# Exploratory Analysis of MNIST Handwritten Digit for Machine Learning Modelling

Mohd Razif Shamsuddin, Shuzlina Abdul-Rahman(✉),
and Azlinah Mohamed(✉)

Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA,
40450 Shah Alam, Selangor, Malaysia
razif@tmsk.uitm.edy.my,
{shuzlina,azlinah}@tmsk.uitm.edu.my

**Abstract.** This paper is an investigation about the MNIST dataset, which is a subset of the NIST data pool. The MNIST dataset contains handwritten digit images that is derived from a larger collection of NIST data which contains handwritten digits. All the images are formatted in 28 × 28 pixels value with grayscale format. MNIST is a handwritten digit images that has often been cited in many leading research and thus has become a benchmark for image recognition and machine learning studies. There have been many attempts by researchers in trying to identify the appropriate models and pre-processing methods to classify the MNIST dataset. However, very little attention has been given to compare binary and normalized pre-processed datasets and its effects on the performance of a model. Pre-processing results are then presented as input datasets for machine learning modelling. The trained models are validated with 4200 random test samples over four different models. Results have shown that the normalized image performed the best with Convolution Neural Network model at 99.4% accuracy.

**Keywords:** Convolution Neural Network · Handwritten digit images
Image recognition · Machine learning · MNIST

## 1 Introduction

The complexity of data in the future is increasing rapidly, consistent with the advances of new technologies and algorithms. Due to the advancements of research in computer vision, machine learning, data mining and data analytics, the importance of having a reliable benchmark and standardized datasets cannot be ignored. Benchmark and standardized datasets help to provide good platforms to test the accuracy of different algorithms [1–4]. Comparing the accuracies of different algorithms can be conducted without having to necessarily recreate previously tested models.

As the behaviors and features of different datasets vary significantly, the capabilities of different machine learning models have always been evaluated differently. This evaluation always happens in isolated research experiments where created models were always biased to a specific dataset. Thus, the perseverance of a differing suite of benchmarks is exceptionally important in enabling a more effective way to deal with

B. W. Yap et al. (Eds.): SCDS 2018, CCIS 937, pp. 134–145, 2019.
https://doi.org/10.1007/978-981-13-3441-2_11

surveying and assessing the execution of a calculation or newly created model. There are several standardized datasets in the machine learning community, which is widely used and have become highly competitive such as the National Institute of Standards and Technology (NIST) and the Modified National Institute of Standards and Technology (MNIST) datasets [1, 2]. Other than the two datasets, the Standard Template Library (STL)-10 dataset, Street View House Numbers (SVHN) dataset, Canadian Institute for Advanced Research (CIFAR-10) and (CIFAR-100) datasets, are among the famous and widely used datasets to evaluate the performance of a newly created model [5]. Additionally, a good pre-processing method is also important to produce good classification results [12, 13].

The above past studies have shown the importance of pre-processing methods. However, very little attention was given to compare binary and normalized pre-processed images datasets and its effects on the performance of the models. Therefore, this study aims to explore the different pre-processing methods on image datasets with several different models. The remainder of this paper is organized as follows: The next section presents the background study on handwritten images, NIST and MNIST datasets. The third section describes the image pre-processing methods for both normalized and binary datasets. The fourth section discusses the results of the experiments, and finally in Sect. 5 is the conclusion of the study.

## 2 Handwritten Images

It is a known fact that handwritten dataset has been widely utilized as a part of machine learning model assessments. Numerous model classifiers utilize primarily the digit classes. However, other researchers handle the alphabet classes to demonstrate vigor and scalability. Each research model tackles the formulation of the classification tasks in a slightly different manner, varying fundamental aspects and algorithm processes. The research model is also varied according to their number of classes. Some vary the training and testing splits while others conduct different pre-processing methods of the images.

### 2.1 NIST Dataset

The NIST Special Database 19 was released in 1995 by the National Institute of Standards and Technology [1, 2]. The institute made use of an encoding and image compression method based on the CCITT Group 4 algorithm. Subsequently, the compressed images are packed into a patented file format. The initial release of the compressed image database includes codes to extract and process the given dataset. However, it remains complex and difficult to compile and run these given tools on modern systems. Due to these problematic issues, an initiative was made as a direct response catered to the problems. A second edition of the NIST dataset was successfully published in September 2016 [2] and contained the same image data encoding using the PNG file format.

The objective of creating the NIST dataset was to provide multiple optical character recognition tasks. Therefore, NIST data has been categorized under five separate organizations referred to as data hierarchies [5]. The hierarchies are as follows:

- By Page: Full page binary scans of many handwriting sample forms are found in this hierarchy. Other hierarchies were collected through a standardized set of forms where the writers were asked to complete a set of handwritten tasks.
- By Author: Individually segmented handwritten characters images organized by writers can be found in this hierarchy. This hierarchy allows for tasks such as identification of writers but is not suitable for classification cases.
- By Field: Digits and characters sorted by the field on the collection are prepared while preserving the unique feature of the handwriting. This hierarchy is very useful for segmenting the digit classes due to the nature of the images which is in its own isolated fields.
- By Class: This hierarchy represents the most useful group of data sampling from a classification perspective. This is because in this hierarchy, the dataset contains the segmented digits and characters arranged by its specific classes. There are 62 classes comprising of handwritten digits from 0 to 9, lowercase letters from a to z and uppercase letters from A to Z. This dataset is also split into a suggested training and testing sets.
- By Merge: This last data hierarchy contains a merged data. This alternative on the dataset combines certain classes, constructing a 47-class classification task. The merged classes, as suggested by the NIST, are for the letters C, I, J, K, L, M, O, P, S, U, V, W, X, Y and Z. This merging of classifications addresses a fascinating problem in the classification of handwritten digits, which tackles the similarity between certain uppercase and lowercase letters such as lowercase letter u and uppercase letter U. Empirically, this kind of classification problems are often understandable when examining the confusion matrix resulting from the evaluation of any learning models.

The NIST dataset is considered challenging to be accessed and utilized. The limitations of storage and high cost during the creation of the NIST dataset have driven it to be stored in an amazingly efficient and compact manner. This however, has made it very hard to be manipulated, analyzed and processed. To cope with this issue, a source code is provided to ease the usage of the dataset. However, it remains challenging for more recent computing systems. Inevitably, as mentioned earlier, NIST has released a second edition of the dataset in 2016 [1, 5, 9]. It is reported that the second edition of the NIST dataset is easily accessible. However, the organization of the image datasets contained in this newly released NIST is different from the MNIST dataset. The MNIST dataset offers a huge training set of sixty thousand samples which contains ten-digit classifications. Moreover, the dataset also offers ten thousand testing samples for further evaluation of any classification models. Further discussions and analysis on MNIST dataset will be elaborated in the next section.

## 2.2 MNIST Dataset

The images contained in MNIST is a downsized sampled image from $128^2$ pixel to $28^2$ pixel. The image format of the 282 pixel MNIST dataset is an 8-bit grayscale resolution. Next, the pre-processed grey level image is centered by computing the center mass pixel. Finally, it is positioned to the center of the 282 pixel sized images resulting in the consistent formats of the MNIST dataset. The dataset is ready to be manipulated and pre-processed further for analysis and experiment. Although the original NIST dataset contains a larger sampling of 814,255 images, MNIST takes only a small portion of the total sampling as it merely covers ten classification of handwritten digits from number zero to nine. The readiness of MNIST data makes it very popular to be used as a benchmark to analyze the competency of classification models. Thousands of researchers have used, manipulated and tested the dataset which proves its reliability and suitability for testing newly created models. The easy access and widespread usage make it easier for researchers to compare the results and share their findings. Table 1 lists a few recent studies on machine learning using MNIST dataset.

**Table 1.** Similar works that used MNIST dataset as benchmark

| Author (Year) | Description of research |
|---|---|
| Shruti *et al.* (2018) | Used a network that employed neurons operating at sparse biological spike rates below 300 Hz, which achieved a classification accuracy of 98.17% on the MNIST dataset [3] |
| Jaehyun *et al.* (2018) | Using Deep Neural Networks with weighted spikes, the author showed that the proposed model with weighted spikes achieved significant reduction in classification latency and number of spikes. This led to faster and more energy-efficient than the conventional spiking neural network [4] |
| Gregory *et al.* (2018) | A research that conducted an extension to MNIST dataset. They created a new dataset that covered more classification problems. The newly created datasets was named EMNIST [5] |
| Mei-Chin *et al.* (2018) | The author performed a systematic device-circuit-architecture co-design for digit recognition with the MNIST handwritten digits dataset to evaluate the feasibility of the model. The device-to-system simulations introduced by the author indicated that the proposed skyrmion-based devices in deep SNNs could possibly achieve huge improvements in energy consumption [6] |
| Shah *et al.* (2018) | Created a handwritten characters recognition via Deep Metric Learning. The author created a new handwritten dataset that followed the MNIST format known as the Urdu-Characters with sets of classes suitable for deep metric learning [7] |
| Paul *et al.* (2018) | The author used Sparse Deep Neural Network Processor for IoT Applications which measured high classification accuracy (98.36% for the MNIST test set) [8] |
| Jiayu *et al.* (2018) | The author used Sparse Representation Learning with variation Auto-Encoder for MNIST data Anomaly Detection [9] |
| Amirreza *et al.* (2018) | Used an Active Perception with Dynamic Vision Sensors to classify N-MNIST dataset, which achieved a 2.4% error rate [10] |

## 3 Image Pre-processing

In this paper, the original MNIST dataset is created and divided into two different pre-processed datasets. The first dataset is in grayscale with normalized values while the second dataset is in grayscale with binary values. Both pre-processing methods were chosen because they allow the dataset to be converted to a low numeric value while preserving their aspect ratio. To run the experiments, MNIST dataset with two different pre-processing formats were constructed. The idea of preparing two sets of pre-processed data samples is to observe the performance of the machine learning models learning accuracy with different pre-processed images. This will help researchers to understand how machine learning behave with different image pre-process formats. The input format values of the neural network will depend on how the pre-processing of the dataset is executed. The created models will be fed with the pre-processed datasets.

### 3.1 Normalized Dataset

Each of the pre-processed data categories is segmented into ten groups of classifications. The data category is a set of ten numbers consisting of numbers varying from zero to nine with a dimension size of 28 × 28 pixels in grayscale format. Grayscale images allow more detailed information to be preserved in an image. However, the representative values of the images contain an array of values from 0 to 255. The activation of the network is expected to be slightly unstable as there will be more variation elements in the network input ranges. Thus, to prevent a high activation of the learning models, the grayscale values are normalized using a min max function with values between zero to one as shown in Eq. (1).

$$y = (x - min)/(max - min) \tag{1}$$

Figure 1 shows nine random samplings of the pre-processed MNIST dataset. This visualization shows that the min max normalization preserves the small details that belong to each individual sample. The representation of the normalized grayscale images is smoother as it preserves the features and details of the handwritten digits. Smoother images mean more details and less jagged edges. These smoother images will help the training models to learn the input patterns with a smaller input activation which is in the range of values from 0 to 1.

### 3.2 Binary Dataset

Figure 2 shows nine random samplings of the binary MNIST dataset. This visualization shows that converting the data sampling to binary format preserves the shape of the digits. However, the small details that belong to some individual samples can be seen missing. This is due to the threshold that was set at a certain value to classify two regions that belong to either 0 or 1. In this experiment, the threshold is set at 180 to preserve the shape of the digits while avoiding the data having too much noise.

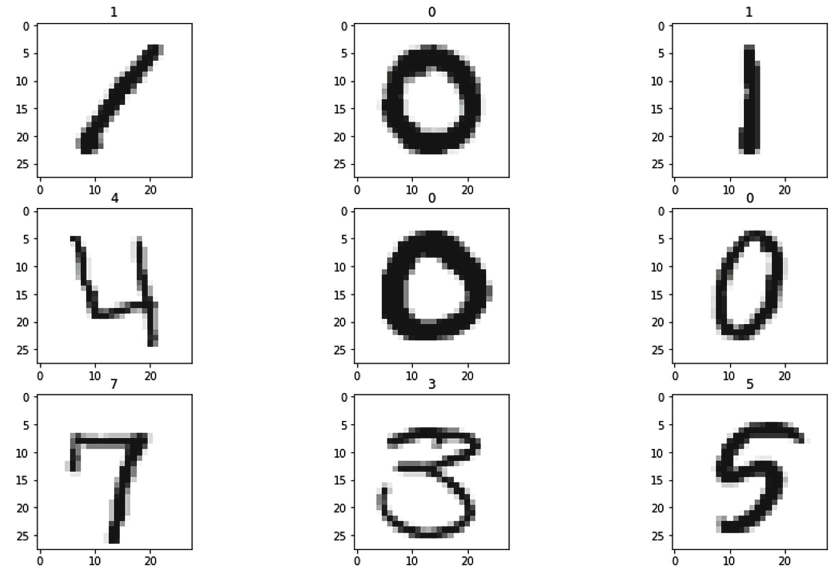

**Fig. 1.** Nine random MNIST samplings of 28 × 28 pixel dimension in grayscale format.

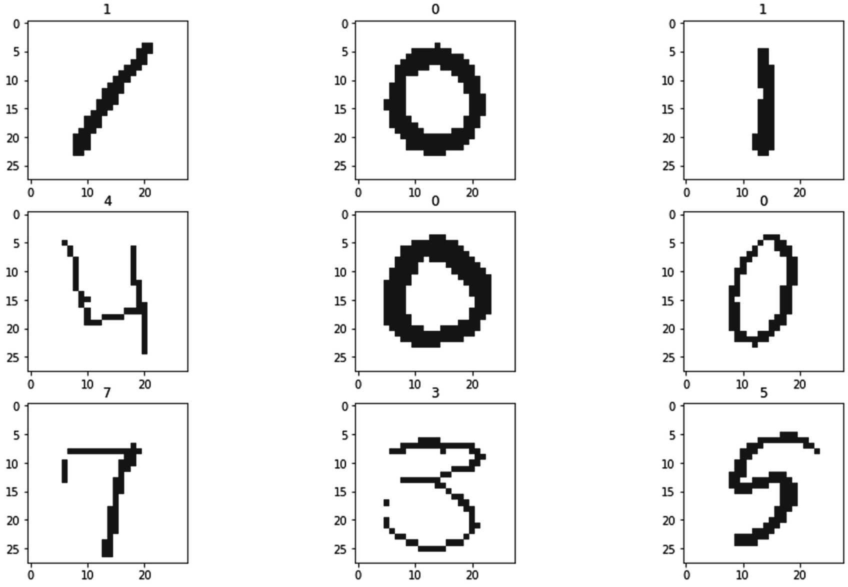

**Fig. 2.** Nine random MNIST samplings of 28 × 28 pixel dimension in binary format.

### 3.3 Machine Learning Models

The pre-processed MNIST datasets are tested with four machine learning models on both binary and normalized images. The accuracy of these models is then compared with several measures. Below are a few short explanations of the models used in this experiment.

Logistic regression is very similar to linear regression. It utilizes probability equation to represent its output classification. In short, logistic regression is a probabilistic linear classifier. By projecting an input onto a set of hyperplanes, classification is possible by identifying the input that corresponds to the most similar vector. Some research has successfully performed a logistic regression model with satisfactory accuracy [11].

Random Forest is a supervised classification algorithm that grows many classification trees [14]. Random forest is also known as random decision trees. It is a group of decision trees used for regression, classification and other task. Random forest works by creating many decision trees during training, which will produce either the classification of the generated classes of regression of an individual tree. Random forest also helps correct the possibility of overfitting problem in decision trees. By observation, a higher number of trees generated can lead to better classification. This generation somehow shows the relation of tree size with the accurate number of classification that a random forest can produce.

Extra Trees classifier, also known as an “Extremely randomized trees” classifier, is a variant of Random Forest. However, unlike Random Forest, at each step, the entire sample is used and decision boundaries are picked at random rather than the best one. Extra Trees method produces piece-wise multilinear approximations. The idea of using a piece-wise multilinear approximation is a good idea as it is considered productive. This is because in the case of multiple classification problems it is often linked to better accuracy [15].

Convolution Neural Network (CNN) is a Deep Neural Network made up of a few convolutional layers. These layers contain a pool of feature maps with a predefined size. Normally, the size of the feature maps is cut in half in the subsequent convolutional layer. Thus, as the network goes deeper, a down-sampled feature map of the original input is created during the training session. Finally, at the end of the convolution network is a fully connected network that works like a normal feed forward network. These networks apply the same concept of a SVM/Softmax loss function. Figure 3 shows the architecture of the created CNN. As depicted in the figure, the created CNN contains three convolutional layers, and two layers of a fully connected layer at the end. The last layer contains only ten outputs that use a softmax function in order to classify ten numbers.

From the input datasets as shown in Figs. 1 and 2, each dataset is supplied with the aforementioned four machine learning models. This is to test how the pre-processing of each test dataset affects the accuracy of the training and validation of the models above.

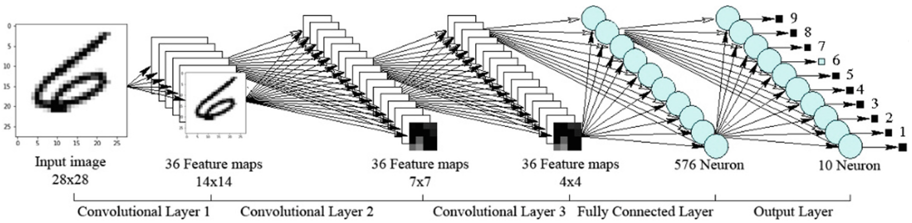

**Fig. 3.** Architecture of the created Convolutional Neural Network

## 4 Experiments and Results

In this section, we discuss the experimental results and findings from this study. All four machine learning models that were discussed earlier were both tested with the normalized and binary datasets.

### 4.1 Experimental Setup

We have set up four machine learning models to be trained with the pre-processed MNIST dataset. Each learning model was analyzed for its training and validation accuracy for both normalized and binary datasets. Further discussions on the analysis of the accuracy is explained in the next subsection.

### 4.2 Machine Learning Model Accuracy

The outcome of the experiment shows fascinating results. In both datasets, all four models would have no or minimal difficulties of training the classification of the handwritten digits. Almost all models manage to get a training and validation accuracy of greater than 90%. However, this does not mean that the errors produced by some of the models are actually good. For 4200 validation samplings, a mere 10% inaccuracies may cost up to 400 or more misclassifications.

The experiment results show that the machine learning models had misclassified some of the training and validation data. This misclassification may be due to some of the training data instances having similar features but classified with a totally different label. The misclassification issue is elaborated further in the next section. Table 2 shows CNN having the least overfitting over other training results as it has the least differences between the training and validation accuracies for both normalized and binary dataset. This is probably due to the design and architecture of the CNN itself that produces a less overfitting models as reported by [16]. Although Extra Trees shows a better training accuracy of 100%, a big difference of its validation and training results mean that there is a possible overfitting in the created model. However, Random Forest, having the highest accuracy for binary dataset of 1.9%, is slightly higher than the CNN model.

**Table 2.** MNIST model accuracy comparison

| Model | MNIST normalized | | MNIST binary | |
|---|---|---|---|---|
| | Training | Validation | Training | Validation |
| Logistic regression | 94% | 92% | 93.3% | 89.5% |
| Random forest | 99.9% | 94% | 99.9% | 91% |
| Extra trees | 100% | 94% | 100% | 92% |
| Convolution Neural Network | 99.5% | 99.4% | 90.6% | 90.1% |

### 4.3 CNN Accuracy and Loss

Figure 4 depicts the training and validation of graph patterns. A close observation of the results show that the normalized dataset generates a better learning curve. The learning of patterns is quite fast as the graph shows a steep curve at the beginning of the training. In Fig. 4(a), as the log step increases, the training and validation accuracies of the model become stable at an outstanding accuracy of 99.43%. The binary dataset shows a good validation and loss at an earlier epoch. Nevertheless, as the training continued, the CNN training model began to decline in its accuracy.

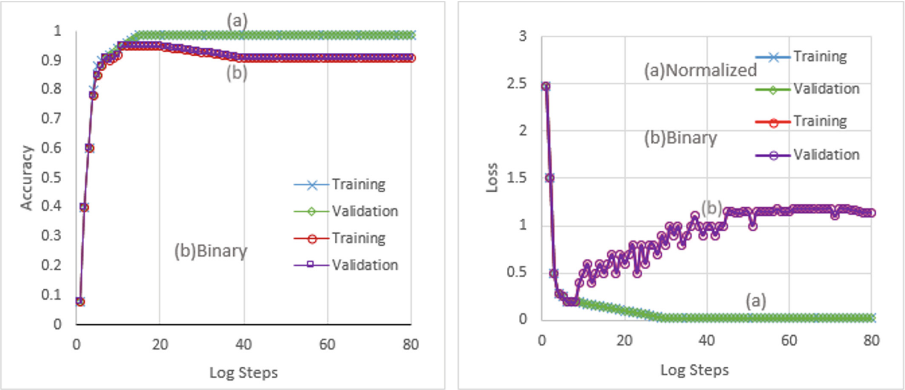

**Fig. 4.** Training accuracy & loss of (a) Normalized dataset (b) Binary dataset fed to CNN

The training declination can be seen as shown in Fig. 4(b). This declination may be caused by the noise and data loss in the binary images that make it difficult for the CNN to learn. Some features of the training and testing images were lost during the process of changing them to binary values. Further analysis of the misclassification of the CNN models of normalized datasets shows that only 24 out of 4200 validation sets are false predictors. More information in the misclassification of the handwritten digits are shown in the Table 3.

**Table 3.** CNN confusion matrix

| Predicted Digit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| True Digit | | | | | | | | | | |
| 0 | 412 | | 1 | | | | 1 | | | |
| 1 | | 470 | | | | | | | | |
| 2 | | | 420 | | | | | 1 | | |
| 3 | | | | 432 | | | | | | |
| 4 | | | | | 410 | | | 2 | | 3 |
| 5 | | | | 1 | | 391 | 1 | | 1 | 2 |
| 6 | 1 | | | | | | 431 | | | |
| 7 | | | | 1 | | | | 420 | | 4 |
| 8 | | 1 | | | | 1 | | 1 | 400 | |
| 9 | | | | | 1 | | | | 1 | 390 |
| Total | 413 | 471 | 421 | 434 | 411 | 392 | 433 | 424 | 402 | 399 |

Further investigation on the results was performed by analyzing the confusion matrix output. From the table, we can see that the CNN model is having a difficulty in classifying digit nine, having the highest misclassification rate. It is clearly stated that some numbers that should be classified as nine may be misinterpreted by the CNN models as a seven, five and four. Other examples of misclassifications are where seven is interpreted as two, four and eight. Figure 5 shows all of the false predictor images.

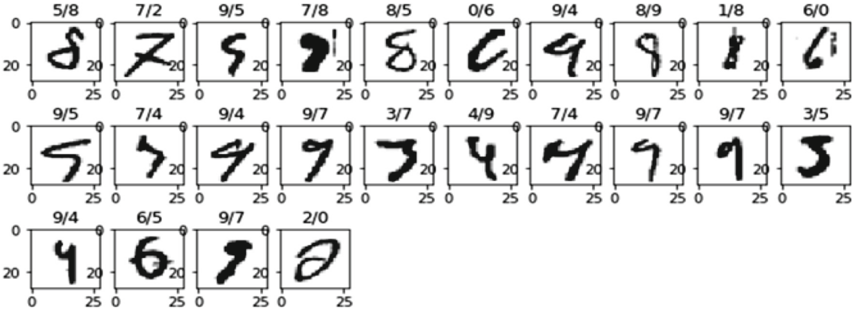

**Fig. 5.** False predictors

## 5 Conclusions

This study has demonstrated the importance of pre-processing methods prior to machine learning modelling. Two different pre-processed images namely the binary and normalized images were fed into four machine learning models. The experiments revealed that both the selection of machine learning models, with regards to the appropriate pre-processing methods, would yield better results. Our experiments show that CNN has better results with 99.6% accuracy for normalized dataset and Extra

Trees gives an accuracy of 92.4% for binary dataset. Moreover, it could also be concluded that normalized datasets from all models out-performed binary datasets. These results suggest that normalized dataset preserves meaningful data in image recognition.

**Acknowledgement.** The authors are grateful to the Research Management Centre (RMC) UiTM Shah Alam for the support under the national Fundamental Research Grant Scheme 600-RMI/FRGS 5/3 (0002/2016).

## References

1. Grother, P., Hanaoka, K.: NIST special database 19 hand printed forms and characters 2nd Edition, National Institute of Standards and Technology (2016) Available: http://www.nist.gov/srd/upload/nistsd19.pdf. Accessed 20 July 2018
2. Grother, P.: NIST special database 19 hand printed forms and characters database. National Institute of Standards and Technology, Technical report (1995). http://s3.amazonaws.com/nist-srd/SD19/1stEditionUserGuide.pdf,last. Accessed 20 July 2018
3. Kulkarni, S.R., Rajendran, B.: Spiking neural networks for handwritten digit recognition, supervised learning and network optimization (2018)
4. Kim, J., Kim, H., Huh, S., Lee, J., Choi, K.: Deep neural networks with weighted spikes. Neurocomputing (2018)
5. Cohen, G., Afshar, S., Tapson, J., van Schaik, A.: EMNIST: an extension of MNIST to handwritten letters. Comput. Vis. Pattern Recognit. (2017)
6. Chen, M.C., Sengupta, A., Roy, K.: Magnetic skyrmion as a spintronic deep learning spiking neuron processor. IEEE Trans. Mag. **54**, 1–7 (2018). IEEE Early Access Articles
7. Shah, N., Alessandro, C., Nisar, A., Ignazio, G.: Hand written characters recognition via deep metric learning. In: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), IEEE Conferences, pp. 417–422. IEEE (2018)
8. Paul, N.W., Sae, K.L., David, B., Gu-Yeon, W.: DNN engine: a 28-nm timing-error tolerant sparse deep neural network processor for IoT applications. IEEE J. Solid-State Circuits **53**, 1–10 (2018)
9. Jiayu, S., Xinzhou, W., Naixue, X., Jie, S.: Learning sparse representation with variational auto-encoder for anomaly detection. IEEE Access, 1 (2018)
10. Amirreza, Y., Garrick, O., Teresa, S.G., Bernabé, L.B.: Active perception with dynamic vision sensors. minimum saccades with optimum recognition. IEEE Trans. Biomed. Circuits Syst. **14**, 1–13 (2018). IEEE Early Access Articles
11. Yap, B.W., Nurain, I., Hamzah, A.H., Shuzlina, A.R., Simon, F.: Feature selection methods: case of filter and wrapper approaches for maximising classification accuracy. Pertanika J. Sci. Technol. **26**(1), 329–340 (2018)
12. Mutalib, S., Abdullah, M.H., Abdul-Rahman, S., Aziz, Z.A: A brief study on paddy applications with image processing and proposed architecture. In: 2016 IEEE Conference on Systems, Process and Control (ICSPC), pp. 124–129. IEEE (2016)
13. Azlin, A., Rubiyah, Y., Yasue M.: Identifying the dominant species of tropical wood species using histogram intersection method. In: Industrial Electronics Society, IECON 2015-41st Annual Conference of the IEEE, pp. 003075–003080. IEEE (2015)

14. Bernard, S., Adam, S., Heutte, L.: Using random forests for handwritten digit recognition. In: Proceedings of the 9th IAPR/IEEE International Conference on Document Analysis and Recognition ICDAR 2007, pp. 1043–1047. IEEE (2007)
15. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. **63**, 3–42 (2006). Engineering, computing & technology: Computer science
16. LeNet-5, convolutional neural networks, http://yann.lecun.com/exdb/lenet/. Accessed 20 July 2018