Comparison of Autoencoder and Principal Component Analysis Followed by Neural Network for E-Learning Using Handwritten Recognition

Jasem Almotiri and Khaled Elleithy
Department of Computer Science and Engineering
University of Bridgeport
Bridgeport, USA
jalmotir@my.bridgeport.edu, elleithy@bridgeport.edu

Abdelrahman Elleithy
Department of Electrical Engineering and Computer Science
Texas A&M University-Kingsville
Kingsville, Texas
Abdelrahman.Elleithy@tamuk.edu

***Abstract*—This paper presents two different implementations for recognition of handwritten numerals using a high performance autoencoder and Principal Component Analysis (PCA) by making use of neural networks. Different from other approaches, the non-linear mapping capability of neural networks is used extensively here. The implementation involves the deployment of a neural network, and the use of an auto encoder and PCA while carrying out the compression and classification of data. The performance of the system was analyzed, and an accuracy of 97.2% for Principal Component Analysis, and 98.1% accuracy for the autoencoder, was recorded in detection of numerals written by school children.**

***Keywords*—autoencoder, PCA, neural networks, image processing, machine learning, data MNIST.**

I. INTRODUCTION

Handwritten Digital Recognition (HDR) is extremely important in daily life, and has many uses. However, HDR faces many challenges, such as the fact that individual students write numbers in different ways [1][2]. HDR is a hot topic in machine learning and artificial intelligence. Additionally, there is much research on techniques to solve the issue [3].

Machine learning algorithms have always been a topic of great interest for research. The major hindrance to further development is that data sets collected from raw data or real-world data have a very high dimensionality. This is another topic of research, and several solutions have been proposed for this.

Data collected from the real world is remodeled in a space which has fewer dimensions than the original space. The resultant output has the same structure as the input. Missing Values Ratio, Low Variance Filter, High Correlation Filter, Random Forests, Backward Feature Elimination, and Forward Feature Construction are some techniques that have been proposed to solve this issue.

We have chosen Principal Component Analysis (PCA) and high performance autoencoder to tackle the problem of efficiently recognizing handwritten numerals. Details regarding this approach follow.

Existing autoencoders and those developed through recent research are incapable of mocking the original structure of data under analysis [4]. Mocking in this scenario means replicating the structure of original data. This is because these approaches try to handle only the variance of resulting structure. After completing a round of deep analysis, it was concluded that mocking the original structure and handling the variance can be solved by creating a new autoencoder structure which makes use of an "unsupervised learning" technique.

We used this newly developed technique to train the autoencoders, and the resulting autoencoders were used to analyze real-world data structures. In addition, the concept of parallel computing was introduced into the method, by which the work was divided amongst n processors. The complete analysis was finished in $1/n$ of the actual time that it should have taken. The actual time values are drawn out from peer researches like [5]. Parallel computing along with backpropagation in data Mixed National Institute of Standards and Technology (MNIST) is carried out.

The solution is developed for the base problem wherein there are shortcomings in the ability of an image recognition system to recognize digits scribbled by a human being such as a child. Trying to solve this, there are several other related problems that come up when

modeling the data, training the autoencoder, and other aspects [6]. A well-written number might be easily identifiable by a computer or an image processor, but this system might fail based on neatness in writing, texture of the material on which the data is presented, and several other factors. With the development of a new system which can be used globally for a number identification purposes, a major hindrance to developing applications which interact with users in this area is removed.

II. Related Work

There has been much research on the use of neural networks and recognition processes which use them. Quality work has been done on computing the relational differences between a pair of models using neural networks in [7]. This paper concerns itself with recognizing whether or not two digits differ from each other by carrying out analysis on digits obtained from the MNIST. In [8], ImageNet dataset is used to tackle the problem of obtaining expensive class labels and the hardships associated with high-scale data classification. However, this approach suffers from low accuracy levels of around 52%. This can be considered one of the studies that motivated the study of parallel computing methodology.

As mentioned in the Introduction, there is a plethora of research available on semi-supervised learning tasks. Work done in [4] this approach produced the lowest error rates, in spite of the requirement of huge amount of resources. Using neural networks, a significant improvement was made over the results from [9] using fewer resources and a less-complex methodology. [10] deals with visual recognition and restoration. The authors propose a new framework where non-linear mapping, the same approach we follow in this paper, is used. The backpropagation technique is also used.

III. Solution Outline

Fig. 1 depicts the overall design of the proposed system. A raw image is fed into the system, which is already trained, and ready to analyze the incoming image. Then the required attributes are extracted and then fed into the classifier. In this research, neural networks are used as the classifier.

A neural network consists of a set of units, which essentially is a set of neurons, stacked on each other as layers. This combination of layers will convert an input vector into an output vector based on the conditions/method specified. Each unit in the network will take the input, process the data using a non-linear method, and pass the output to the next unit in the network. There will be *n* layers in the network, based on the architecture. Another point to note here is that the input layer will consist of multiple units, while the output/final layer will have a single unit which will provide the processed input.

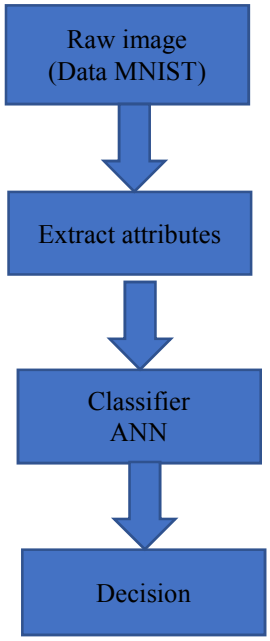


Figure 1: Design of proposed technique

Autoencoders are artificial neural networks. These networks are used in this research for "unsupervised learning." Unsupervised learning gives the network no indication as to what the output should be. While the training is done, the data supplied to the learner is not labeled. Instead of relying on knowledge of what the output should look like, we are relying on the relationship between variables in the supplied input. This approach is not completely dependent on clustering. While you receive feedback based on the expected output in the case of supervised learning, with autoencoders you do not receive any feedback.

In this scenario, autoencoders are used for learning a set of data and for achieving dimensionality reduction. Dimensionality reduction convert the given input into a minimal representation form in which data is still minimized but retains the variance of the supplied data, about which we are concerned [8]. In other words, it is the process of breaking down data, extracting the required content without losing its essence, and mirroring it in the minimal form. However, choosing the best architecture for the auto encoder is still a topic of study [11].

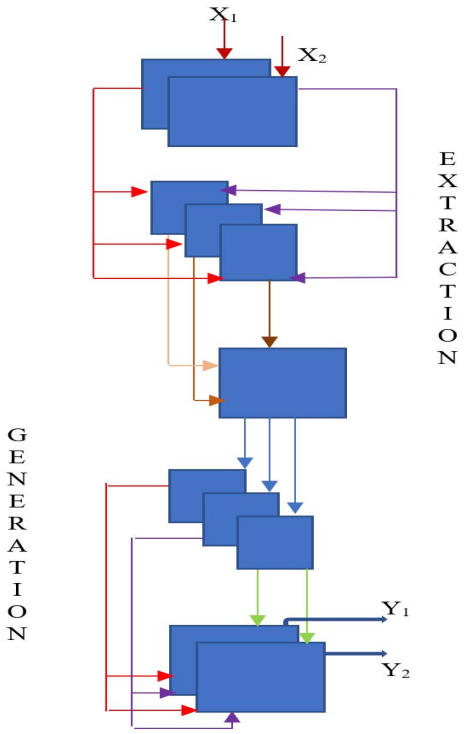


Figure 2: Breakdown of auto encoding

In Fig. 2, the process of auto encoding is represented. A nonlinear method is used to process the input, which converts the straight-lined principal components to nonlinear curves. The main point to note here is that the new representation created by the autoencoder will be in a space of fewer dimensions than the original space. Another point to note is that the autoencoder output will be equal to the input. This technique is called identity mapping.

On deeper analysis of Fig. 2, it is observed that the middle layer enforces dimensional reduction. This layer plays the role of a "bottleneck" enforcer. It also gives the "component values", which will be used later. In further layers, the data is converted again and the output Y_1 and Y_2 are given as identical to the supplied input, X_1 and X_2. In other words, the "EXTRACTION" layer carries out the encoding, the "BOTTLENECK" layer does the processing, and the "GENERATION" layer does the decoding. The middle, or "bottleneck," layers are also hidden.

Another methodology used here is the k's Nearest Neighbor (KNN) algorithm. With this algorithm, the neighboring classes of the node under test can be found. In our case this is used in classification. The result from this classification will be a class type. The class type is chosen as the majority of the classes of a particular type from the neighboring classes in a specified radius. *k* is the number of neighbors that are required from completing the classification. KNN falls under the category of lazy learning algorithms, which also means that the training will be short. The confidence level of the result is based on the majority level of output type. If *k* number of classes of same type are found in the neighbors, the confidence level is high. In other words, the confidence level is proportional to the number of classes of the same type.

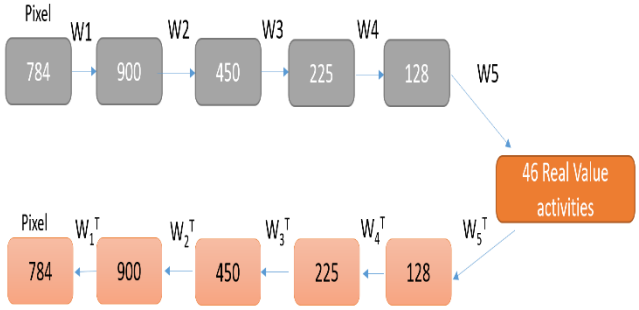


Figure 3: Steps involved in image recognition

Fig. 3 shows the working of a deep autoencoder with raw image from data MNIST [12] that has 784 pixels, which is encoded via four hidden layers, into 46 real valued activities. Then, we decode those 46 real valued activities back to the raw image that has 784 pixels, and reconstruct it using a weight matrix by transposition of the weight. Finally, we apply backpropagation.

Principal Component Analysis [13] was the other technique implemented in the comparison. The core difference between PCA and auto encoder lies in the way they carry out dimensionality reduction. In PCA, the encoders and decoders are linear methods, while in autoencoders, they can be either linear or non-linear. In other words, an auto encoder with one layer of neural network and T(i) the transfer function, which is [Y(i)/X(i)], can be considered similar to a PCA.

It is also worth noting that the "bottleneck" layer in an autoencoder might have more dimensions than input, which is the opposite of dimensionality reduction. However, the simulation was carried out using PCA also, and the results were analyzed.

After setting up the environment, the first phase included training with the help of a data MNIST [12]. The MNIST database holds 60,000 samples of images for training the system, and another 10,000 for testing the system. Parallel processing capability was made use of in the training phase, which reduced the training time considerably.

IV. RESULT

The results collected are tabulated Tables 1 and 2.

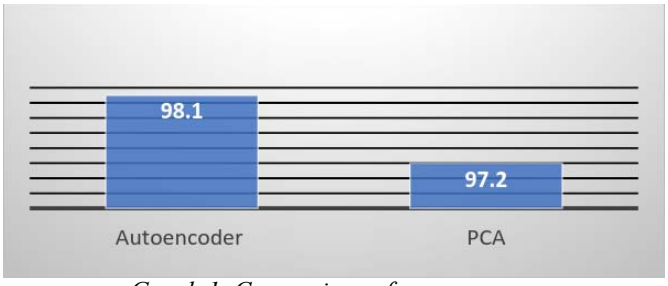
Samples for Training	40,000
Samples for Testing	6,000
PCA Accuracy (%)	97.2
Time taken by PCA(hour)	Less 1

Table 1. Results of simulation- PCA

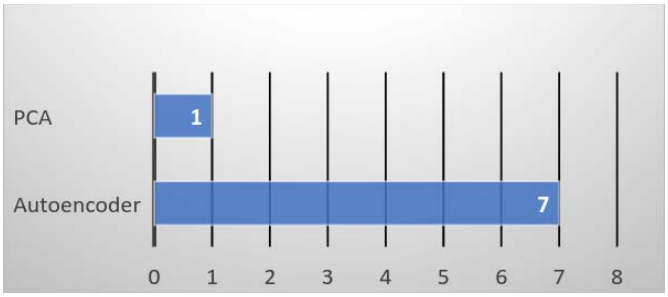
Samples for Training	40,000
Samples for Testing	6,000
Autoencoder Accuracy (%)	98.1
Time taken by Autoencoder (hours)	9

Table 2. Results of simulation- Autoencoder

In this scenario, the simulation was carried out with 40,000 samples for training and 6000 samples for testing. The system used parallel processing and showed considerable improvement over the single-core system. With a larger number of cores, the speed of processing can be increased on an exponential factor. This will be an area for future research.



Graph 1. Comparison of accuracy



Graph 2. Comparison of time taken

The PCA showed an accuracy of 97.2%, which is better in some cases than the other methodologies used in the industry [14] [15]. PCA was completed in roughly less one hour. With the same scenarios, autoencoder could achieve accuracy of 98.1, and it took around 9 hours to complete. The improvement in execution time makes PCA a better candidate than auto encoder.

Another test for future research would be to increase the number of samples for training and testing. By doing this, a relation can be drawn between the number of samples, accuracy, and the time taken for processing. Using more samples from the MNIST database and utilizing a processor with a higher degree of parallelism would help extend the scope of the research.

References

[1] Tai-Shan Yan, Yong-Qing Tao and Du-Wu Cui, "Research on handwritten numeral recognition method based on improved genetic algorithm and neural network," *2007 International Conference on Wavelet Analysis and Pattern Recognition*, Beijing, 2007, pp. 1271-1276.

[2] R. E. a. M. Jampour., " Efficient Handwritten Digit Recognition based on Histogram of Oriented Gradients and SVM," *International Journal of Computer Applications*, vol. 104, pp. 10-13, October 2014.

[3] C. L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: Benchmarking of state-of-the-art techniques," *Pattern Recogniion.*, vol. 36, no. 10, pp. 2271–2285, 2003.

[4] R. Tachibana, T. Matsubara and K. Uehara, "Semi-Supervised learning using adversarial networks," *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, Okayama, 2016, pp. 1-6.

[5] R. Gu, F. Shen and Y. Huang, "A parallel computing platform for training large scale neural networks," *2013 IEEE International Conference on Big Data*, Silicon Valley, CA, 2013, pp. 376-384.

[6] Yan Yin, JunMin Wu and HuanXin Zheng, "Ncfm: Accurate handwritten digits recognition using Convolutional Neural Networks," *2016 International Joint Conference on Neural Networks (IJCNN)*, Vancouver, BC, 2016, pp. 525-531.

[7] T. Du and L. Liao, "Deep Neural Networks with Parallel Autoencoders for Learning Pairwise Relations: Handwritten Digits Subtraction," *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, Miami, FL, 2015, pp.582-587.

[8] Y. Lin *et al*., "Large-scale image classification: Fast feature extraction and SVM training," *CVPR 2011*, Providence, RI, 2011, pp. 1689-1696.

[9] G. Chen, Y. Li and S. N. Srihari, "Joint visual denoising and classification using deep learning," *2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, 2016, pp. 3673-3677.

[10] M. Naseer and S. Y. Qin, "Performance Comparison of Nonlinear Dimensionality Reduction Methods for Image Data Using Different Distance Measures," *2008 International Conference on Computational Intelligence and Security*, Suzhou, 2008, pp. 41-46.

[11] C. C. Tan and C. Eswaran, "Reconstruction of handwritten digit images using autoencoder neural networks," *2008 Canadian Conference on Electrical and Computer Engineering*, Niagara Falls, ON, 2008, pp. 000465-000470.

[12] L. Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Research," in *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141-142, Nov. 2012.

[13] A. Kane and N. Shiri, "Selecting the Top-k Discriminative Features Using Principal Component Analysis," *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, Barcelona, Spain, 2016, pp. 639-646.

[14] C. Hu, X. Hou and Y. Lu, "Improving the Architecture of an Autoencoder for Dimension Reduction," *2014 IEEE 11th Intl Conf on Ubiquitous Intelligence and Computing and 2014 IEEE 11th Intl Conf on Autonomic and Trusted Computing and 2014 IEEE 14th Intl Conf on Scalable Computing and Communications and Its Associated Workshops*, Bali, 2014, pp. 855-858.

[15] P. de Chazal, J. Tapson and A. van Schaik, "A comparison of extreme learning machines and back-propagation trained feed-forward networks processing the mnist database," *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, South Brisbane, QLD, 2015, pp. 2165-2168.