Fraud Detection Using Deep Neural Networks & Optical character recognition

Golmehr Khosrokhavar

Student,ECE Department;
University of Tehran, College of Engineering
Course: Neural networks & Deep learning

*Corresponding Author: khosrokhavar.g@gmail.com

Abstract

This project integrates a Denoising Autoencoder for imbalanced data classification and a modified Deep Convolutional Neural Network for enhanced Persian/Arabic handwritten digit recognition. Through simulations, the integrated model demonstrates superior performance, offering a versatile and robust solution to challenges in machine learning with heightened accuracy and adaptability.

Part A - Fraud Detection Using Deep Neural Networks

First What are the biggest challenges in developing fraud detection models? This article to solve What methods did these challenges use?

According to the mentioned article, many researches have been done in the field of fraud detection in credit cards, been But the classical methods were impractical and unenforceable because in these issues, we are dealing with Big Data.
In this issue, we are dealing with an unbalanced dataset. This means that the number of members of the classes are significantly different from each other. For example, only 1% of the total data is a member of the Fraud transaction class, and 99% of the data is a member of the normal transaction class. In other words, with a majority class we are dealing with a minority class.
In this example, the minority class is more important in fraud detection. Solving this problem is one of the main challenges of this problem because, for example, when the minority class is less than 1% of the entire data set, the overall accuracy reaches more than 99% even if all the minority classes are wrong to be classified.
To solve this problem, oversampling is proposed. The method used is SMOTE. (Synthetic Minority Over-sampling Technique) (SMOTE) is a method to solve the problem of Imbalanced Datasets.

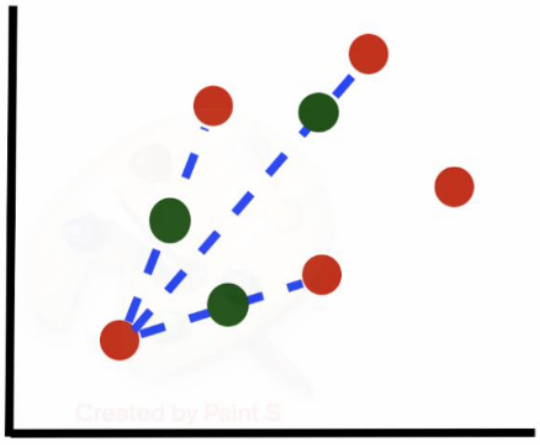

Figure 1: Plot-1

In this method, the following steps take place according to figure 1:

- The difference between the sample and its nearest neighbor is considered.
- This difference is multiplied by a random number between 0 and 1.
- This difference is added to the instance to create a new synthetic instance in the feature space.
- These steps are repeated with the next sample.

Second Briefly explain the network architecture presented in the article.

The figure below shows that the neural network architecture used is one AutoEncoder and a built-in Classifier. The architecture of AutoEncoder

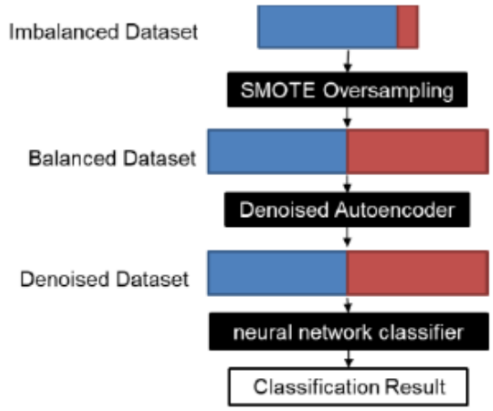

Figure 2: Algorithm

and Classifier is as follows, which consists of Fully Connected Neural Networks.

Dataset with noise (29)
Fully-Connected-Layer (22)
Fully-Connected-Layer (15)
Fully-Connected-Layer (10)
Fully-Connected-Layer (15)
Fully-Connected-Layer (22)
Fully-Connected-Layer (29)
Square Loss Function

Figure 3: Autoencoder Architecture

Denoised Dataset (29)
Fully-Connected-Layer (22)
Fully-Connected-Layer (15)
Fully-Connected-Layer (10)
Fully-Connected-Layer (5)
Fully-Connected-Layer (2)
SoftMax Cross Entropy Loss Function

Figure 4: Classifier Architecture

Third Name the types of resampling methods available to balance the data

In general, there are two methods to balance data.

Undersampling This method balances the dataset by reducing the size of the majority class. This time method It is used when the amount of data is sufficient. By keeping all samples in the minority class and randomly selecting an equal number of samples in the majority class, a new balanced dataset can be recovered for further modeling.

Over-sampling In contrast, oversampling is used when the data amount is insufficient. This way It tries to balance the data set by increasing the size of the minority class. Instead of getting rid of many instances, new minority classes are generated using e.g. SMOTE or repetition, bootstrapping

Fourth Do you think that in problems where the distribution of labels is uneven, the use of accuracy criterion Enough? If not, which criteria can show the model's performance in a complementary way?

no, As can be seen in this example. In this case, the use of the Accuracy criterion alone cannot show the correct performance of the model because in cases where less than 1% of the data have the Fraud class label, the model may misclassify all these data, and despite this, The problem is to reach accuracy performance above 99%
In addition, in the field of Accuracy, the Recall criterion can be used, which is the number of TruePositives showing us the fraction of Possible positives.

Implementation of the model - Part A

Fifth Checked the performance of the model with different thresholds for oversampling.

In this section, I have considered 5 different values for Ratio in Oversampling than Number Ratio; I changed minority class to majority class by 0.1 to 0.5 with steps of 0.1.
A scatter diagram of normal data and fraud-related data for each ratio according to the figures below is :

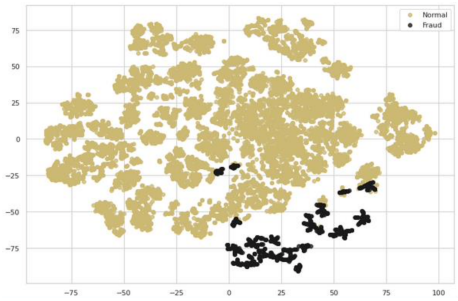

Figure 5: Scatter plot for ratio = 0.1

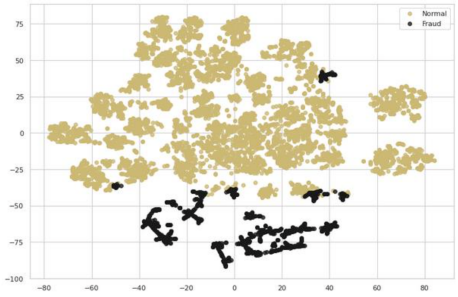

Figure 6: Scatter plot for Ratio = 0.2

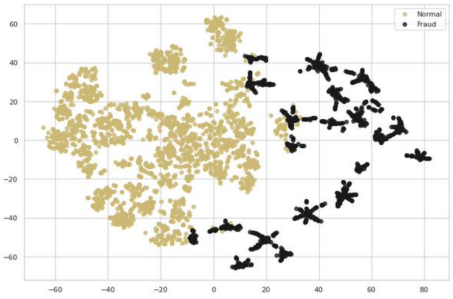

Figure 7: Scatter plot for Ratio = 0.5

After going through the Denoising AutoEncoder and removing the noise, the scatter plot of the data changes, as shown in Fig 8 and Fig 9:

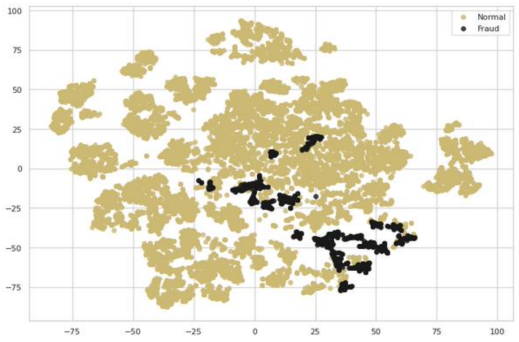

Figure 8: After DAE for Ratio = 0.1

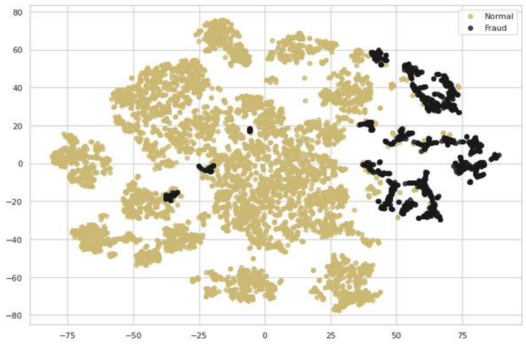

Figure 9: After DAE for Ratio = 0.2

As can be seen, the effect of Autoencoder and noise removal is clear in the above data. In practice, the data becomes denser and more linear to be easily separated with the help of a linear classifier.

Accuracy & Recall Graph in Part A

In the following, we will draw the ACCURACY & RECALL graph for the above ratio values to see the effect of changing the ratio of minority class to majority class on the performance of the model:

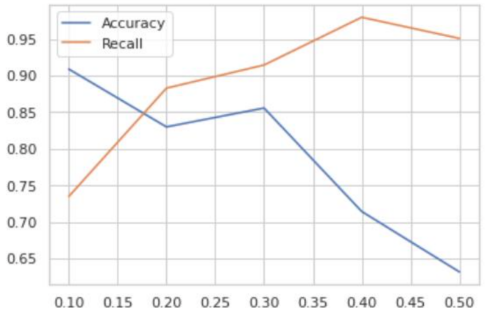

Figure 10: Accuracy & Recall Graph

As we expected, increasing the ratio of members of the minority class to the majority class leads to a decrease in improving accuracy and recall, which means increasing the number of true positives.

Sixth We train the model with unbalanced data.

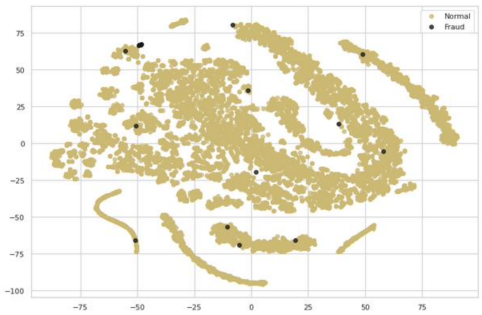

Figure 11: Unbalanced Data

As can be seen in the above diagram, if we do not oversample the number of members of the minority class, their dispersion in the model will be very high. Their number will be minimal compared to your whole finger, which, as we expect, will lead to very high accuracy of the model and very low recall.

Part B - Recognition Character Optical

Introduction

In this question, to simulate the article A recognition model for handwritten Persian/Arabic numbers based on optimized deep convolutional neural network We use the HODA dataset.

First Explain the difference between CNN and DCNN networks.

They are not so different from each other, but deep means that fewer convolution layers are used. Today's CNNs are between 30 and 100 layers deep.

Second Explain the three optimization methods Adam, Adadelta, and Momentum.

Adam Adam helps to control the exponential decline of moving averages by simulating the exponential moving averages values of RMSProp gradient and its square as Adadelta.

Adadelta Adadelta prevents the accumulation of squared gradients by setting the window size.

Momentum In Momentum, the value of the updated step length in the previous step affects the current step length update has it. This action helps to reduce the parameter that is repeated in each step.

Third Implement the DCNN architecture used in the paper. Preprocessing and State the normalizations used. Explain the number of layers used, the type of layer, and the reason for using them. In order to avoid overfitting, A technique has been used.

First, it is said in the article that the photos should be resized, and all of them should be 40x40 reach. Then, in order to change all indices to 0 or 1, one hot coding must be done. Finally, we have been asked to change the background of all the photos to white and all the objects in the photos to black.

Network Details - Part B

In this network, 4 convolution layers are used, each one of two one-dimensional convolution layers, Batch normalization, pooling, and dropout layer. At the end of the network, dropout, flatten, and dense have been used for classification, which finally brings the output to 10 classes, which are the classification of characters numbers 0 to 9.

Avoid Overfitting

To avoid overfitting the model on the data, dropout has been used in each layer to drop the number of neurons in each network layer with probability.

Convolution Layers

The convolutional layer is used to extract features for each photo. The reason for using the pooling layer is to reduce the dimensions of the feature map. The reason for using the dropout layer has been explained earlier. We first use the flattened layer for classification, which converts the obtained two-dimensional features into a vector. Then, using fully connected and dropout layers, as mentioned earlier done, from the output of 1024 to 10 for classification.

Fourth accuracy and loss charts as well as confusion matrix and State and compare the recall, precision, and f1score values for each of the three optimization methods.

Result For Adam Model

ADAM Optimizer: batch size = 128

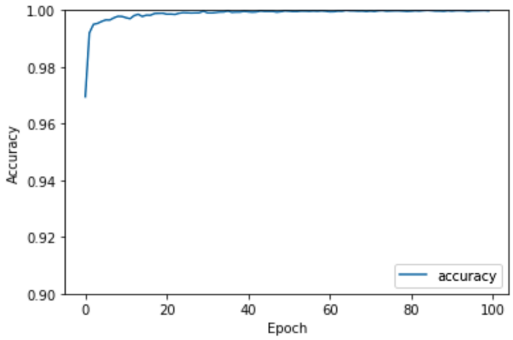

Figure 12: adam optimizer model accuracy

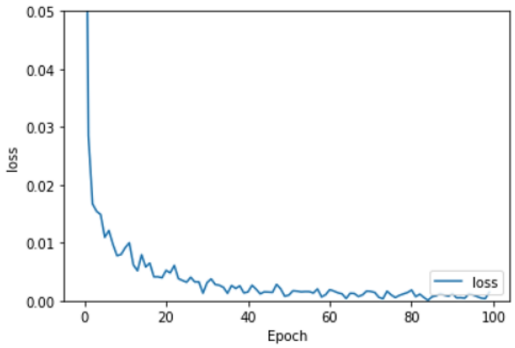

Figure 13: Model error with adam optimizer

Classification Report:	precision	recall	f1-score	support
0.0	1.00	0.99	1.00	2000
1.0	1.00	1.00	1.00	2000
2.0	0.99	0.99	0.99	2000
3.0	1.00	0.99	0.99	2000
4.0	0.99	1.00	1.00	2000
5.0	0.99	1.00	1.00	2000
6.0	1.00	1.00	1.00	2000
7.0	1.00	1.00	1.00	2000
8.0	1.00	1.00	1.00	2000
9.0	1.00	1.00	1.00	2000
accuracy			1.00	20000
macro avg	1.00	1.00	1.00	20000
weighted avg	1.00	1.00	1.00	20000

Figure 14: precision, f1-score and recall values

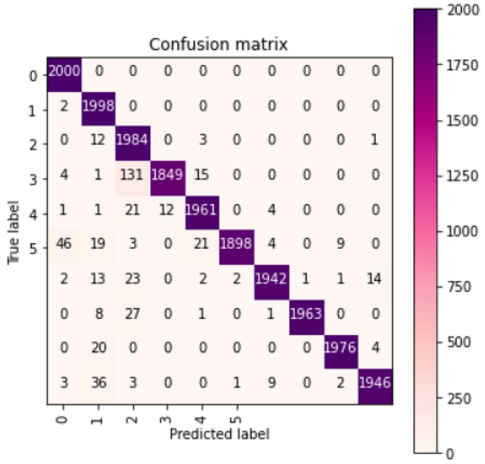

Figure 15: Confusion Matrix

Result For Adadelta Model

We do the same for other models to have a comparison between them

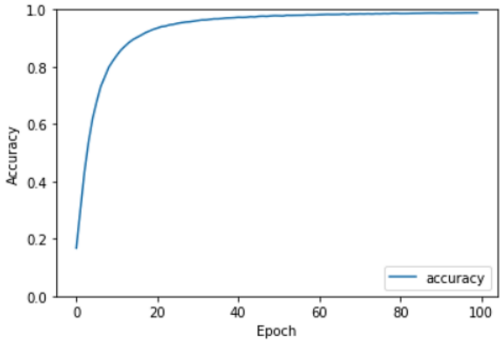

Figure 16: Adadelta optimizer model accuracy

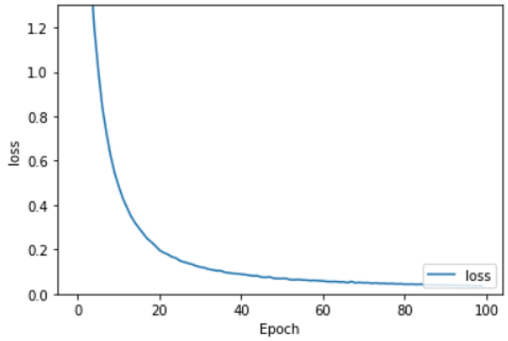

Figure 17: Model error with Adadelta optimizer

Classification Report:

	precision	recall	f1-score	support
0.0	0.96	1.00	0.98	2000
1.0	0.95	1.00	0.97	2000
2.0	0.92	0.99	0.96	2000
3.0	0.99	0.94	0.96	2000
4.0	0.99	0.97	0.98	2000
5.0	1.00	0.92	0.96	2000
6.0	0.98	0.98	0.98	2000
7.0	1.00	0.99	0.99	2000
8.0	0.97	0.99	0.98	2000
9.0	0.99	0.96	0.98	2000
accuracy			0.97	20000
macro avg	0.98	0.97	0.97	20000
weighted avg	0.98	0.97	0.97	20000

Figure 18: precision, f1-score and recall values

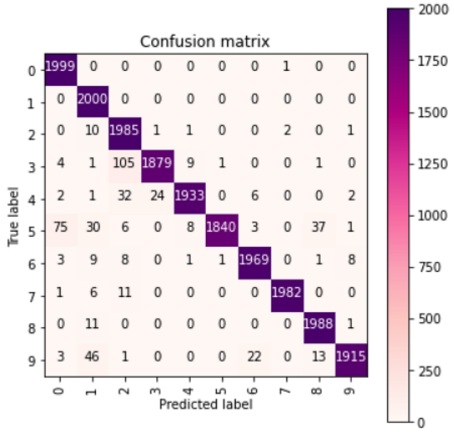

Figure 19: Confusion Matrix

Result For Momentum Model

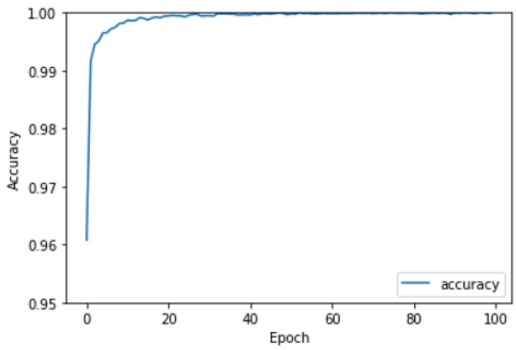

Figure 20: Momentum optimizer model accuracy

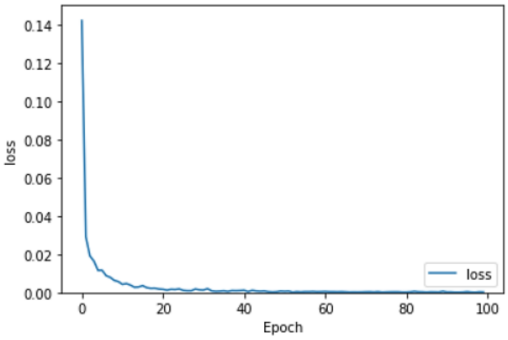

Figure 21: Model error with Momentum optimizer

Classification Report:

	precision	recall	f1-score	support
0.0	1.00	1.00	1.00	2000
1.0	0.99	1.00	1.00	2000
2.0	0.99	1.00	0.99	2000
3.0	0.99	0.99	0.99	2000
4.0	1.00	0.99	1.00	2000
5.0	1.00	1.00	1.00	2000
6.0	1.00	1.00	1.00	2000
7.0	1.00	1.00	1.00	2000
8.0	1.00	1.00	1.00	2000
9.0	1.00	1.00	1.00	2000
accuracy			1.00	20000
macro avg	1.00	1.00	1.00	20000
weighted avg	1.00	1.00	1.00	20000

Figure 22: precision, f1-score and recall values

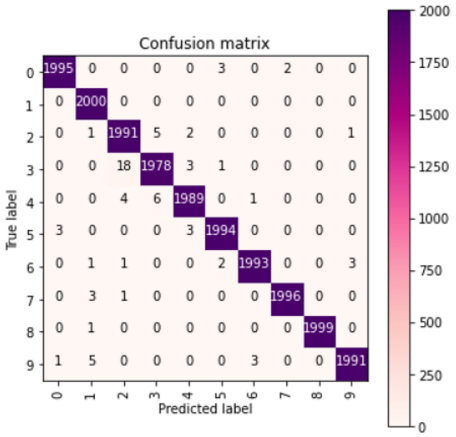

Figure 23: Confusion Matrix

1 Conclusion

By comparing the results obtained from three different optimizers, we find that Adam and Momentum optimizers have the best results, But Adam is getting closer to ultimate accuracy faster. The difference between these two with the Adadelta optimizer is noticeable but very little. So, when we need a small number of epochs to model, we can use the Adam optimizer if reasonable accuracy is reached.

Fifth Describe the architecture and parameters of the best network.

As mentioned in the previous section, the network with Adam and momentum optimizers performs better ; But by comparing these two, we conclude that the momentum optimizer is better. This network is applied with the mini-batch method and with 128=batch size.
Also, its architecture consists of 4 convolutional networks that are used in each of the convolution, batch dropout, pooling, normalization and activation layers. At the end of it, fully connected, flatten and dropout layers are used.

References

Ali, S., Sahiba, S., Azeem, M., Shaukat, Z., Mahmood, T., Sakhawat, Z., Aslam, M. S. (Year). A recognition model for handwritten Persian/Arabic numbers based on optimized deep convolutional neural network. Journal Name, Volume(Issue), Page range. DOI or URL (if available).

Zou, J. (Year). Credit Card Fraud Detection Using Autoencoder Neural Network. Department of Electrical Computer Engineering, University of Western Ontario. Email: jzou44@uwo.ca. Student ID: 250833154.