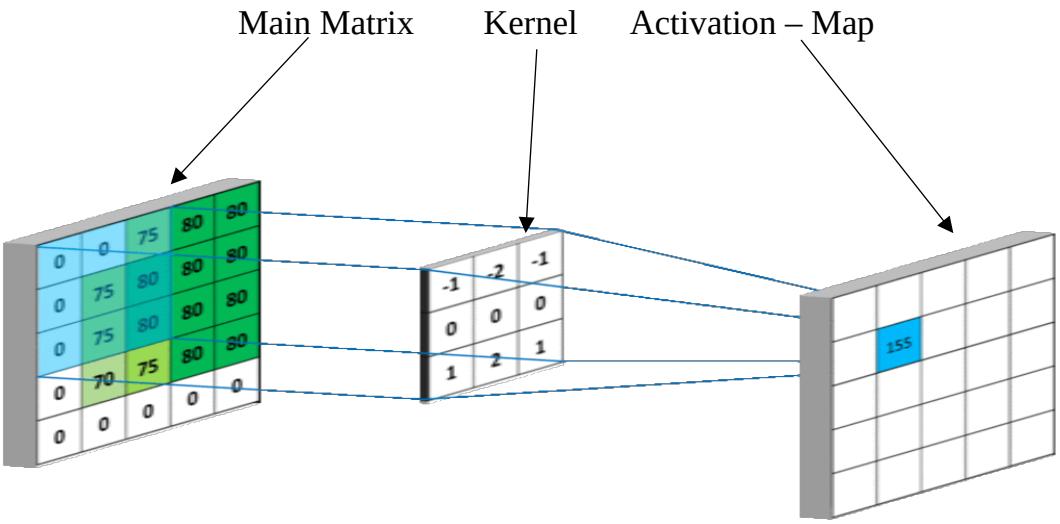
CNN: It is another type of neural network that specializes in processing data that has a grid-like topology, such as an image.

It is consisting of layers and neurons. Each neuron in any layer it fully connected with neurons in next layer.

It has three types of layers convolution layer, pooling layer and fully connected layer. Every layer has specific role in network to facilitate detection process.

Convolution layer: it is the core of CNN and it carries the main portion of the network's computational processes. It makes some dot product between kernels and part of the main matrix and it make kernel to slide on all the matrix to produce 2-D matrix which represent the image, this 2-D matrix is known as activation map gives the response of the kernel at each spatial position of the matrix.

CONVULUTION LAYER VISUALISATION

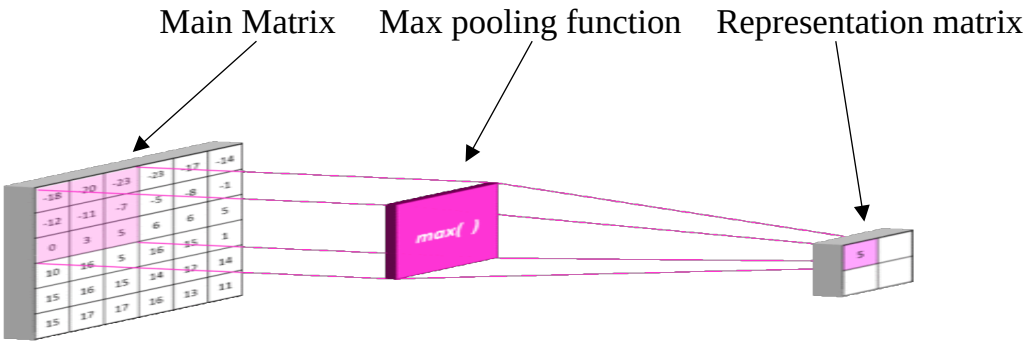


Pooling Layer: it replaces the output of the previous layer by summary statistics which represent the output.

This reduces the size of the representation and decrease the computations and weights. The pooling operation is processed on every slice of the representation individually.

There are types of pooling function as MAX POOLING and AVERAGE POOLING …etc.

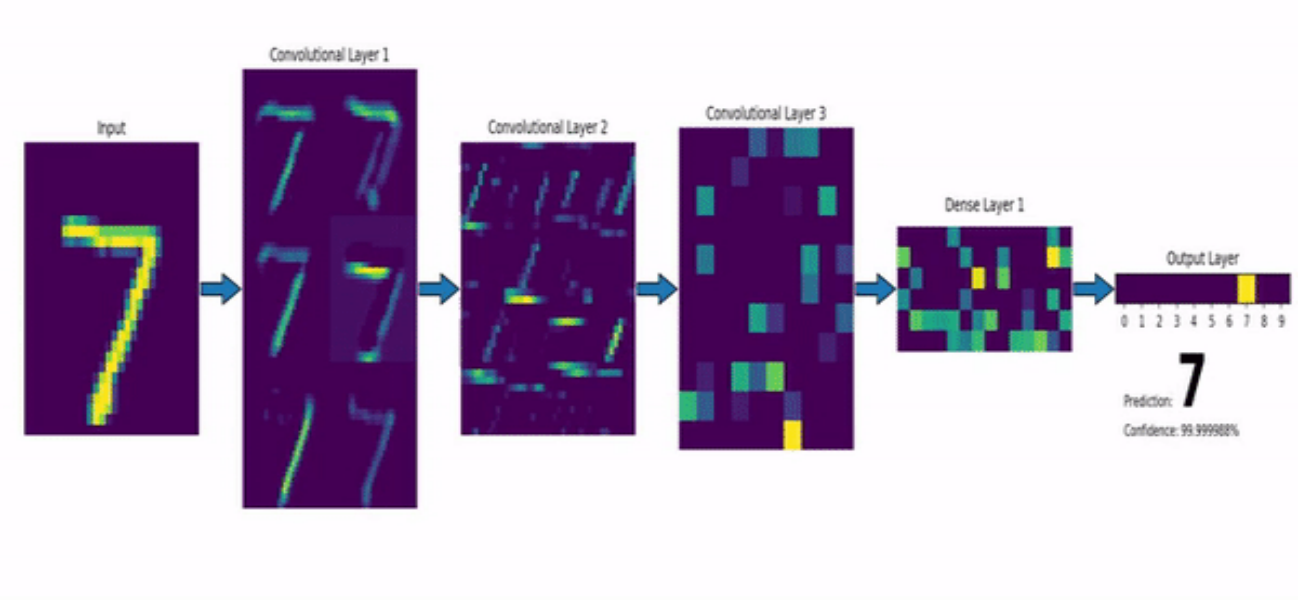
POOLING LAYER VISUALISATION



Fully Connected Layer: Neurons in this layer have full connectivity with all neurons in the preceding and succeeding layer as seen in regular FCNN. This is why it can be computed as usual by a matrix multiplication followed by a bias effect.

The FC layer helps to map the representation between the input and the output

CNN VISUALISATION



2- Model Refinement

The CNN is powerful model which generates high accuracy
So, we try to change some parameters to get higher accuracy
We change size of filters or increase number of iterations layers to get more accuracy.
We get some accuracy as 99%, 99.1%, 99.2%.

3- Model Testing and Evaluation

We tested different number of filters and iterations as we mentioned before and now, we will see confusion matrices, precision, recall, and r2_score of three highest accuracies and what is the number of layers and neurons in every one and compare them to get the best.

R2_Score

Accuracy = 99%	Accuracy = 99.1%	Accuracy = 99.2%
97.81519727878909	98.02628356790171	98.61899478648351

1- Filters = 7 (64 64 128 128 256 256 512)

Accuracy = 99% iterations = 20

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	980	0	0	0	0	0	0	0	0	0
1	2	1123	2	3	0	0	3	2	0	0
2	2	0	1027	1	0	0	1	1	0	0
3	0	0	0	1009	0	0	0	0	0	1
4	0	0	0	0	972	0	1	0	3	6
5	1	0	0	7	0	883	1	0	0	0
6	6	1	0	1	0	0	950	0	0	0
7	1	5	10	4	0	0	0	1004	2	2
8	1	0	1	2	0	1	0	0	968	1
9	0	0	0	5	3	6	0	1	9	985

2- Filters = 6 (64,64,128,128,256,512)

Accuracy = 99.1% iterations = 10

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	979	0	0	0	0	0	0	0	1	0
1	1	1132	0	0	0	1	1	0	0	0
2	1	1	1020	4	0	0	2	3	1	0
3	0	0	1	1006	0	2	0	0	1	0
4	0	0	0	0	975	0	1	0	1	5
5	0	0	0	5	0	884	1	0	0	2
6	5	2	0	1	1	3	994	0	2	0
7	0	3	11	0	0	0	0	1011	0	3
8	1	0	2	3	0	3	0	0	963	2
9	1	1	0	0	4	1	0	3	3	996

3- Filters = 6 (32, 64, 128, 256, 256, 512)

Accuracy = 99.2% iterations = 15

Confusion matrix

Classes	0	1	2	3	4	5	6	7	8	9
0	975	0	0	0	0	0	1	2	2	0
1	0	1126	4	1	0	2	0	2	0	0
2	1	0	1022	4	0	0	0	5	0	0
3	0	0	0	1004	0	5	0	0	1	0
4	0	0	0	0	979	0	0	0	0	3
5	0	0	0	3	0	887	1	1	0	0
6	2	3	0	0	3	9	939	0	2	0
7	0	2	1	2	0	0	0	1019	0	4
8	0	0	0	2	0	1	0	0	970	1
9	0	0	0	0	2	5	0	0	3	999

Precision & Recall

Classes	Accuracy = 99%		Accuracy = 99.1%		Accuracy = 99.2%	
	Precision	Recall	Precision	Recall	Precision	Recall
0	98.69083585	100	99.08906883	99.89795918	99.69325153	99.48979592
1	99.46855624	98.94273128	99.38542581	99.73568282	99.55791335	99.20704846
2	98.75	99.51550388	98.64603482	98.8372093	99.51314508	99.03100775
3	97.77131783	99.9009901	98.72423945	99.6039604	98.81889764	99.40594059
4	99.69230769	98.98167006	99.48979592	99.28716904	99.49186992	99.69450102
5	99.21348315	98.99103139	98.88143177	99.10313901	97.57975798	99.43946188
6	99.37238494	99.16492693	99.47312961	98.53862213	99.78746015	98.01670146
7	99.6031746	97.66536965	99.4100295	98.3463035	99.0281827	99.12451362
8	98.57433809	99.38398357	99.07407407	98.87063655	99.18200409	99.58932238
9	98.99497487	97.62140733	98.80952381	98.71159564	99.20556107	99.00891972

Due to confusion matrices, precision, recall, and r2_score we get 6 layers with filters 32, 64, 128, 256, 256, 512and iterations = 15 are the best parameters.

Conclusion

After those test cases we find there is different between accuracies and this which distinguish between models which one is the best. So, we compare the best accuracies of these models and find the convolution neural network is the best one with higher accuracy 99.2%, then support vector machine with 98.61% then artificial neural network with 98.45% and finally KNN with 96.619%. So, we decided to depend on convolution neural network to make our GUI because it is the best and powerful one.