

Comparison of Machine Learning Classifiers for Recognition of Online and Offline Handwritten Digits*

Elijah Olusayo Omidiora, Ibrahim Adepoju Adeyanju*, Olusayo Deborah Fenwa*

Department of Computer Science and Engineering, Ladoke Akintola University of Technology,

P.M.B. 4000, Ogbomoso, 210214, Nigeria

*E-mail of the corresponding authors: iaadeyanju@lautech.edu.ng, odfenwa@lautech.edu.ng

This work was done as part of a research project funded by LAURESCON (LAUTECH Research and Consultancy unit) as Ladoke Akintola University of Technology (LAUTECH) Senate Research Grant (LAU/SRG/13/025).

Abstract

Automated recognition of handwritten digits has applications in several industries such as Postal and Banking for reading of addressed packages and cheques respectively. This paper compares four machine learning classifiers namely Naive Bayes, Instance Based Learner, Decision Tree and Neural Network for single digit recognition. Our experiments were conducted using the WEKA machine learning tool on two datasets; the MNIST offline handwritten digits and a collection of online ISGL handwritten digits acquired with a pen digitiser. Experiments were designed to allow for comparison within the datasets in a cross validation and across them where the online dataset is used for training and the offline dataset for testing and vice versa. We also compared classification accuracy at different levels of down sampling. Results indicate that the lazy learning instance based classifier performed slightly better than the neural network with a maximal accuracy of 97.86% and they both outperformed the other two classifiers: Naive Bayes and Decision Tree. The decision tree gave the worst performance of the four classifiers. We also discovered that better results were obtained with using the online digits when tested in a cross validation experiment. However, the pre-processed MNIST offline digits gave higher accuracies when used for training and tested with the online ISGL digits not vice versa. Also, we discovered down sampled size of 14x14 gave the best results for most of the four classifiers although these were not significantly different from the other down sampled sizes of 7x7, 21x21 and 28x28. We intend to investigate the performance of these classifiers in recognition of other characters (alphabets, punctuation and other symbols) as well as extend the recognition task to other levels of text granularity such as words, sentences and paragraphs.

Keywords: Digits recognition, machine learning, classifiers, handwritten character recognition, Weka

1. Introduction

Character recognition is the process of applying pattern matching methods to character shapes that have been read into a computer to determine which alpha-numeric character, punctuation marks, and symbols they represent. Recognition systems can be categorised as online or offline, based on their mode of data acquisition. The online systems process handwritten data captured during the writing process using a digital sketch pad and pen while the offline systems work have their data written on paper and then scanned as static images once the writing process is over. Various approaches that have been used to develop character recognition systems include template matching approach, statistical approach, structural approach, neural networks approach and hybrid approach (Cheriet, Kharma, Liu, & Suen 2007).

In this paper, we compare the performance of four different machine learning classifiers for recognition of digits. The four classifiers namely Naive Bayes, Instance Based Learner, Decision Tree and Multilayer perceptron neural network belong to Bayes, Lazy learner, Inductive Trees and Function machine learning categories respectively. Our main contribution in this work is the experiments with these classifiers on a new collection of online character recognition dataset and its comparison with existing datasets. Section 2 discusses our related work while the machine learning classifiers are explained in Section 3. The character recognition datasets used in our experiments are discussed in Section 4 with Experimental setup in Section 5. Discussion of evaluation results and our conclusion appear in Sections 6 and 7 respectively.

2. Related Work

Humans have been writing important information on paper (and other primitive materials such as woods, clay tablets etc.) even before the advent of computer systems. However, storing such information has typically been very cumbersome and expensive. The use of these handwritten data on paper is generally limited to those around where it is stored though several others across the globe might benefit from it. Digitising such information and converting them to recognisable text allows search engines such as Google, Yahoo and MSN to be able to index and make them available for user around the world once they are uploaded to the web.

Recognition of handwritten characters with respect to any language is difficult due to variability in writing styles and cursive representation of character among others (Shanthi & Duraiwamy 2007). A lot of the character recognition systems only consider constrained recognition problems based on small vocabularies from a specific domain. A good example is the recognition of handwritten cheque amount or postal addresses (Kaltenmeler, Ceaser, Gloger, & Mandler 1993;Camastra & Vinciarelli 2001;Atallah 2009). The recognition systems (Marti & Bunke 2001;Muhammad, Dzuulkifli, & Razib 2005) that addressed freely handwritten characters without domain specific constraints and large vocabularies reported accuracies significantly worse than those of the constrained system. Hence, there is opportunity for improvement in developing unconstrained character recognition systems. Our work simulates a constrained handwritten character recognition system.

Selection of a feature extraction method is undoubtedly the single most important factor in achieving high recognition performance in character recognition system (Pradeep, Srinivasan, & Himavathi 2011). No matter how sophisticated the classifiers and learning algorithms, poor feature extraction will always lead to poor system performance (Marc, Alexandre, & Christian 2001;Ahmad 2006;Fenwa, Omidiora, & Fakolujo 2012;Simon & Horst 2004a). The main feature of any handwritten character image is the pixel values. However, online data acquisition of character images with electronic devices such as a pen digitizer allows extraction of other features such as number of stroke, pressure of the pen and contour pixels. We only used the pixel value features in our work to enable reasonable comparison between our online and offline acquired datasets.

Several algorithms have been applied to recognition of characters at different levels of text granularity producing varying classification accuracies. These include Hidden Markov Models (Kaltenmeler, Ceaser, Gloger, & Mandler 1993;Marti & Bunke 2001), Vector quantisation (Camastra & Vinciarelli 2001) and Neural Networks (Fenwa, Omidiora, Fakolujo, Ajala, & Oke 2012;Freeman & Skapura 1992; Muhammad, Dzuulkifli, & Razib 2005). Other algorithms are Support Vector Machines (Shanthi & Duraiwamy 2007; Nasien, Haron, & Yuhaniz 2010), Genetic algorithm (Simon & Horst 2004b) and probabilistic models (Kozielski, Rybach, Hahn, Schluter, & Ney 2013). Our work compares some of these classifiers in recognition for both online and offline handwritten digits.

3. Machine Learning Classifiers for Character Recognition

In this section, we briefly discuss the details of the four classifiers used in our experiments. Extensive details of each classifier can be found in the cited references in Section 3.1 to 3.4.

3.1 Naive Bayes

A naive Bayes classifier assumes that the presence or absence of a particular feature is unrelated to the presence or absence of any other feature, given the class variable (John & Langley 1995; Wikipedia 2013f). It considers each feature to contribute independently to the probability regardless of the presence or absence of the other features. In practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood which means that one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods. Despite its seemingly naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. An analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers (Zhang 2004). Nevertheless, a comprehensive comparison with other classification algorithms showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests (Caruana & Niculescu-Mizil 2006).

An advantage of Naive Bayes is that it only requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix. The naive Bayes classifier combines this model with a decision rule. One common rule is to pick the hypothesis that is most probable; this is known as the maximum aposteriori (MAP) decision rule. The corresponding classifier, a Bayes classifier, is the function classify defined as follows:

$$\mathrm{classify}(f_1, \dots, f_n) = \underset{c}{\mathrm{argmax}}\ p(C = c) \prod_{i=1}^{n} p(F_i = f_i | C = c).$$

where C is a dependent class variable with a small number of outcomes or classes, conditional on several feature variables F_1 through F_n (Wikipedi 2013f).

3.2 Instance Based Learner (IBk)

Instance-based learning, also known as memory-based learning, is a family of learning algorithms that, instead of performing explicit generalization, compare new problem instances with instances seen in training, which have been stored in memory. Instance-based learning is a kind of lazy learning. It is called instance-based because it constructs hypotheses directly from the training instances themselves (Aha, Kibler, & Albert 1991;Russell & Norvig, 2003;Wikipedia 2013c). This means that the hypothesis complexity can grow with the data. One advantage that instance-based learning has over other methods of machine learning is its ability to adapt its model to previously unseen data. Where other methods generally require the entire set of training data to be re-examined when one instance is changed, instance-based learners may simply store a new instance or throw away old instances.

A typical example of an instance-based learning algorithm is the k-nearest neighbour algorithm (Hall, Park, & Samworth 2008). The k-nearest neighbours algorithm (k-NN) is a non-parametric method for classification and regression that predicts objects' "values" or class memberships based on the k closest training examples in the feature space. k-NN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification. The k-nearest neighbour algorithm is amongst the simplest of all machine learning algorithms: an object is classified by a majority vote of its neighbours, with the object being assigned to the class most common amongst its k nearest neighbours (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbour. The same method can be used for regression, by simply assigning the property value for the object to be the average of the values of its k nearest neighbours. It can be useful to weight the contributions of the neighbours, so that the nearer neighbours contribute more to the average than the more distant ones. A common weighting scheme is to give each neighbour a weight of 1/d, where d is the distance to the neighbour. The neighbours are taken from a set of objects for which the correct classification (or, in the case of regression, the value of the property) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. The k-nearest neighbour (k-NN) algorithm is sensitive to the local structure of the data.

The training examples in k-NN are vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the classification phase, k is a user-defined constant, and an unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the k training samples nearest to that query point. A commonly used distance metric for continuous variables is Euclidean distance. The Euclidean distance between two vectors is as defined below, where a and a' are vectors of same length.

$$d(a, a') = \sqrt{(a_1 - a_1')^2 + \ldots + (a_n - a_n')^2}$$

The accuracy of the k-NN algorithm can be severely degraded by the presence of noisy or irrelevant features, or if the feature scales are not consistent with their importance. Much research effort has been put into selecting or scaling features to improve classification (Wikipedia, k-nearest neighbors algorithm, 2013c). The k-NN instance based learner is implemented as IBk in Weka tool (Bouckaert et al. 2011).

3.3 Decision Trees (C4.5)

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility (Wikipedia 2013b). Decision trees build hierarchical structures for classification using a divide and conquer strategy. This is done by choosing an attribute for the root node and branch for each value of that attribute. Instances or training examples are then split according to branches. This process is repeated for each branch until all instances in the branch have the same class. A basic assumption is that the simplest tree which classifies the training examples is best. Each attribute contributes an amount of information to the classification. Therefore, attributes that produces the "purest" nodes are selected first. The purity criterion is typically determined using information gain which is based on information entropy. Information gain increases with the average purity of the subsets that an attribute produces. In general terms, the expected information gain is the change in information entropy from a prior state to a state that takes some information as given:

$$IG(T,a) = H(T) - H(T|a)$$

where H denotes the information entropy.

Decision tree learning is the construction of a decision tree from class-labeled training tuples. A decision tree is a flow-chart-like structure, where each internal (non-leaf) node denotes a test on an attribute, each branch represents the outcome of a test, and each leaf (or terminal) node holds a class label. The topmost node in a tree is the root node. C4.5 is an algorithm used to generate a decision tree and an extension of the ID3 algorithm (Quinlan R. 1993;Quinlan J. R. 1996). The decision trees generated by C4.5 can be used for classification and was implemented in Weka (Bouckaert et al. 2011) which we used in our experiments.

3.4 Artificial Neural Network (Multi Layer Perceptron)

Artificial neural networks are computational models inspired by the central nervous systems. They are usually presented as systems of interconnected "neurons" that can compute values from inputs by feeding information through the network. For example, in a neural network for handwriting recognition, a set of input neurons may be activated by the pixels of an input image representing a letter or digit. The activations of these neurons are then passed on, weighted and transformed by some function determined by the network's designer, to other neurons, etc., until finally an output neuron is activated that determines which character was read (Wikipedia 2013a). Neural networks are also similar to biological neural networks in performing functions collectively and in parallel by the units, rather than there being a clear delineation of subtasks to which various units are assigned. The word network in the term 'artificial neural network' refers to the inter–connections between the neurons in the different layers of each system. An example system has three layers. The first layer has input neurons, which send data via synapses to the second layer of neurons, and then via more synapses to the third layer of output neurons. More complex systems will have more layers of neurons with some having increased layers of input neurons and output neurons. The synapses store parameters called "weights" that manipulate the data in the calculations (Rosenblatt 1958).

A multilayer perceptron (MLP) is a feed forward artificial neural network model that maps sets of input data onto a set of appropriate outputs. An MLP consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. Except for the input nodes, each node is a neuron (or processing element) with a nonlinear activation function. MLP utilizes a supervised learning technique called back propagation for training the network (Wikipedia 2013e). MLP is a modification of the standard linear perceptron and can distinguish data that are not linearly separable. If a multilayer perceptron has a linear activation function in all neurons, that is, a simple on-off mechanism to determine whether or not a neuron fires, then it is easily proved with linear algebra that any number of layers can be reduced to the standard two-layer input-output model. What makes a multilayer perceptron different is that each neuron uses a nonlinear activation function which was developed to model the frequency of action potentials, or firing, of biological neurons in the brain. This function is modelled in several ways, but must always be normalizable and differentiable.

4. Digits (Character) Recognition Datasets

We discuss the two datasets used in our experiments in this section. The first is an offline handwritten collection of digits provided by NIST discussed in Section 4.1 while the other is a new collection of online handwritten digits collated by the Intelligent Systems Group of the Ladoke Akintola University of Technology, Ogbomoso, Nigeria.

4. 1 *Offline Acquired MNIST Digits Dataset*

This dataset is a part of the Modified National Institute of Standards and Technology (MNIST) character

recognition database (Lecun 1998). The MNIST database was constructed from NIST's Special Database 3 and Special Database 1 which contain binary images of handwritten digits. The MNIST training set is composed of 30,000 patterns from SD-3 and 30,000 patterns from SD-1. 60,000 pattern training set containing examples from approximately 250 writers were extracted as well as 10,000 test images. The original black and white (bi-level) images from NIST (Grother 1995) were size normalized to fit in a 20x20 pixel box while preserving their aspect ratio. The resulting images contain grey levels as a result of the anti-aliasing technique used by the normalization algorithm. The images were centred in a 28x28 image by computing the centre of mass of the pixels, and translating the image so as to position this point at the centre of the 28x28 field.
In our experiments, we sampled 1000 images without replacement from the 10,000 test images ensuring an equal class distribution. Only 1000 images were extracted to make the dataset comparable to the ISGL dataset discussed in Section 4.2. The extraction was done using the WEKA (Witten & Frank 2005) filter called "resample" which produces a random subsample of a dataset with or without replacement. Figure 1 shows examples of images from our extracted MNIST digit recognition dataset.

Figure 1: Example of digit images from the MNIST dataset

4.2 Online Acquired ISGL dataset

This dataset consists of single characters collected by the Intelligent System Group LAUTECH (ISGL) using a Digitizer tablet (G-Pen 450) as part of a doctoral research work (Fenwa 2012). The G-Pen has an electric pen with sensing writing board. A programming interface was developed to acquire a set of characters from fifty (50) different authors using the Digitizer tablet in a constrained environment. Characters considered were 26 upper case (A-Z), 26 lower case (a-z) English alphabets and 10 digits (0-9) making a total number of 62 characters. Each character was collected twice from each author to allow for possible variations of a single character written by the same individual. A total of 6,200 characters were collected with 100 samples per character. Each character had an original dimension of 200x130. For our experiments, we used only the digits (0-9) consisting of 1000 images. Figure 2 shows a sample of digits images in the ISGL dataset.

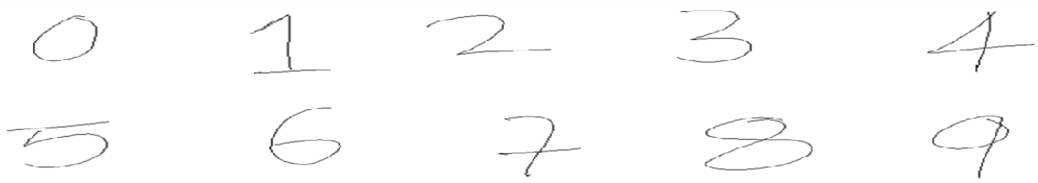

Figure 2: Example of digit images from the ISGL dataset

5. Experimental Setup

Experiments were designed to determine which of the machine learning classifiers (Naive Bayes, Nearest Neighbour, Decision Tree and Neural Network) as discussed in Section 3, is best to use when building a character (digit) recognition system. The Waikato Environment for Knowledge Analysis (Weka) (Witten & Frank 2005; Bouckaert et al 2011) tool (version 3.6.6) was used in experimentation as it has all these classifiers implemented. The specific configurations used were as follows:

1. Naive Bayes: All parameters were left at the default values as indicated in Weka.
2. Instance Based Learner: The IBk algorithm was configured with k=3 using the 1/distance weighting function while the Euclidean distance was used as the similarity metric. Other parameters were left at default.

3. Decision Tree: The C4.5 (J48) inductive learner was used with the default parameter values.
4. Artificial Neural Network: The Multilayer Perceptron algorithm was used with the number of hidden layers chosen as the number of possible classes (i.e. 0-9 or ten classes in our case) while the training time was set at 200 epochs. All other parameters were left at the default values.

Two types of experiments were designed. Firstly, we wanted to compare the four classifiers on the two separate datasets (online ISGL and offline MNIST). For this, we used a ten-fold cross validation experimental design repeated ten times for each dataset. The second design involved using the online ISGL dataset for training and the offline MNIST dataset for testing and vice versa.

The same pre-processing step of down sampling (Fenwa 2012) the original digit images to a 7x7 dimension was carried out on both datasets to make our experiments computationally less expensive. Since we are dealing with single character images, each pixel value can be binarised since the background colour will be distinctive from the colour of the character (digit) text. We assumed a value of one (1) for each pixel that is part of the handwritten character digit while the background get a zero (0) value. Therefore, the only features used were the forty-nine (49) binary pixel values of the down sampled images. These were nominalised for use in Weka since viewing them as a numeric value will lead models to producing decimal values that are meaningless for the classification task. Other features such as number of strokes and pen pressure that might be associated with online data acquisition were ignored to make experiments on the two (online and offline) datasets comparable.

6. Results and Discussion

In all evaluation results discussed in this section, test of significance was done at 95% confidence. Two common classification metrics were used in our evaluation named percentage accuracy and root mean squared error. Values shown in tables with "v" beside it indicate that a value is statistically better at 95% confidence than others in the same row. An asterisk (*) beside a value in a table shows that it is significantly worse than others.

6.1 Cross validation experiments

The evaluation results of our cross validation experiments are shown in Tables 1 to 4. Table 1 shows classification accuracy across the four classifiers: Naive Bayes (NB), Instance Based learner (IBk), Decision Tree (C4.5) and Multilayer Perceptron (MLP) while Table 2 contains the equivalent root mean squared (rms) error. Tables 3 and 4 show the same metrics (accuracy and rms error) for four different down sampled sizes of both the MNIST and ISGL datasets. A summary of the performance of each classifier relative to the first classifier (Naive Bayes) across the datasets is also in the last row of each table.

The Instance Based Learner, IBk, outperforms the other three classifiers on both datasets with a best classification accuracy of 97.28% on the ISGL dataset. However, the IBk results were statistically similar to that of the Multilayer Perceptron, MLP, where they were both statistically better than the other two classifiers, Naive Bayes (NB) and Decision Tree (C4.5). Nevertheless, the absolute values of 97.28% and 79.43% for IBk were greater than those (95.67% and 75.12%) obtained with MLP on the ISGL and MNIST datasets respectively. This indicates that the lazy learner (IBk) is well suited for character (digit) recognition with an advantage of being computationally less expensive than MLP since it has no training phase.

Table 1: Comparison of percentage accuracy of machine learning classifiers in single digit recognition

Dataset/ Classifiers	**NB**	**IBk**	**C4.5**	**MLP**
ISGL online digits	93.46	97.28 v	90.96 *	95.67 v
MNIST offline digits	74.86	79.43 v	71.24 *	75.12
	(v/ /*)	(2/0/0)	(0/0/2)	(1/1/0)

C4.5 performed significantly worse than the other three classifiers as shown in Table 1. This means that decision trees are not suited for character recognition tasks probably due to the high number of attributes applicable from using the pixel values. NB had an average performance relative to the other classifiers and might be as suited for character recognition tasks. Across the two datasets, the accuracy values obtained from the ISGL dataset seem more promising and better than those obtained from the MNIST dataset. This might not be unconnected to the fact that Machine Learning algorithms are able decipher written directly on digital surfaces as opposed to those written on paper and then scanned despite the extensive pre-processing. This is also in line with results obtained by other researchers in the field of character recognition (Pradeep, Srinivasan, & Himavathi, 2011).

Table 2 validates the accuracy results shown in Table 1 as the trend across the classifiers and datasets remain the same. The rms error values are inversely proportional to the accuracies and the lower the error, the better the performance of the classifier.

Table 2: Comparison of root mean squared error of machine learning classifiers in single digit recognition

Dataset/ Classifiers	**NB**	**IBk**	**C4.5**	**MLP**
ISGL online digits	0.10	0.06 *	0.12 v	0.08 *
MNIST offline digits	0.21	0.17 *	0.22 v	0.20
	(v/ /*)	(0/0/2)	(2/0/0)	(0/1/1)

We compared the ISGL digit images at four different down sampling sizes: 7x7, 14x14, 21x21 and 28x28. The results are shown in Tables 3 and 4. The result pattern across the classifiers remains largely the same for the different down samples with IBk and MLP outperforming others. However, differences across the down samples do not seem significant with no consistent pattern as the down sample size increases. One interesting observation though is that the results for the 28x28 down samples were worse than others for all the classifiers apart from Naive Bayes. This indicates that a 7x7 down sample might be sufficient to justify any results since it is the least computationally intensive. Table 4 validates the results in Table 3 with the inversely proportional rms errors.

Table 3: Comparison of percentage accuracy of classifiers in single digit recognition using different down sampled sizes of the ISGL online digit images

Down sampled size / Classifier	**NB**	**IBk**	**C4.5**	**MLP**
7x7	93.46	97.28 v	90.96 *	95.67 v
14x14	94.80	97.86 v	89.51 *	96.22 v
21x21	94.52	96.97 v	85.86 *	96.26 v
28x28	94.09	95.50	84.15 *	94.73
	(v/ /*)	(3/1/0)	(0/0/5)	(3/1/0)

Table 4: Comparison of root mean squared error of classifiers in single digit recognition using different down sampled sizes of the ISGL online digit images

Down sampled size / Classifier	**NB**	**IBk**	**C4.5**	**MLP**
7x7	0.10	0.06 *	0.12 v	0.08 *
14x14	0.09	0.05 *	0.14 v	0.08 *
21x21	0.10	0.07 *	0.16 v	0.08 *
28x28	0.10	0.09 *	0.17 v	0.09 *
	(v/ /*)	(0/0/4)	(4/0/0)	(0/0/4)

6.2 Experiments with alternating the two datasets for training and testing

Tables 5 and 6 show the evaluation results when two datasets are alternately used for training and testing at different times. Accuracies are shown in Table 5 while rms errors are shown in Table 6. The first observation from the Tables is that the accuracy values are mostly far less than what was obtained in the cross validation experiments in Tables 1 and 3. This is not unexpected as the training and testing data are from different sources and by different writers. However, this could be used as the realistic or ideal accuracies expected from the classifiers when they are used in the industry. The highest accuracy of 62.9% was obtained with the IBk classifier which still outperforms the other three clasifiers. A peculiar occurrence is the performance of MLP which performs even worse than C4.5 when ISGL dataset was used for training and the MNIST for testing. This might indicate that MLP is not as robust as purported especially if the test data is from a significantly different source.

Table 5: Comparison of percentage accuracy of machine learning classifiers in single digit recognition

Training dataset	Testing dataset	NB	IBk	C4.5	MLP
ISGL online digits	MNIST offline digits	39.6	50.5	33.7	27.2
MNIST offline digits	ISGL online digits	52.1	62.9	28.3	46.3

Higher accuracies were generally obtained when the offline MNIST digits were used for training and the online ISGL digits for testing except for C4.5. This might be as a result of the extensive pre-processing carried out initially on the offline MNIST dataset. It could also be an indication that offline datasets are best for training a digit recogniser. The error values in Table 6 also validate our accuracy results of Table 5.

Table 6: Comparison of root mean squared error of machine learning classifiers in single digit recognition

Training dataset	Testing dataset	NB	IBk	C4.5	MLP
ISGL online digits	MNIST offline digits	0.2871	0.2791	0.3549	0.2989
MNIST offline digits	ISGL online digits	0.3327	0.2276	0.3419	0.3384

7. Conclusion and Future Work

This paper compared four classifiers for recognition of both online and offline handwritten digits. Our results indicate an instance based learner (IBk) performed better than the other three classifiers and is more robust under different configurations. The other three classifiers are Naive Bayes, Decision Tree and Multilayer perceptron neural network. We intend to extend this work further by investigating the performance of other classifiers and experimenting with other levels of text granularity.

References

Aha, D., Kibler, D., & Albert, M. K. (1991), "Instance-based learning algorithms", *Machine Learning* 6, 37-66.

Ahmad, T. (2006), "An Efficient Feature Extraction Algorithm for the Recognition of Handwritten Arabic Digits", *International Journal of Computational Intelligence* 2 (2), 107-112.

Atallah, M. (2009), "Skew Detection and Correction Technique for Arabic Document Images based on Centre of Gravity", *Journal of Computer Science* 5 (5), 363-368.

Bouckaert, R. R., Frank, E., Hall, M., Kirkby, R., Reutemann, P., Seewald, A., et al. (2011), "WEKA Manual for Version 3-6-6".

Camastra, F., & Vinciarelli, A. (2001), "Cursive Character Recognition by Learning Vector Quantization", *Pattern Recognition Letters* 1 (22), 625–629.

Caruana, R., & Niculescu-Mizil, A. (2006), "An Empirical Comparison of Supervised Learning Algorithms", *Proceedings of the International Conference on Machine Learning (ICML 06)*, 161-168.

Cheriet, M., Kharma, N., Liu, C.-L., & Suen, C. Y. (2007), *Character Recognition Systems: A Guide for Students and Practitioners*, John Wiley & Sons Inc.

Fenwa, O. D. (2012), "Development of a Writer-Independent Online Handwritten Character Recognition System Using Modified Hybrid Neural Network Model", *PhD thesis*, Ladoke Akintola University of Technology, Ogbomoso.

Fenwa, O. D., Omidiora, E. O., & Fakolujo, O. A. (2012), "Development of a Feature Extraction Technique for Online Character Recognition System", *Journal of Innovative Systems Design and Engineering* 3 (3), 10-23.

Fenwa, O. D., Omidiora, E. O., Fakolujo, O. A., Ajala, F. A., & Oke, A. (2012), "Modified Optical Backpropagation Neural Network Model in Online Handwritten Character Recognition System", *International Journal of Computer Application* 4 (2), 190-201.

Freeman, J., & Skapura, D. (1992), *Backpropagation Neural Networks Algorithm Applications and Programming Techniques*, Addison-Wesley Publishing Company.

Grother, P. J. (1995), "NIST Special Database 19 Handprinted Forms and Characters Database".

Hall, P., Park, B., & Samworth, R. (2008), "Choice of neighbor order in nearest-neighbor classification", *Annals of Statistics* 36, 2135–2152.

John, G. H., & Langley, P. (1995), "Estimating Continuous Distributions in Bayesian Classifiers", *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, 338-345.

Kaltenmeler, A., Ceaser, T., Gloger, J., & Mandler, E. (1993), "Sophisticated Topology of Hidden Markov Models for Cursive Script Recognition", *Proceedings of the 2nd International Conference on Document Analysis and Recognition*, 139-142.

Kozielski, M., Rybach, D., Hahn, S., Schluter, R., & Ney, H. (2013), "Open Vocabulary Handwriting Recognition Using Combined Word-Level and Character-Level Language Models", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).*

Lecun, Y. (1998), "The MNIST Database of Handwritten Digits".

Marc, P., Alexandre, L., & Christian, G. (2001), "Character Recognition Experiment using Unipen Data", *Proceeding of International Conference on Document Analysis and Recognition (ICDAR)*, 10-13.

Marti, U., & Bunke, H. (2001), "Using a Statistical Language Model to Improve the Performance of an HMM-based Handwriting Recognition System", *International Journal of Pattern Recognition and Art Intelligence* 15, 65-90.

Muhammad, F., Dzuulkifli, M., & Razib, M. (2005), "Online Handwritten Character Recognition System: An Implementation of Counter-propagation Neural Network", *Proceeding of the World Academy of Science, Engineering and Technology*, 232-237.

Nasien, D., Haron, H., & Yuhaniz, S. S. (2010), "Support Vector Machine (SVM) for English Handwritten Character Recognition", *Proceedings of the Second International Conference on Computer Engineering and Applications*, 249-252.

Pradeep, J., Srinivasan, E., & Himavathi, S. (2011), "Diagonal Based Feature Extraction for Handwritten Alphabets Recognition using Neural Network", *International Journal of Computer Science and Information Technology (IJCSIT)* 3 (1), 27-37.

Quinlan, J. R. (1996), "Improved use of continuous attributes in C4.5", *Journal of Artificial Intelligence Research* 4, 77-90.

Quinlan, R. (1993), *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers.

Rosenblatt, F. (1958), "The Perceptron: A Probabilistic Model For Information Storage And Organization In The Brain", *Psychological Review* 65 (6), 386–408.

Russell, S., & Norvig, P. (2003), *Artificial Intelligence: A Modern Approach* (second edition ed.), Prentice Hall.

Shanthi, N., & Duraiwamy, K. (2007), "Performance Comparison of Different Image Sizes for Recognizing Unconstrained Handwritten Tamil Characters Using SVM", *Journal of Science* 3 (9), 760-764.

Simon, G., & Horst, B. (2004a), "Feature Selection Algorithms for the Generalization of Multiple Classifier Systems and their Application to Handwritten Word Recognition", *Pattern Recognition Letters* 25 (11), 1323-1336.

Simon, G., & Horst, B. (2004b), "Optimization of weights in a multiple Classifier Handwritten Word Recognition System using a Genetic Algorithm". *Electronic Letters on Computer Vision and Image Analysis* 3 (1), 25-41.

Wikipedia (2013a), "Artificial neural network", [Online] Available: http://en.wikipedia.org/wiki/Artificial_neural_network (Nov 15, 2013).

Wikipedia (2013b), "Decision tree", [Online] Available: http://en.wikipedia.org/wiki/Decision_tree (Nov 15, 2013).

Wikipedia (2013c), "Instance-based Learning", [Online] Available: http://en.wikipedia.org/wiki/ Instance-based_learning (Nov 15, 2013).

Wikipedia (2013d), "k-nearest neighbors algorithm", [Online] Available: http://en.wikipedia.org/wiki/K-nearest_neighbor_algorithm (Nov 15, 2013).

Wikipedia (2013e), "Multilayer perceptron", Available: http://en.wikipedia.org/wiki/Multilayer_perceptron (Nov 15, 2013).

Wikipedia (2013f), "Naive Bayes classifier", Available: http://en.wikipedia.org/wiki/Naive_Bayes_classifier (Nov 15, 2013).

Witten, I., & Frank, E. (2005), *Data Mining: Practical machine learning tools and techniques* (2nd ed.), Morgan Kaufmann.

Zhang, H. (2004), "The Optimality of Naive Bayes", *Proceedings of the International Florida Artificial Intelligence Research Society (FLAIRS) Conference.*