Transfer Learning on MINST dataset

Millicent Omondi[1], *JosueNguinabe*[2],
Reham Jamal[3], *Atou Koffi ko ugbanhoun*[4]

Abstract

We applied transfer learning on the MNIST dataset that contains digits from 0 to 9 after dividing it into two, even and odd numbers. In comparison to the softmax regression model, transfer learning outperformed softmax regression on both datasets. On the odd data set, transfer learning had an accuracy of 0.95 while softmax had an accuracy of 0.9804 on the training data set, whereas, on the even data set, transfer learning had an accuracy of 0.9707 while softmax regression had an accuracy of 0.9641 on the training data set. From this comparison, we observed that transfer learning allows the model to benefit from the knowledge learned on related task leading to improved accuracy and faster convergence.

1 Introduction

Transfer learning is ML model technique where a model on one task is adapted for use on a second related task. Transfer methods are dependent on the machine learning algorithms being used to learn the tasks and are usually considered as extensions of those algorithms. Some of the work in transfer learning are based in the context of inductive learning, and involves classification and inference algorithms such as neural networks, Bayesian networks etc.

1.1 Background information

As humans, we have the ability to transfer knowledge between tasks. That is, we are able to recognize and apply our previous knowledge on new tasks and the more related the new tasks are, the easily we master it. On the other hand, common ML algorithms, address isolated tasks. Transfer learning attempts to improve on traditional learning by transferring the knowledge learned in one task to other related tasks and by doing so improving its learning.

The goal of transfer learning is to improve the learning performance of the model. We have three ways on measuring how transfer learning might improve the performance:

- The initial performance achievable in the target task using only the transferred knowledge, before any further learning is done, compared to the initial performance.
- The amount of time it takes to fully learn the target task given the transferred knowledge compared to the amount of time to learn it from scratch.
- The final performance level achievable in the target task compared to the final level without transfer.

1.1.1 Problem Statement

We aim to apply transfer learning on the MNIST data set to compare the performance of softmax regression and neural network models in classifying odd and even numbers from the given dataset. The MNIST data set contains handwritten numbers form 0 to 9. We will explore the effectiveness of transfer learning by utilizing the pre-trained neural network that we will build for the opposite parameter task. That is, we will use the odd neural network weights on the even data set and vice versa. By comparing these, three methods, our goal is to identify the most effective method for this classification task.

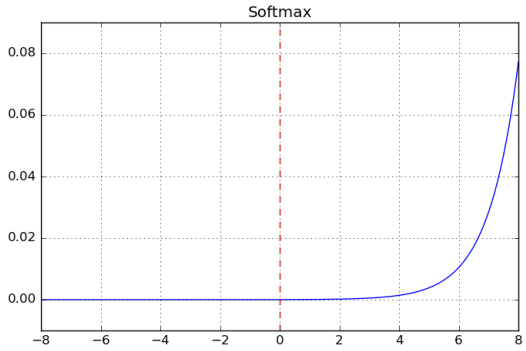
2 Methodology

Data Preparation: We started by preparing our data set to be in good shape before proceeding with our tasks. The MNIST dataset consists of $70,000$ images of handwritten numbers ranging from 0 to 9, with a resolution of $28 * 28$ pixels. These images were accompanied by corresponding labels indicating the presented digit number. So, here we started by splitting the data set into two parts, one with odd numbers, $0, 1, 3, 5, 7, 9$ and the other one with even numbers $0, 2, 4, 6, 8$. We then proceeded to divide each of the dataset into train (80%)and test (20%). This was followed by normalizing the data set, flattening the 2D pixel arrays into 1D arrays and finally we applied onehot encoding on the labels to have a binary representation for easy computation.

- **Task 1: Softmax regression model** In the first task, we built a softmax regression model to serve as our base model on each of the datasets. It is a simple linear model with a softmax activation function. We trained the model using the PyTorch library in Python. We optimized the parameters using the Adam optimizer and tuning hyperparameters such as the learning rate and number of epochs.
- **Softmax**

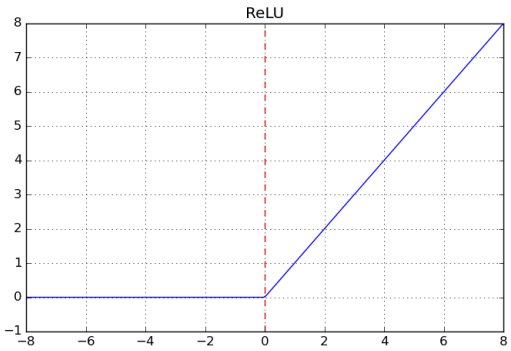
$$g(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

  - Softmax converts raw scores into probabilities, commonly used in multi-class classification tasks.



- **Task 2: Neural Networks** Our second task was on training two neural networks, one even numbers and the other one on odd numbers. Each of the neural networks had one hidden layer with 300 neurons, a ReLU activation function on the hidden layer and a sigmoid activation function on the output layer. ReLU sets negative values to zero and leaves positive values unchanged.
- **ReLU**

$$g(z) = \begin{cases} 0, & \text{if } z < 0 \\ z, & \text{otherwise} \end{cases}$$



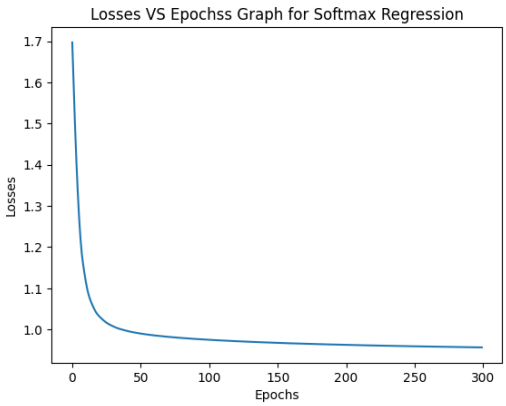
- **Task 3: Transfer learning on Neural Networks** Here, we applied transfer learning on the two neural network models that we built. The first instance was to use the even neural network weights to train the odd set and the second instance was using odd neural network weights and applying it on the even dataset.

3 Experiments and results

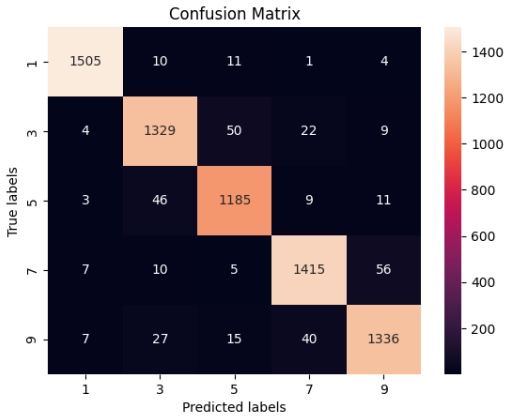
- **Task 1: Softmax regresssion**

  Model Selection: We used a ***Soft-max Regression model*** to perform our multi-class classification task.

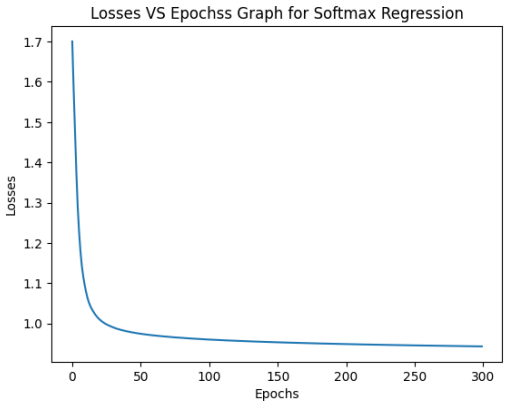
  Hyperparameters: Our hyperparameters in this task included the number of iteration / epochs which were 300 and the learning rate which was 0.01. For the even data set, the model achieved an accuracy of 0.9641 on the training dataset with a training time of 16.69 seconds. For odd numbers, we obtained an accuracy of 0.9804 with a training time of 26.07 seconds.



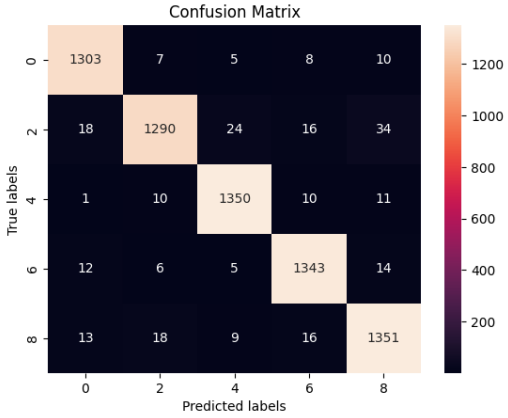
(a) Training of the " odd"



(b) The corresponding Confusion Matrix



(c) Training on the "Even"



(d) The corresponding Confusion Matrix

Figure 1: Metric performance evaluation of Softmax Regression

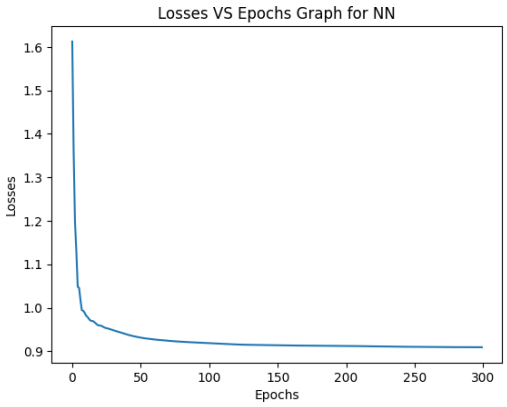
From Figure 1, we observe that as the number of epochs were increasing, the losses were also reducing. Looking at the respective confusion matrices, we observe that we have a large proportion on true positives(the ones in diagonal) as compared to the wrongly predicted value (dark shade).

- **Task 2: Neural Network**

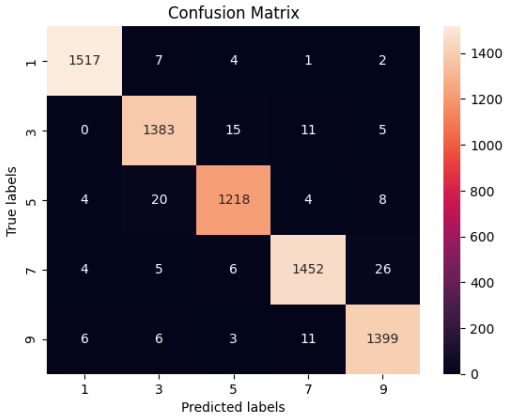
  Model Selection: We used a ***Neural Network*** to perform our multi-class classification task. Our hidden layer had the ReLU activation function to capture the complex patterns in the dataset.

  Hyperparameters: Our hyperparameters in this task included the number of iteration / epochs which were 300 and the learning rate which was 0.01. We used the adam optimizer in our hyparparameter tuning and the SGD optimizer had lower results as compared to adam. For the even data set, the model achieved an accuracy of 0.9807 on the training dataset with a training time of 202.24 seconds. For odd numbers, we obtained an accuracy of 0.9792 with a training time of 237.65 seconds.

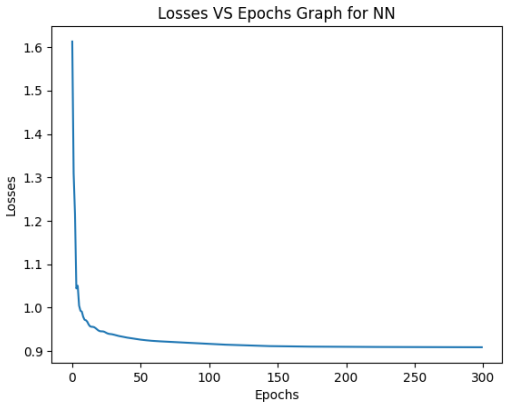
  From this, we observe that our model performed slightly higher on the odd dataset as compared to the even numbers.



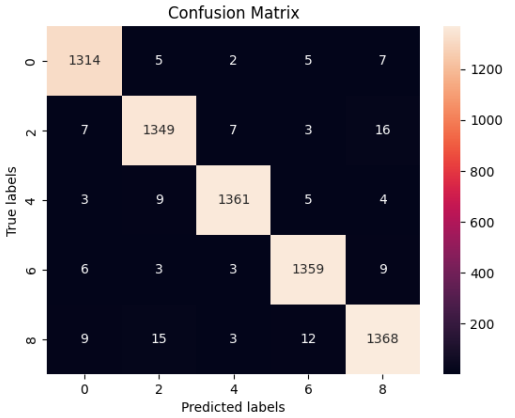
(a) Training of NN with 1-hidden-500 units on "Odd"



(b) The corresponding Confusion Matrix



(c) Training of NN with 1-hidden-500 units on "Even"



(d) The corresponding Confusion Matrix

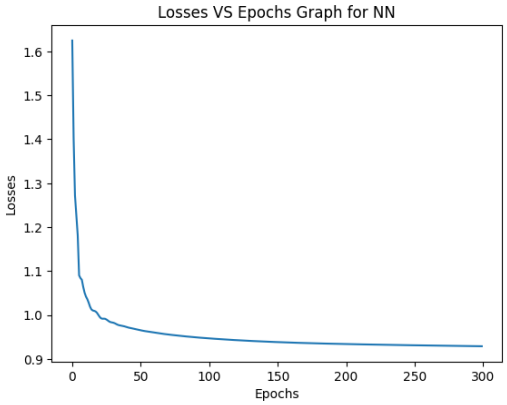
Figure 2: Metric performance evaluation of Neural Network,

From Figure 2, we observe that as the number of epochs were increasing, the losses were also reducing as seen on the smooth loss curves. Looking at the respective confusion matrices, we observe that we have a large proportion on true positives(the ones in diagonal) as compared to the wrongly predicted value (dark shade).

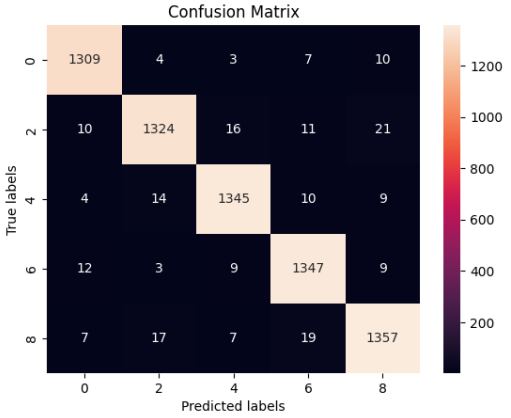
- **Task 3: Transfer Learning**

  Model Selection: Transfer learning was applied by using the parameters of the trained neural network as a transfer function for the opposite classification task.

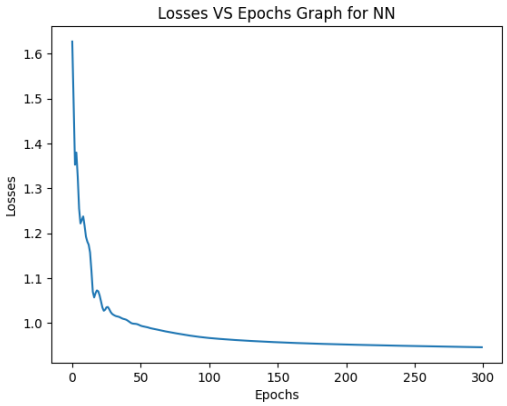
  Hyperparameters: Our hyperparameters in this task included the number of iteration / epochs which were 300 and the learning rate which was 0.01. We used the adam optimizer as well. Using the weights obtained on the odd neural network and applying it on the even dataset, the model obtained an accuracy of 0.9707 on the training dataset with a training time of 113 seconds. On the other hand, using the weights obtained on the even neural network and applying it on the odd dataset, the model obtained an accuracy of 0.95 on the training dataset with a training time of 97.92 seconds.



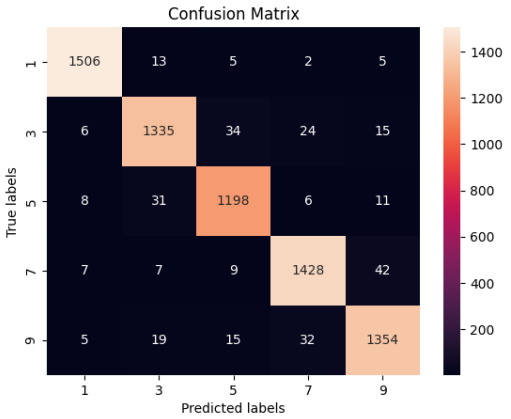
(a) 'Odd' features transferred to 'Even' training



(b) The corresponding Confusion Matrix



(c) 'Even' features transferred to 'Odd' training.



(d) The corresponding Confusion Matrix.

Figure 3: Metric performance evaluation of Neural Network

From Figure 3, we can see that the plot for the loss function of even is smoother as compared to even. This might illustrate why the transfer learning performance on predicting the odd values was lower as compared to the softmax regression model. The confusion matrix shows that we have a large proportion on true positives(the ones in diagonal) as compared to the wrongly predicted value (dark shade).

From this, we observe that, the transfer learning models took less time as compared to the neural networks without transfer. Looking at the accuracy, we see that the accuracy on the even numbers increased slightly while on the odd transfer learning slightly decreased. This implies that for the odd numbers, a negative transfer learning had occurred since the transfer method decreased the performance. One of the major challenges in developing transfer methods is to produce positive transfer between appropriately related tasks while avoiding negative transfer between tasks that are less related.

4 Conclusion

The comparison highlights the superior performance of the Neural Network models, particularly in terms of accuracy. Transfer learning further enhances the effectiveness of Neural Networks by leveraging pre-trained parameters, leading to improved performance and faster convergence. Despite longer training times, Neural Networks offer greater flexibility and effectiveness in capturing complex patterns in the data. However, Softmax Regression provides simplicity and interpretability, making it suitable for simpler classification tasks.

Supplementary Material

All materials related to this work can be found here.

References

Torrey, L., Shavlik, J. (2010). Transfer learning. In Handbook of research on machine learning applications and trends: algorithms, methods, and techniques (pp. 242-264). IGI global.

Weiss, K., Khoshgoftaar, T. M., Wang, D. (2016). A survey of transfer learning. Journal of Big data, 3, 1-40.