Reproducibility Project: Multi-label Classification Performance on Psychotic Disorder Diseases

Manupriya Arora and Vraj Patel
{manupri2, vpate33}@illinois.edu

Group ID: 83
Paper ID: 240, Difficulty: Easy
Presentation link: https://youtu.be/7GJ_lF9Jg4g
Code link: https://github.com/vp-98/multilabel-classification-pdd

1 Introduction

Electronic Health Records (EHRs) contain symptoms of many diseases including the symptoms of Psychotic Disorders. In the original paper, the use of deep neural network and machine learning techniques such as multilayer perceptron (MLP), random forest (RF), decision tree (DT) and support vector machine (SVM) have been investigated to identify the patterns of psychotic diseases in a patient's data (Elujide et al., 2021). The study presents the multi-label classification problem with bipolar disorder, schizophrenia, attention-deficit/hyperactivity disorder (ADHD), vascular dementia and insomnia as five labels. The study also addresses concerns regarding multi-label classification with class imbalance present within a dataset. Class imbalance occurs when some classes may have more instances than other classes in a given dataset.

Mostly non-severe symptoms precede mental illnesses, but these symptoms are hard to diagnose. Emerging technologies like deep and machine learning can provide a promising and reliable approach to identifying these symptoms, and hence the early diagnosis of psychotic disorders. Since there is no cure for mental disorders, the early diagnosis can help in slowing down the progression of these disorders.

2 Scope of Reproducibility

Two interesting claims were made in the paper. Both the claims that will be addressed concern the model that the original paper created called the Deep Neural Network Mulilayer Perceptron (DNN-MLP). The first of which is the claim that the DNN-MLP will be able to provide noticeably higher accuracies and will outperform several of the baseline machine learning approaches such as multilayer perceptron (MLP), random forest (RF), decision tree (DT) and support vector machine (SVM) classifiers when trained with a dataset that has class imbalance present. Alternatively, the other claim that will be addressed is the claim that the random forest (RF) and multilayer perceptron (MLP) will be able to reliably predict psychotic disorders and provide higher accuracies compared to the DNN-MLP when working with a dataset without class imbalance present.

The motivation behind these claims was that class imbalance present in data can bring up many concerns regarding how accurate the prediction step can be for both the majority and the minority classes. Knowing how to properly handle imbalanced datasets and knowing which approaches to use can be a significant factor.

2.1 Addressed claims from the original paper

Clearly itemize the claims you are testing:

- The proposed deep neural network multilayer perceptron (DNN-MLP) will provide better accuracies and will outperform machine learning techniques such as multilayer perceptron (MLP), random forest (RF), decision tree (DT) and support vector machine (SVM) when working with a dataset where the classes are not balanced.
- The random forest (RF) and multilayer perceptron (MLP) will outperform the proposed deep neural network multilayer perceptron (DNN-MLP) when working with a dataset where the classes are balanced with random forest (RF) being the most accurate.

3 Methodology

The original study has employed deep learning and machine learning techniques for the multi-label classification of the psychotic disorders. The paper does not provide any references or access to

source code for their implementation. However, the paper provides great amount of detail for the the approach taken towards the implementation and training steps of the models, so we would be re-implementing the models that are described in the paper ourselves.

3.1 Model descriptions

The original paper created and utilized machine learning and deep learning models. This included a deep neural network called the DNN-MLP and several machine learning models such as support vector machine, random forest, decision tree, and multilayer perceptron classifiers. For the machine learning approaches, the data was split into 80-20 for training and test while the DNN-MLP was trained and tested with a training and test data split of 70-30 for 40 epochs. The training data running for 40 epochs had early stopping monitor on the validation loss. The DNN-MLP model was trained using Keras APIs. The architecture of the DNN-MLP consisted of three fully connected layer deep network with a ReLU activation function for the connected layers and sigmoid activation function for the output layer. The optimizer used for this model was Adam with a learning rate of 0.01, the loss function used for training was binary cross-entropy, and accuracy was the evaluation metrics. Drop-out was used as regularization method in layers 1 and 2 of the model to prevent overfitting. The fine tuning of the hyper-parameters was done using grid-based search approach.

3.2 Data Description

The dataset is from Yaba Psychiatry Hospital, Yaba, Lagos State, Nigeria by Adejumo et al. (2017). It is available as supplementary data as part of another paper *"Quantitative exploration of factors influencing psychotic disorder ailments in Nigeria."*2017. The dataset consists of 500 records of patients, and there are 16 columns in the dataset of which 11 are independent and 5 are dependent variables.

The original dataset had P and N for each dependent variable in every row, it was modified later to replace P with 1 and N with 0. All the dependent variables were later combined to create a combined target converting a multi-label classification to conventional multi-classification problem where there is only one target but there are more than two classes. The modified dataset contained very few instances (less than 6) for some of the classes, this implied that we had class imbalance. For better reliability and prediction accuracy, it is better to have balance in classes. To resolve this issue, we generated synthetic samples using Synthetic Minority Oversampling Technique (SMOTE). One-hot-vector encoding can be applied to the categorical feature vectors to transform them into binary as SMOTE is applicable to only numeric features vectors.

For good performance, it was recommended that each class should have at least 6 occurrences, so we removed all the classes with less than 6 occurrences and then applied SMOTE to the updated dataset. After applying SMOTE, synthetic samples got generated and all the classes had equal number of samples. The majority class had 101 samples, so after applying SMOTE the number of samples were 101 for each class. At this point, we got the dataset without class imbalance, and this dataset had 1212 records.

For dataset with class imbalance, we just replaced P with 1 and N with 0 for all the dependent variables, and combined all the dependent variables to create a combined target converting a multi-label classification to conventional multi-classification problem where there is only one target but there are more than two classes.

3.3 Hyperparameters

The hyper-parameters used for the deep learning model in this paper are similar to hyper-parameters used for the deep learning model in the original paper. The fine-tuning of the hyper-parameters was done using the grid-based search approach in the original paper. However, we did not use grid-based search approach for the hyper-parameters in our model, we simply adopted the values mentioned in the original paper. As mentioned in original paper, we used the same deep learning architecture with 3-layer deep fully connected network and output dimensions 15-20-20-40-5. There are a total of 2190 trainable parameters for the DNN-MLP model. The training and test data split is 70% and 30%, and the training data runs for 40 epochs. The activation function used are Rectifier Linear Unit (ReLU) and sigmoid, and the drop-out is chosen for layers L1 and L2 only. The loss function used is binary cross-entropy, and evaluation metrics used is accuracy. The only differences between the hyper-parameters proposed in the original paper and the hyper-parameters used in our paper are that we have used the default learning rate of 0.001 for op-

timizer Adam whereas the learning rate is 0.01 in original paper, we were getting poor accuracy with learning rate 0.01, and that we haven't used early stopping monitor on validation loss.

There is no information on the hyper-parameters used in the machine learning techniques in the original paper. So, we have used the default parameters for Random Forest (RF) and Decision Tree (DT). For support vector machine (SVM), we have used linear kernel which is the simplest classifier and gives higher performance accuracy than default classifier, and the cache size of 1500 as it decreases the processing time by a few seconds. For machine learning MLP classifier, we have used Rectifier Linear Unit (ReLU) activation function as specified in the original paper. For all machine learning models, we have set the random state to 42.

3.4 Implementation

The original paper did not provide any reference to the any existing source code. With that being said, we are implementing our own code. Our github repo link is (https://github.com/vp-98/multilabel-classification-pdd). The code that we have implemented is described in the README.md file of the github repo. Description of the code that we have implemented so far is as follows:

- Create One-hot encoding of the categorical feature vectors so that SMOTE can use it to generate synthetic samples
- Apply SMOTE implementation to balance the classes
- Training ML models such as multilayer perceptron (MLP), random forest (RF), Decision tree (DT) and support vector machine (SVM) with training data, and calculating the accuracy w.r.t test data for both balanced and imbalanced datasets
- Training the deep neural network multilayer perceptron model with training data, and calculating the accuracy w.r.t test data for both balanced and imbalanced datasets

3.5 Computational Requirements

Like the original study, our implementation was done in Google Colaboratory Pro notebook with Nvidia GPU running CUDA version 11.2. Our two Intel(R) Xeon(R) CPUs ran at 2.20GHz and consisted of 2 cores each. Our RAM was limited to 13.6 GB. Before creating and running the models, it was expected that training and testing our models would be a feasible task in terms of resources available due to the small dataset provided. Throughout training and testing, execution times were recorded for each of the models. The machine learning models had an average CPU time of 1.72s and a wall time of 0.95s for training and testing, of which the MLP classifier with the balanced dataset had the longest CPU time of 8.35s and longest wall time of 4.3s. The replicated DNN-MLP had an average CPU time of 5.61s and a wall time of 4.59s. On average our memory usage was around 15% or roughly 2.13GB.

4 Results

The work we have done in our paper does support the claims we have made, and they also match with the conclusions of the original paper. We compared the accuracies obtained from our models with accuracies reported in the original paper and noticed that for some of the models accuracies match, for others we obtained better accuracies and for rest of the models accuracies were far off the accuracies reported in the original paper.

4.1 Result 1

For imbalanced dataset, the accuracies obtained from DNN-MLP model and all the machine learning models that we implemented are mentioned under column "Obt. Accuracy" in Table 1, and the accuracies reported in the original paper for the corresponding models are mentioned under column "Rept. Accuracy".

Models	Obt. Accuracy	Rept. Accuracy
MLP	38.00	58.44
SVM	43.00	46.91
RF	26.00	46.91
DT	27.00	46.91
DNN-MLP	72.00	75.17

Table 1: Comparison of deep learning and machine learning techniques accuracies(%) for imbalanced dataset

All of the accuracies listed in table 1 were lower than the reported values in the original paper. However, we were still able to achieve the same pattern mentioned in the paper. The DNN-MLP model was successfully able to achieve an accuracy of 72% with the imbalanced data and outperformed all the

machine learning models upholding the first claim outlined in 2.1.

4.2 Result 2

For the balanced dataset, the accuracies obtained from DNN-MLP model and all the machine learning models that we implemented are mentioned under column "Obt. Accuracy" in Table 2, and the accuracies reported in the original paper for the corresponding models are mentioned under column "Rept. Accuracy".

Models	Obt. Accuracy	Rept. Accuracy
MLP	58.84	58.53
SVM	64.60	49.82
RF	69.54	64.06
DT	66.67	49.82
DNN-MLP	53.02	55.87

Table 2: Comparison of deep learning and machine learning techniques accuracies(%) for balanced dataset

Using the balanced data, the machine learning models obtained a better accuracy of 11.89% more than the accuracy obtained by the DNN-MLP model on average. The multilayer perceptron (ML) classifier, random forest (RF) classifier, and DNN-MLP all produced accuracies similar to the ones outlined in the paper. The machine learning models on average differed 9.35 percentage points from the reported values. The accuracy provided by our MLP classifier was almost identical to the reported accuracy of 58.53%. With an accuracy of 53.02%, the replicated DNN-MLP is within 2.85% of the reported value at 55.87%. The DNN-MLP also had the worst accuracy compared to the rest of the machine learning models. Similar to the original paper, our the Random Forest (RF) and Multilayer Perceptron (MLP) outperformed the DNN-MLP in terms of accuracy. The best of which was the Random Forest (RF) with accuracy of 69.54% upholding the second claim outlined in section 2.1.

4.3 Additional Results

In addition to the replication of the original findings from the paper, we performed our own ablation study and we also created our own deep neural network model in attempts to see if we could improve the accuracies for both the balanced and imbalanced datasets.

Since the DNN-MLP model is relatively simple, we decided to remove certain hidden layers in the DNN-MLP alternatively to see how the performance was affected. To accomplish this, we created three additional variations of the DNN-MLP model each with a missing hidden layer. These newly created models were then trained and tested with the balanced dataset. The findings are outlined in table 3. From the ablation study, we were able to determine that each of the hidden fully connected hidden layers played a significant role when it comes to performance. Accuracies can be seen dropping nearly 10% each time a hidden layer was removed when compared to the fully intact DNN-MLP model. The results from the ablation study indicated that each hidden layer played a significant role when it comes to reliably predicting PDD for this model.

Removed Layer	Accuracy
1st	40.65
2nd	45.60
3rd	37.08

Table 3: The accuracies(%) of the DNN-MLP with certain hidden layers removed with the balanced dataset

In efforts to see if we could increase the accuracies for both the imbalanced and balanced data, we created our own model as an additional experiment. To accomplish this, we used the PyTorch Neural Network Modules. Our model consisted of an input layer and a output layer along with four fully connected hidden layers. Each of the hidden layers had a ReLU activation function while the final output layer had a sigmoid function. Dropouts were applied to the first two hidden layers. The hyperparameters used were similar to the ones used before for the DNN-MLP. The only difference was that we had our own custom Dataloader class uses a batch size of 32 and our model also had 19125 trainable parameters. The accuracies and ROC AUC values obtained can be seen in table 4.

Our model obtained an accuracy of 45.05% for the balanced data and 33.33% for the imbalanced data. With that being said our model was also able to achieve good ROC AUC values of 77.24% and 86.62% for the imbalanced and balanced datasets respectively.

5 Discussion

Though at times our accuracies varied from the original paper, we conclude that the claims pre-

Dataset	Accuracy	ROC AUC
Imbalanced	33.33	77.24
Balanced	45.05	86.62

Table 4: The accuracy(%) and ROC AUC(%) for both the datasets using our custom DNN model.

sented in the paper can be and were successfully reproduced. From our attempts of replication, we got results that defended both the claims outlined in 2.1. Given the original imbalanced dataset, we were able to come to the conclusion that the DNN-MLP model delivered the best performance when compared to the other machine learning models. Though our accuracies did not match perfectly, our results still successfully confirmed that the DNN-MLP was better suited for the imbalanced datasets rather than the machine learning techniques. Given the balanced dataset, we determined that the machine learning techniques were a better alternative to the DNN-MLP model. Matching closely, we successfully observed that of all the machine learning techniques, the Random Forest (RF) provided the best accuracies compared to the DNN-MLP model. In fact, all of our machine learning models were collectively able to outperform the DNN-MLP model.

The only issue we noted throughout the procedure was that the accuracy obtained at times was lower than what was reported in the paper. This could be due to our approach using a predefined SMOTE algorithm provided by the Imbalanced-Learn Library. The algorithm provided by this library may have created new synthetic data in a different manner which could have potentially affected the accuracies of the models. We also did not perform grid-based search approach to fine-tune our hyper-parameters. Maybe if we had used that approach we could have got better accuracies for models trained on imbalance dataset. Also, we did not use early stop monitor on validation loss on the replicated DNN-MLP, so we could potentially be overfitting. Implementing early stop validation could in turn help us get even better accuracies for our DNN-MLP models for both balanced and imbalanced datasets.

For our additional experiment, we tried to implement our own DNN-MLP model using PyTorch Neural Network Modules. The accuracies we obtained from our model were really low compared to the ones obtained from the model created using the Keras API. If we had more time, we would have tried to re-design and improve our model which could consecutively increase the accuracies of our model for both the balanced and imbalanced datasets.

5.1 What was Easy

Since the deep learning model was described in detail in the original paper, it was relatively simple to create a replicate model. The authors stated that they used the Keras functional API. This library contains extensive documentation and examples which made it easy to construct a model with a similar architecture. We were also able to access and use the proper data easily. The data processing steps were simple to follow too. Setting up a similar environment was quite easy since the paper mentioned how and where their models were ran.

5.2 What was Difficult

Since the paper did not provide existing code or starting code, we had to implement everything ourselves from scratch. Most of the difficulties associated with this replication came from creating the identical machine learning models and creating a matching SMOTE algorithm described in the paper. The types of classifiers used were the only information about the machine learning models that was provided in the original paper. Creating the machine learning models ourselves left a lot of room for assumptions for the hyperparameters used during the construction and training of these models. The SMOTE algorithm that was used in the study was explained in a rather complex and abstract way making it difficult to implement without prior exposure to SMOTE algorithms.

5.3 Recommendations for Reproducibility

Some recommendations for reproducibility include providing more in-depth information about the algorithms and models used if no source code is provided. It can be extremely helpful to have more information regarding how the models are created and trained because leaving room for assumption for things like hyperparameters can affect the performance. Information regarding implementation of custom algorithms can also aid in the process of reproduction since using predefined algorithms can negatively affect the performance.

6 Communication with Original Authors

An attempt to communicate with the original authors was made. Using the email provided in their original paper, we attempted to contact Dr. Stephen G. Fashoto but unfortunately have not heard back from them.

References

Adebowale O. Adejumo, Nehemiah A. Ikoba, Esivue A. Suleiman, Hilary I. Okagbue, Pelumi E. Oguntunde, Oluwole A. Odetunmibi, and Obalowu Job. 2017. Quantitative exploration of factors influencing psychotic disorder ailments in nigeria. *Data in Brief*, 14:175–185.

Israel Elujide, Stephen G. Fashoto, Bunmi Fashoto, Elliot Mbunge, Sakinat O. Folorunso, and Jeremiah O. Olamijuwon. 2021. Application of deep and machine learning techniques for multi-label classification performance on psychotic disorder diseases. *Informatics in Medicine Unlocked*, 23:100545.