Comparing Complexity of Neural Networks

BIOSTAT823 Statistical Programming for Big Data
Duke University

Jackson Dial

December 2022

1 Abstract

Exploration is conducted on the effect of additional, identical, hidden layers on a neural network's classification ability. This is done in the context of predicting whether an individual in the ICU will die while still in the hospital. Recorded lab values as well as demographic variables are used. Each model is evaluated using the same metrics, and an optimal model is selected. The model with the highest number of hidden layers (6) performed the best under the same conditions as the comparative models. Not only did the selected model perform better than others, but it performed generally well, with an AUROC of 0.767 and overall accuracy of 82.6%. This model could help clinicians in making data-driven decisions when treating ICU patients.

2 Introduction

Intensive Care Units (ICUs) provide critical care for patients facing life-threatening conditions. Various measurements are taken at different times, including many vital measurements. This rich source of data can and should be leveraged to draw insights and support data-driven clinical decisions. This project explores different structures of a tool that uses all available data and could be used in clinical decision-making.

3 Data

Data used for this project was the Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC II) Clinical Database. It is a collection of ICU clinical data collected from 2001-2008 from a single tertiary hospital. It was first downloaded as a set of text files, one for each patient. These files can be found in the "set-a" folder in the referenced GitHub repository, and the file that accomplishes this task is "read_data.py". This resulting .csv file, named "combined_data.csv" is then further processed in the "DataCleaning.R" file. As the "set-a" folder does not have outcome data, the "Outcomes-a" file was processed in a similar way as above, and the resulting file is "outcomes.csv". An important concept for this project is the Last Observation Carried Forward (LOCF) imputation method. This assumes that in time series data, an observed value does not change until it is recorded as changed. This imputation method is used in the "DataCleaning.R" file, as well as other data cleaning such as outlier removal, and a train/test split, using 70% of the data for training. The outputs of this file are "train.csv" and "test.csv". These data files were further transformed in such a way that a time window was created, using the "train_recode.R" and "test_recode.R" files. This time window summarizes a column per 12-hour period. The average value was used, exclusive of missing values. This transformation took data from long format to wide format, where each lab value now has 4 columns, where the values in the column are the average of that lab value over each 12-hour time interval. If there were no observed values for the entirety of a time interval, that value was left missing in the dataset. The resulting datasets are "train_recode.csv" and "test_recode.csv", which were used in the building of the model, found in the "823Final_code.ipynb" file.

4 Methods

A neural network was used to predict "in hospital mortality", using the first 48 hours of recorded data. This model was chosen due to its ability to include high amounts of features, as well as incorporate non-linearity that may be present in the data. It was important to use all features in the data, as each feature may be of different importance for various individuals. The data as described previously was further imputed, using -1 values to replace missing values. This was determined reasonable based on the criteria that -1 is not a possible value for all features and having a numeric value will still allow the back-propagation algorithm to compute the weights and gradients of the network, allowing for some sort of relationship to be explored between those missing values and the outcome. Three separate models were built and evaluated on the same data. A set of layers was added to the second and third models to examine the advantages/disadvantages of such added complexity based on evaluation metrics. The first model included an input layer, a dense layer, a dropout layer, and finally a dense layer. The second model added both a dense and dropout layer after the initial dropout layer, and the third model included a third set of dense and dropout layers. Each of the dense layers (except for the final dense layer) used the same hyperparameters as follows:

Input dimension: 153, the number of input features.

Units: Chosen to be 153, the number of input features.

Activation function: Rectified Linear Unit (ReLU) function, which is MAX(x, 0).

Kernel Initializer: Chosen to be truncated normal, which specifies the statistical distribution that is sampled from to initialize the weights and biases.

The final dense layer used a sigmoid activation function, which takes as input any real number, and will output a value to be on the interval (0,1). The dropout layer randomly sets a specified percentage of weights to 0 to help prevent overfitting. The specified dropout rate was chosen to be 20%. The loss function used was binary cross-entropy, which uses the difference between actual and predicted values to measure the loss, and the optimizer was adaptive moment estimation (Adam). Each model was run over a total of 100 epochs, with a batch size of 32. This means that for one iteration of internal parameter updates, or for each epoch, only 32 data points were used, helping to prevent overfitting. The validation split was set to be 0.3, meaning 30% of the training data was used for validating each iteration of the model. All other hyperparameters used were the default values. The metrics used to compare each of the models are overall accuracy, Area Under the Precision-Recall Curve (AUPRC), and Area Under the Receiver Operator Characteristic Curve (AUROC). The overall accuracy is simply the percentage of observations that the model can correctly predict. The AUPRC can be interpreted as the percentage of the time that the model will correctly assign a higher absolute risk of the outcome to a randomly sampled individual who did experience the outcome, than to a randomly sampled individual who did not experience the outcome. The AUROC does not account for true negative rate, and thus should only be used when sensitivity is of no concern. In this situation, true negatives are not of primary interest. It is also important to note that given the output of the model is a probability, a cutoff must be made to determine if that observation is labeled as death or not, according to the outcome definition. This cutoff has been determined by the prevalence of the outcome in the data, which is approximately 13

5 Conclusion

Table 1 (Section 7.1) clearly shows that model 3 is the most optimal model. Given the already complex nature of a neural network with just a single hidden layer, tripling the layers as in model 3 does not necessarily add much complexity from an interpretability standpoint. The accuracy of the model is not particularly great, especially considering the prevalence of outcome is 13%, though it can be confidently concluded that the model is in fact learning, and not simply labeling each observation as not experiencing the outcome. The AUPRC of 0.302 is rather good, considering the baseline for this metric is the prevalence of the outcome in the test dataset, 13%. It shows a good ability to correctly classify survivors (no outcome) without incorrectly classifying the non-survivors (experience outcome). The AUROC of 0.767 means that 76.7% of the time, the model will assign a higher absolute risk of outcome to a randomly selected individual who did experience the outcome than to an individual who did not experience the outcome. This metric shows slightly better than a 50% improvement over a coin flip.

6 Limitations/Next Steps

The conclusion of the project is that adding more layers to a neural network led to an overall better model, using the specified metrics for evaluation. A clear next step would be to follow the pattern of adding a hidden layer, ideally until no further improvements are made. Further exploration of hyperparameter tuning would also be important, as these were not explored in this project. Specific hyperparameters that should be explored for their effect on the model are the learning rate, dropout rate, batch size, and units within each dense layer.

7 Appendix

7.1 Table 1

	Model 1	Model 2	Model 3
Overall Accuracy	0.776	0.780	0.826
AUPRC	0.219	0.263	0.302
AUROC	0.656	0.718	0.767