A Comparative Study of SVM and MLP on MNIST Digit Dataset

Chadi El-Hajj and Muaaz Bin Sarfaraz

Abstract—In this paper we present a comparative analysis of two famous methods in neural networks: Multilayer Perceptron and Support Vector Machines. We describe, experiment and compare the results of these methods with respect to their accuracy and time complexity on a subset of the well-known MNIST handwritten digit recognition classification task. A grid search of 3-fold cross-validation is performed to select the best fine-tuned model for each method. The performance of SVM with linear kernel was slightly better than MLP on the holdout test set. Moreover, one hidden layer perceptron of 50 neurons surprisingly gave the best result at a high learning rate of 0.9. This proves the complexity of error surface and shows that shaking the system with high learning rate with coincidental good initial weight selection could lead to good models.

1 Introduction and Motivation

Over the last decades, neural computing has been a great influence in the success of speech and image recognition. Several years ago, it was impossible to implement a precise pattern recognition system by computer, thus it was mostly combined with hand-crafted features and automatic learning techniques. In this report, we will use different neural computing techniques and build models with limited or no human intervention to classify/predict labels of handwritten digits. This means that, the models will rely on available data to learn and extract features on their own for better classification accuracy rather than hand-crafted feature engineering [12].

The recent technological advancement, low cost and powerful high speed machines made it possible to depend on computational power to train neural networks using back-propagation for critical image recognition problems. Furthermore, advanced machine learning methods enabled us to work with high dimensional data sets to solve complex issues. The combination of powerful machine learning methods and the availability of large data sets enabled scientist to improve the accuracy of image and speech recognition. A typical critical application is optical character recognition (OCR). For example, LeNet is a commercial system heavily employed in the USA for check recognition used in the banking industry. This system reads several million checks per month used by many banks across the country [12].

In this report, we will build two neural network models trained on sub-set of MNIST training dataset and test it on a separate unseen set of examples. The methods used for this classification task are support vector machine (SVM) and single hidden layer perceptron (MLP) feed-forward artificial neural network trained with momentum and back-propagation. Additionally, we will evaluate, compare and analyse the performance of these methods based on how accurately they were able to predict the labels of the handwritten numbers.

2 Dataset

The data chosen for this study is a subset of samples from MNIST, a widely used dataset to perform analysis of image recognition using neural network methods. MNIST is collection of binary images of handwritten digits derived from NISTs databases, Figure 1 shows examples of MNIST images. The original NIST dataset was split into SD-3 for training and SD-1 for testing. SD-3 contains images collected from Census Bureau employees while SD-1 consists of images collected from high-school students. However, MNIST was built from both SD-3 and SD-1. The dataset comprise 60000 rows of training samples and 10000 test samples from 250 writers. Moreover, the writers of the training set and test set are different. This means that the images of handwritten digits in the test set have completely different handwriting styles [13].

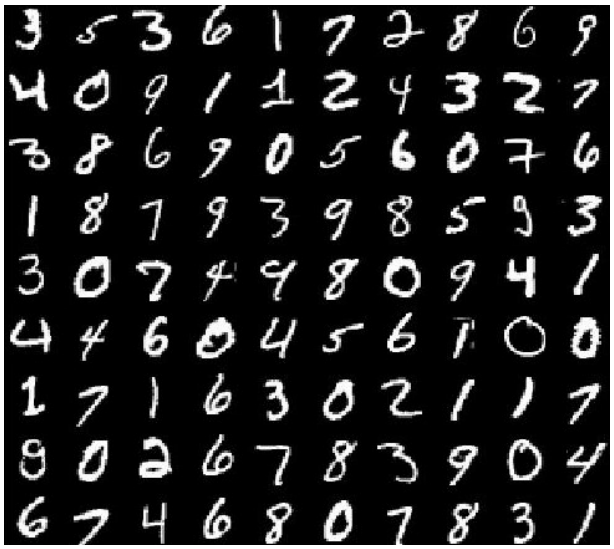

Fig. 1. Example of MNIST handwritten digits

We randomly selected 6000 observations from the training set and 1000 observations from the test set. Images in the dataset are 28x28 pixels wide. Each image was transformed into 784 pixels, forming 784 dimensions where each point corresponds to particular pixel. The variables are continuous values range between 0 and 255, Figure 2 below represents this transformation. Additionally, each image represent a single digit, between 0 to 9, therefore, we have 10 classes 0,1,2,3,4,5,6,7,8,9.

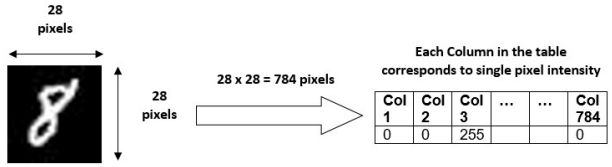

Fig. 2. Diagram shows a 28x28 image transformation to 784 dimensions. Each pixel is stored in one column with values between 0 to 255

We have two separate datasets for training and testing. Also, both datasets were already size normalised and centred in a fixed size image and transformed into pixels. Therefore, the data was already prepared and no data pre-processing was needed.

Since MNIST consists of 784 variables, which makes it a high-dimensional dataset. Therefore, as initial investigation, we applied Principal Component Analysis (PCA) and t-distributed stochastic neighbour embedding (T-SNE) for the sole purpose of visualising the data and

study whether the projection of data into two dimensional space is linearly separable.

PCA: is a statistical method for dimension reduction. It uses an orthogonal transformation to change a set of input variables into uncorrelated principal components. The number of principal components is less than or equal to the number of dimensions/variables. PCA tries to preserve as much information as possible by placing the component with the highest variance as the first principle component followed by the components with the highest possible variance (in a descendant order). PCA is employed to find patterns in the data and identify similarities, thus it can be used for exploratory data analysis [19].

T-SNE: is a machine learning algorithms for dimensionality reduction. T-SNE is a nonlinear dimension reduction technique used to convert high dimensional data and visualise it into a two or three dimensional space. It classifies objects based on dimensional points, similar data points are modelled by the surrounding data point, and otherwise it uses distant points to model objects [14].

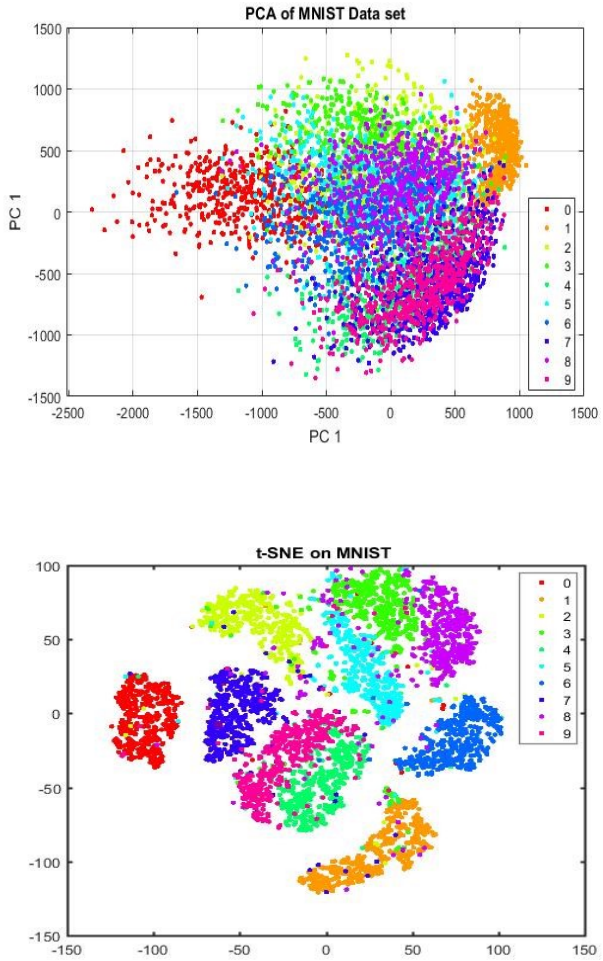

Fig. 3. Top: PCA projection into two dimensional space showing that data points are not linearly separable. Bottom: t-SNE projection and non-linear separation of data point.

Figure 3 top, shows that when PCA was applied, the projected data points appear to be not linearly separable, and many classes are overlapping, apart from digit 0 and digit 1 which are clearly distinguishable in PCA. Therefore, t-SNE was employed on the data point to support the nonlinear dimensionality reductions and visualise the projection into two dimensional space Figure 3 bottom. T-SNE figure revealed that it is possible to draw a separation boundaries after dimension reduction. It also shows that same digits have similarities and thus clustered together. For example zeros represented by red dots are grouped together and ones shown in orange forms different groups etc. This suggests that dataset is linearly separable in the high dimensional space since t-SNE tries to maintain the high dimensional distance mapping.Thus, we expect SVM with linear kernel to give decent results.

METHODS

This section will introduce the two neural network models, SVM and MLP, used in this report to predict the labels of each image. We will discuss advantages and disadvantages of each model along with the optimised hyper-parameters chosen for each one. We hypothesise that SVM should give better results than MLP, however it will take more computational time.

2.1 MLP

First method used in classification of MNIST digits is Feed-forward Neural Network, also known as Multi-Layer Perceptron. MLP maps input data to a set of targets through interconnected nodes with weighted links, example presented in the Figure 4. The weights represents the probability of neuron been triggered by an activation function e.g. sigmoid. Adding hidden layers between input and output layer gives the flexibility of estimating complex target function [4].

It has been proven that even a single hidden layer with sigmoid activation function is capable of predicting any target function, thus making MLP a very powerful technique [20]. Multi-layer perceptron perform exceptionally well at learning features without the need of feeding the features manually. However, it can easily over-fit and it is often recommended to do early stopping to avoid it [3]. Moreover, feed-forward networks are a black box with no interpretability and requires fine tuning of several hyper-parameters (eg: Hidden Layer Size, Number of layers, learning rate, momentum, epochs, activation function, initial set of weights, weight regularization etc)

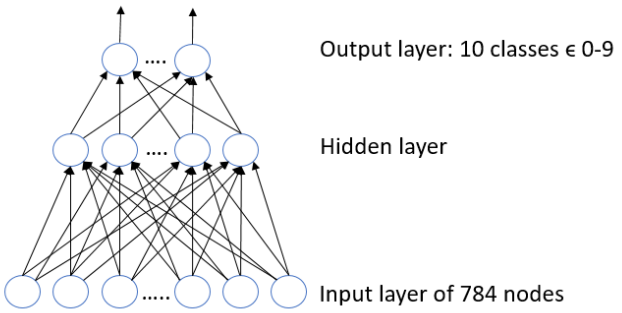

Fig. 4. MLP example with single hidden layer

2.1.1 Architecture and Parameter for training

A Feed-forward network with single hidden layer is trained using back-propagation algorithm with momentum but it can still converge at local minima instead of finding a global minima [3]. Batch mode is used for updating the weights after passing through the whole training set. The optimization is based on cross entropy error and the effectiveness is known by monitoring the misclassifaction error.Activation function for hidden layer is tansig and a softmax output layer. The following stopping criteria is used in training:

- The maximum number of epochs (as specified in grid search) is reached.
- Validation error increased more than 6 times consecutively since the last drop.

2.1.2 Evaluation: Grid Search, cross-entropy, 3-fold cross-validation and Accuracy

The following set of hyper-parameters are tested to find the best model:

- Number of layers: 1
- Number of hidden neurons: 25 50 100 200
- Epochs: 100 300 700
- Learning rate: 0.01 0.9
- Momentum: 0.1, 0.9

2.2 SVM

The second classification methods employed is SVM. It is a popular supervised learning technique used for classification and regression [8]. The main objective of SVM is to find the perfect linear decision line that separate classes in high-dimensional space in such a way that the gap, margin hyperplane, between classes is maximised. The maximum margin hyperplane offers good generalisation by choosing the right parameters which also make it more robust even with bias in the data. SVM maps each point in the training examples to either one of the binary class output, and this can be achieved with a relatively small subset of training data [2].

SVM can be applied in cases where the data is linearly separable (hard margin) or non-linearly separable (soft margin). Kernel representation defines an implicit non-linear transformation, and works with inner product instead of full feature vector space. The kernel projects the input data into a higher dimensional space which in turn allows the classes to be separated. Using kernel, adds more flexibility for SVM in terms of the form of threshold for good separation [1]. Moreover, SVM is guaranteed to find a global minimum through multiple passes, due to the fact that it is defined by convex optimisation [8].

Although SVM is a powerful classification technique, it has many disadvantages. A common drawback is that training on a large dataset is computationally expensive. Also, with more noise in the dataset, SVM does not perform well i.e. output class overlaps [1].

SVM by nature is a binary classifier, however, MNIST is a multi-class domain, therefore a one-versus-one classifier was used and the data point will be classified based on the majority vote of all classifiers [16]. Before training an SVM model we would also centralize the data around its mean and scale to unit standard deviation. It is a recommended approach to avoid a dimension with large range being considered more significant that features having smaller ranges [11].

2.2.1 Evaluation: misclassification error

The following set of hyper-parameters are tested to find the best model:

- Kernel: Linear , Radial basis function
- Box Constraint: 0.05 0.075 0.1 0.5
- Kernel Scale:1 7.5 10 15

Cross validation result based on classification error and accuracy is used to shortlist the best model which is then used for predicting the test set labels.

2.3 Autoencoder

In Scholkopf [5] , artificial examples were generated, known as virtual support vectors, by invariant transformation of the existing support vectors. This means the generated support vectors are approximately similar to the original support vectors. The reported error of their experiment was 0.8. In this report, we use a similar concept that generates new output data which will feed the neural network method.

we further attempted to improve the accuracy of the best SVM model by applying sparse autoencoder on the original training dataset. The approximated output of the autoencoder is than used as input for SVM model. It is an unsupervised learning technique which tries to learn the important features by applying constraint on the network. In our case, restriction is applied to the number of neurons in the hidden layer with Sparsity regularization factor of 4. Network has to predict the same number of output nodes as input nodes using only 100 neurons in the hidden layer as shown in Figure 5. [17].

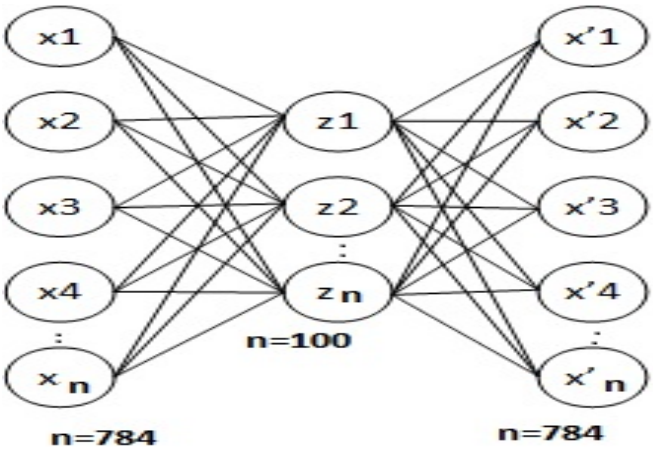

Fig. 5. Sparse Autoencoder Structure

- Number of layers: 1
- Number of hidden neurons: 100
- Weight regularization: 0.004
- Sparsity Regularization: 4
- Sparsity Proportion: 0.15
- Maximum Epochs: 10

Instead of performing a grid search for selecting the optimal Autoencoder hyperparameters, the default setting shown above are used with the training set to learn the structure in the data.

3 Results, Analysis and Evaluation

In order to find the optimal parameters, we used grid search on the training set and based on the results the best model was selected. The model will be used to evaluate the set of unseen example in the test set.

3.1 Gird Search

A combination of 48 different parameters set were tested to find the best hyper-parameters for MLP. The results of the grid search shows that, increasing the number of epochs and number of neurons in the hidden layer consistently increased the accuracy with small learning rate and momentum values Table 1. Unlike a small value momentum, a very high value makes it inconsistent and difficult for the model to learn properly since it is resistant to local minimum [9].

For the learning rate, with a high value of 0.9, the combination of 50 neurons in the hidden layer, with 700 epochs, and 0.1 momentum surprisingly gave the best results with 5.33 percent classification error. Further analysis showed that the training actually terminated with 6 Epochs Figure:6 Left, because random initial weights coincidentally gave a good result and training stopped after six consecutive increases in the validation error satisfying our early stopping criteria mentioned in 2.1.1 . A large learning rate shakes the model in an attempt to find the global minimum. Although we have different models with different number of neurons, it was interesting to notice that the best model was produced by only 50 neurons in the hidden layer Table 2.

During the analysis, we noticed that, on one hand, using a small number of epochs and neurons in the training resulted in an under-fitted model. On the other hand, a large number of epochs and neurons

Hidden Size	100 Epochs	300 Epochs	700 Epochs
25	0.566	0.334	0.202
50	0.427	0.306	0.178
75	0.398	0.222	0.153
100	0.306	0.192	0.135

Table 1. MLP Grid Search showing Error Rate with Learning rate set at 0.01 and Momentum at 0.1

Hidden Size	100 Epochs	300 Epochs	700 Epochs
25	0.693	0.374	0.442
50	0.115	0.817	0.533
75	0.183	0.902	0.259
100	0.927	0.701	0.911

Table 2. MLP Grid Search showing Error Rate with Learning rate set at 0.9 and Momentum at 0.1

resulted in an over-fitted model. Therefore, early stopping criteria was used to avoid over-fitting and produce the best results.

A limitation of MLP is that, there are so many different combinations of parameters, function, and methods that can be employed to find the best model. Unfortunately, we were able to test with relatively small set of combinations that were mentioned in the previous section.

Unlike MLP, SVM is a non-parametric methods and grid search of two parameters was performed, i.e. box constraint and kernel scale. The kernels that were tested are linear and radial basis function (RBF).

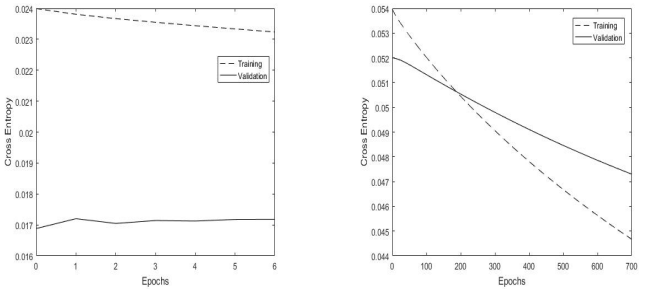

Fig. 6. Left: The figure is for best Feedforward model with 700 Epochs,50 neurons in hidden layer,Learning rate 0.9 and Momentum 0.1 but Early Stopping Criteria finished training at 6 epochs. Right:The next figure is of Feedforward net trained with 700 Epochs, 50 neurons in hidden layer, Learning rate 0.01 and momentum 0.1.

Read footnote for more information on errors mentioned in Tables[1]

SVM was more computationally expensive in comparison to MLP as we hypothesised. Additionally, training the SVM model using RBF was at least twice the computational time compared to the linear kernel. However, in LeCun (1998) paper on MNIST dataset, reported that RBF kernel yield to the best performance. On contrary, our linear SVM proved to be very efficient on both training and test [6, 15] and produced the best result using a small subset of MNIST. This could suggest that RBF requires a larger data set for better performance or detailed exploration in the grid search.

The parameters set for the best SVM model were: 0.5 box constraint, linear kernel function and 10 kernel scale. The lowest error achieve in SVM using RBF was 17.9 percent compared to 8 percent using linear kernel. It is worth mentioning, that the largest error achieved by the

[1]Note: The error shown in the table with respect to the number of epochs is the maximum Number of epochs allowed. However, training can stop if other stopping criteria is satisfied as mentioned in 2.1.1

Box Constraint	1 KS	7.5 KS	10 KS	15 KS
0.05	0.089	0.089	0.099	0.119
0.075	0.091	0.086	0.093	0.106
0.1	0.085	0.083	0.089	0.101
0.5	0.087	0.085	0.080	0.083

Table 3. SVM misclassification error with linear kernel, Box Constraint against Kernel Scale (KS). Lowest error achieved using kernel scale 10 and box constraint 0.5

Box Constraint	1 KS	7.5 KS	10 KS	15 KS
0.05	0.888	0.852	0.714	0.334
0.075	0.888	0.857	0.672	0.312
0.1	0.888	0.851	0.618	0.289
0.5	0.888	0.662	0.385	0.179

Table 4. SVM misclassifcation error with Radial basis function(RBF) shown using Box Constraint against Kernel Scale (KS). Lowest error achieved with RBF kernel using Kernel scale 15 and Box constraint of 0.5.

linear kernel 11.8 percent was better than the lowest error achieved by RFB kernel Table 3 & Table 4. In both cases, we reached the best accuracy using the highest box constraint value in our grid search. These results are logical because higher box constraint value leads to a strict separation of data by increasing the cost of misclassifying the object [7].

3.2 Autoencoder

The trained autoencoder was not validated, since it is an unsupervised learning technique that was only referred for feature extraction. The weights were also plotted to see what the auto-encoder has learnt but the features do not seem recognisable by human eye as shown in Figure 8 , actual images are shown in Figure 7 for comparison.

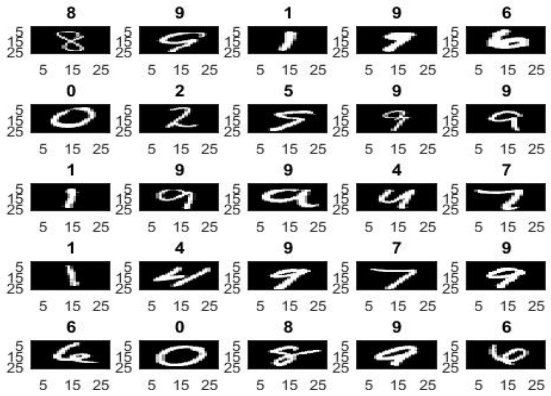

Fig. 7. Actual handwwritten images with labels on Top

4 Test set evaluation

In order to find the best generalised model, we selected the optimised models with the lowest error during the training. These models will be tested on the holdout examples and the performances will be compared.

The final model MLP gave an error of 9.5 percent on test set. Although, the user must be aware that training the same net again with random weights would give high variation in error rate. Only a particular set of initial random weights gave such exceptional accuracy with 50 neurons.

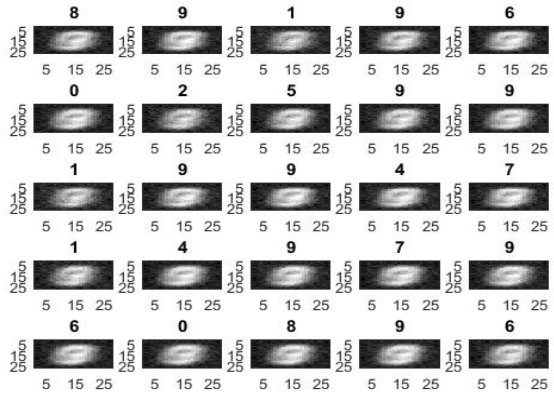

Fig. 8. Features learnt by Autoencoder, labels shown on top of each image

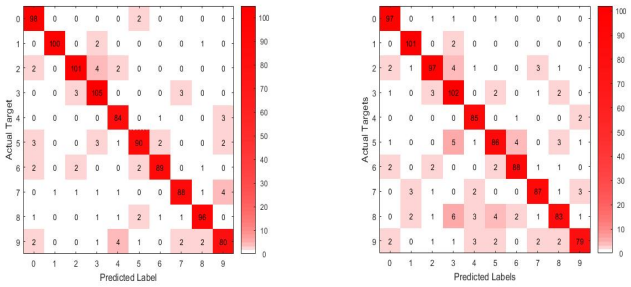

Fig. 9. left to right. This figure shows both SVM and MLP misclassified points on the test set

Figure 9 right, shows the misclassified examples using MLP model. It shows that the most misclassified class was class 8 with 20 wrong predictions and the least misclassified class was class one. Overall, out of 1000 examples only 95 were misclassified, therefore, this model is very well generalised with a reasonable accuracy.

For the SVM final model, the results were slightly better and more consistent than MLP with an error of 8.233 percent. This model with linear kernel, box constraint of 0.5, and kernel scale of 10 is well generalised on the test set. Again, running SVM using linear kernel showed the best results just like during training. Variation of the initial settings i.e. box constraint and kernel scale had little effect on the error. Also, an advantage of SVM is that, it generates good results even without domain knowledge.

SVM outperformed MLP on the holdout test and thus was that model used for classifying the output from the autoencoder. Both the test set and training set were preprocessed by passing the input through the autoencoder. The reconstructed training set was used for training the SVM model with exactly the same parameters that had the best cross validation result. The reconstructed test set was then classified using the SVM model with an error of 21 percent.

5 Conclusion

In this study we attempted two neural computing methods to classify MNIST handwritten images. The results of both SVM and MLP were very competitive in terms of accuracy. During the training, MLP performed better than SVM, however, SVM outperformed MLP on the test set. Although SVM is considered a piece of art algorithm in classification tasks, it works by multiplying the inner product which was proven to be more computationally efficient. Nonetheless, it was evident in our experiment that SVM took more time during training than MLP, 25.19 and 8.15 seconds respectively (using the best models). But, on the test set SVM was faster than MLP, 1.12 and 2.42 seconds respectively.

Comparing SVM and MLP in terms of setup, SVM was very simple and fast to setup and work with. In MNIST, it can work even without a kernel transformation, as in our case using linear kernel and produce relatively good results. As for MLP, it was frustrating to work with as it needs more domain knowledge and parameter tuning. Setting up these parameters in a way to find the best optimised set of values for the network is very complex.

Reflecting our results on the domain, SVM and MLP were able to predict the labels correctly with an accuracy above 90 percent. However, SVM showed worst performance in predicting number 9 (misclassified 12 times) followed by number 5 (misclassified 10 times). As for MLP, the most misclassified numbers were 8, 5, and 9 where they were misclassified 20, 15 and 13 times respectively. The reason behind MLPs poor performance in comparison with SVM is that, MLP is a non-linear function thus it is better for non-linear models. It was shown earlier in this report, in the Dataset section, that the labels were linearly separable when projected on two dimensional space, this was shown in Figure 3 bottom. Also, MLP could have more than one local minimum due to the non-convex loss function associated with the hidden layers.

We expected the autoencoder to increase the accuracy, although the error almost doubled. The features learnt by the autoencoder as shown in Figure8 seem similar for each digit if observed by the naked eye. This could suggest co-adaptation,that all neurons in hidden layer have picked the same feature instead of a different one [10]. Fine tuning or using more training samples might have increased the accuracy as many literatures by Yann Le Cunn have reported an accuracy of more than 95 percent using the whole MNIST training set [13].

A further work to improve the accuracy of these model is to use Restricted Boltzmann Machine (RBM) instead of the autoencoder. RBM is a generative model, as oppose to autoencoder which favours some data vectors over other. Thus, RBM, after training the model, can generate new samples from the learnt probability distribution. RBM is also more flexible and feature rich. The idea behind RBM is very similar to the autoencoder. We train the model by passing the training sample through input units, propagate the data forward towards the hidden unit. The hidden unit becomes the feature vector instead of the raw data. We then use this vector as input to train the classifiers, SVM and MLP [18].

References

[1] L. Auria and R. A. Moro. Support vector machines (svm) as a technique for solvency analysis. *DIW Berlin Discussion*, 2008.

[2] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152. ACM, 1992.

[3] R. Caruana, S. Lawrence, and L. Giles. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *NIPS*, pp. 402–408, 2000.

[4] R. Collobert and S. Bengio. Links between perceptrons, mlps and svms. In *Proceedings of the twenty-first international conference on Machine learning*, p. 23. ACM, 2004.

[5] D. Decoste and B. Schölkopf. Training invariant support vector machines. *Machine learning*, 46(1):161–190, 2002.

[6] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874, 2008.

[7] T. Glasmachers and C. Igel. Maximum-gain working set selection for svms. *Journal of Machine Learning Research*, 7(Jul):1437–1466, 2006.

[8] H. P. Graf, E. Cosatto, L. Bottou, I. Durdanovic, and V. Vapnik. Parallel support vector machines: The cascade svm. In *NIPS*, vol. 17, 2004.

[9] M. T. Hagan, H. B. Demuth, and M. Beale. Neural network design, 1996. *Boston, PWS Pub*.

[10] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[11] C.-W. Hsu, C.-C. Chang, C.-J. Lin, et al. A practical guide to support vector classification. 2003.

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[13] Y. LeCun, C. Cortes, and C. J. Burges. The mnist dataset of handwritten digits. *URL http://yann. lecun. com/exdb/mnist*, 1998.

[14] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[15] S. Maji and J. Malik. Fast and accurate digit classification. *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-159*, 2009.

[16] C. D. Manning, P. Raghavan, H. Schütze, et al. *Introduction to information retrieval*, vol. 1. Cambridge university press Cambridge, 2008.

[17] A. Ng. Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19, 2011.

[18] R. Salakhutdinov and G. E. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In *AISTATS*, vol. 11, 2007.

[19] R. Schutt and C. O'Neil. *Doing data science: Straight talk from the frontline*. " O'Reilly Media, Inc.", 2013.

[20] S. Seung. Multilayer perceptrons and backpropagation learning. *Lecture Notes*, 2002.