TADPOLE Challenge Project Final Report

Jianhua Fan (jf773), Yanfei Xu (yx427), Haoyuan Zheng (hz463)

November 30, 2017

Background on Machine Learning Problem

The project topic and the dataset is from a data science challenge named The Alzheimer?s Disease Prediction of Longitudinal Evolution, TADPOLE. The prediction of Alzheimer?s Disease is crucial because AD treatment is most likely to be effective at early disease stages and prediction of the change could help access the effect of treatment. In the dataset, we are given regional MRI, PET, DTI and CSF measurements. Using this information, we would employ machine learning algorithm to predict limited clinic information including diagnose type, ADAS13, Ventricles, and MMSE.

Literature Survey

In [1], researchers developed SVM model to classify persons with and without common diseases. 14 features commonly associated with the risk for diabetes were selected, including family history, age, gender, weight, etc. Researchers tried various kernel models and gained 83.47% of classification accuracy which proved that SVM is a promising approach in complex disease detection. The given dataset for this project has thousands of variables for each patient, which must be carefully selected to avoid unnecessary huge computations. In [2], Chen discussed different feature selection strategies including F-score, Random Forest (RF) and radius margin (RM). We also noticed that the given dataset has many missing values, which should be pre-processed first to get better prediction accuracy. In [3], several methods for completing missing values in small samples like last observation carried forward (LOCF) and multiple imputation (MI) approaches was investigated. Researchers compared the performance of the above imputation methods in terms of standardized bias of the mean change estimate and coverage rates and confidence interval widths. It was concluded from the simulation results that the BLS performed the best.

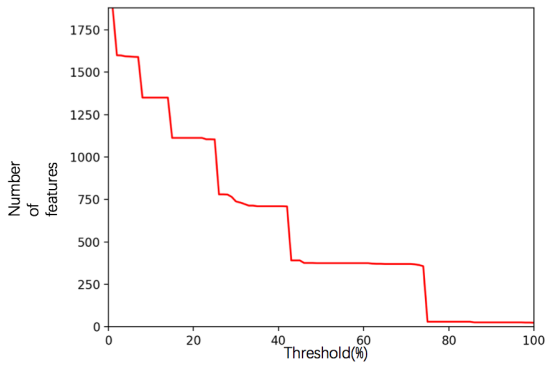

Figure 1: percentage of data filled for each featuer

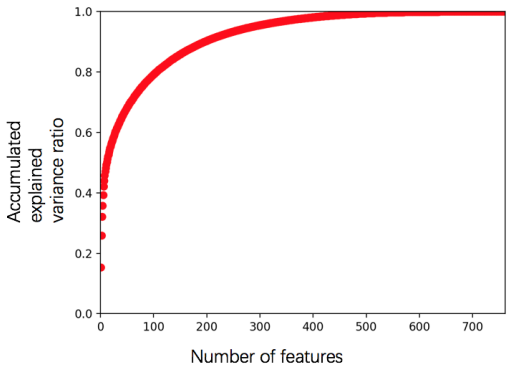

Figure 2: PCA

Pre-processing Steps

Data pre-processing mainly composed of three parts: data-cleaning, imputation and feature selection. Original data is messy and far from enough to be the input of our machine learning models.

1. **Data-cleaning**
   Data with extremely low percent of completeness will bring in lots of noise if we fill them all by means of linear interpolation. Therefore, we first checked percentage of data filled for each feature. The graph was obtained as Figure 1. We chose 70% as our threshold and throw away all the features that below this threshold. Finally we got 370 features for each patient.

   Features of date and labels are in string format, transformations are needed to transform them into scalar. Other examine dates could be roughly represented by the first shown date, so other dates were deleted. Other string information is converted to numbers incrementally and hot encoding is adopted to make it more suitable for this machine learning problem.

2. **Imputation**
   We tried various methods of imputation and finally used LOCF and median imputation. A LOCF algorithm was written to fill in the missing data with closest previous data if there are previous data. However, for some feature value, no previous value is found so LOCF can?t only fill up to 86% percent of the missing data, as a result, another base imputation method called median imputation was adopted to fill the remaining missing value. Since there exists hot encoding values which need integer value, so we use median rather than mean as our method to impute. In our baseline SVM and SVR algorithm, intervals between input and target of a same PID will be considered as a parameter of input to train the model. This part will be further

discussed in SVM model.

3. **Feature Selection**
   We also did Scaling and PCA. Given the accumulated explained variance ratio graph for training input and test input respectively (see Figure 2 for training data), we selected 242 features with at least 92% percent as a threshold to select relatively fewer number of transformed feature. However, in the later implementation of model, data without PCA seems to perform better in SVC/CVR. Some other feature selection method like variance threshold has also been tested, but it turned out both PCA and other feature selection methods make little difference to results while scaling is quite important.

Baseline Algorithm

The baseline approach is the Support Vector Machine. The diagnosis result could be predicted using multi-class classification SVM. For the other three quantitative/ continuous variables, we used support vector regression (SVR) algorithm to predict their future values.

1. **Input and Output**
   Input data is composed of selected features, all of them have been imputed using the strategy discussed above. The time intervals between the exam date and target date is a core issue in building the SVR/SVC model. The form of Input could be described as follows:

   $$x^i = [\Delta t, x_1, x_2, x_3, ..., x_{m-1}]$$

   Each x^i is a single input vector in SVM. $x_1, ..., x_{m_1}$ are selected features from single visit. $\Delta t = t_v - t_x$, t_v is the time in target dataset while t_x is the time in input dataset. t_x and t_v are paired in the following way: for all the vectors with a same PID in both input and target dataset, we pair each vector in input data with each vector in output dataset. Exam date in the output was minused by exam date in the input and get a time interval as a new input value replacing the original exam date. The problem of predicting value at future time-points is now transformed into given data excluding time to predict value after a time interval.

   There are six output variables to be predicted, three of them are mutually exclusive categories variables. So, we treat the classification problem as multiclass classification problem. As for the other continuous variables, we use SVR model to train them separately.

2. **Result and Analysis**

   Table 1 are our prediction accuracy and mean absolute difference using SVC and SVR model. In this way, we have prediction accuracy and mean absolute difference as follows. (best kernel for each variable is marked ?red?). Additionally, we plot the ROC

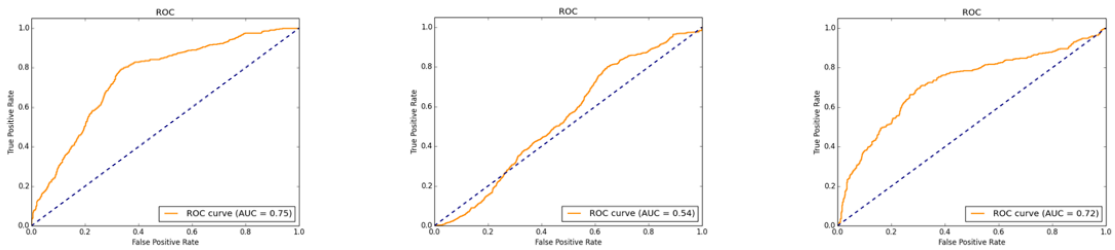

Figure 3: ROC curve

Output variable	sigmoid	rbf	linear	poly
CN_Diag, MCI_Diag, AD_Diag	45.0%	48.7%	51.5%	54.2%

Table 1: Prediction Accuracy Using Different Kernels

curve for each diagnosis variable as shown in Figure 4 and compute the mAUC. AUC for each variable are 0.75, 0.54, 0.72 separately, So $mAUC = (0.75 + 0.54 + 0.72)/3 = 0.67$.

For each output variable, we choose kernel that achieved highest accuracy and smallest mean absolute difference, so for the first three variables, we use poly kernel; for MMSE, we use rbf kernel.

Proposed Algorithm

The second approach is RNN (recurrent neural network). RNN model is especially useful when dealing with time series data set. In this TADPOLE challenge, the input data for each patient is a sequential of visiting indicators with respect to time. So we developed a LSTM multi-layer model for prediction of all target variables.

1. **Overview of RNN Model**
   In our RNN model, we used ?many to one? to denote the relation between input data

Output variable	sigmoid	rbf	linear	poly
ADAS	136.8	7.87	8.18	7.87
Ventricles Norm	0.017	0.017	0.017	0.017
MMSE	133.2	2.66	2.75	2.79

Table 2: Mean Absolute Different Using Different Kernels

and output data. In addition, we adopted LSTM (Long Short Term Memory) as RNN cell because it can effectively avoid gradient vanishing or explosion. [4]

2. **Input and Output**
In the training dataset, the input data and output data for each patient are both sequences. To denote the difference between different dates of output data for the same patient, we add additional input variable: time difference between each date of input data and each date of output data:

$$x_i = [\Delta t, f_1, f_2, f_3, ..., f_m]$$

where x_i is the input of the RNN, Δt is the time difference between visiting time of patients in input data and time in target data, $f_1, f_2, f_3, ..., f_m$ are other selected features, similar to the definition of input in SVR approach. Normally for each time step RNN model will produce one output, in our model, we just use the output from the last time step as final output. Figure 6 shows the process.
Then we add a dense layer (fully connected layer) to make our RNN model work better. In our problem, the first three variables belong to mutually exclusively categories. Thus we used a softmax layer to transform the output of the neural network to the probability for each class. On the other hand, the last three variables are continuous, so we did not use softmax layer or cross entropy when predicting these variables. Instead, we use MAE (mean absolute error) as loss function. The output for the regression task becomes just one variable. The LSTM model is shown in Figure 7.

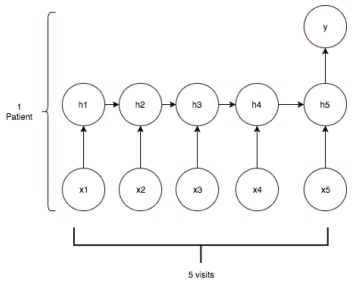

Figure 4: Input data and output data of LSTM

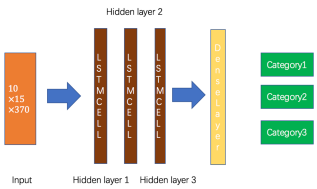

Figure 5: The architecture of LSTM model

	Accuracy of Diagnosis	MAE for ADAS	MAE for Ventricles Norm	MAE for MMSE
train	93.3%	2.21	0.0078	3.58
test	77.5%	4.93	0.0096	3.99
validation	77.0%	4.95	0.0084	4.23

Table 3: Results of LSTM Model

	layers	hidden units	batch size	max epochs	learning rate	sequence length
Diagnosis	3	128	10	80	0.001	15
ADAS	2	512	100	100	0.001	15
Ventricles	3	256	300	70	0.001	15
MMSE	3	256	300	200	0.001	15

Table 4: Parameters of LSTM Model

3. **Methods**
   For optimizer, we use Adam optimizer. As each patient?s visiting time is different, the sequence length for different sequence is not the same. So we use zero-padding to make their length equal. We compute the length of each sequence and find the maximum value is 15. So we make each sequence length become 15 by adding extra zero vectors. Since all the extra padding vectors are zero, weights will not affect outputs and don?t get trained on them. To prevent overfitting, we used dropout to randomly drop connections between some neurons. We set the drop out rate of input for each layer is 0.2, probability of dropping out output is 0.5. In addition, we observed the training loss curve during training and early stopped the training process if we thought that the training has been overfitting.

4. **Results and Analysis**
   We listed our training, test, validation results for each of the target variables in Table 3. Figure 8, 9, 10, 11 shows corresponding training curve.

   In addition, we listed the detailed parameters of our LSTM model in Table4.

   From the comparison of training accuracy and test (validation) accuracy we can see that the training is overfitting, because our model can achieve 93.3% classification accuracy while only get around 77% accuracy on test and validation data. Similar situation happens on the regression of ADAS, which gets 2.21 for training loss while gets nearly 5 for test and validation loss. Our model performs similarly on training, test and validation for prediction of Ventricles Norm and MMSE. By comparing the results of our baseline algorithm (SVM) and proposed algorithm (RNN), we find that RNN

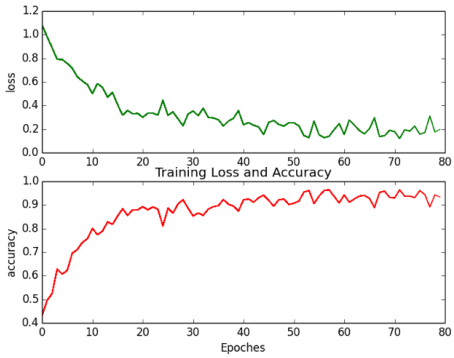

Figure 6: Training curve for diagnosis

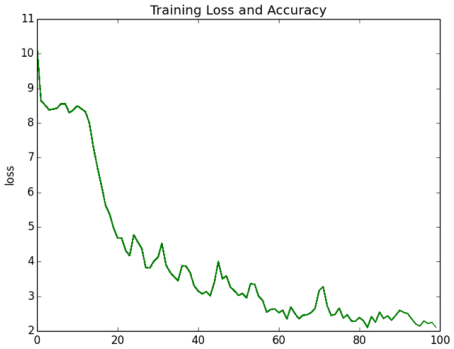

Figure 7: Training curve for Ventricles Norm

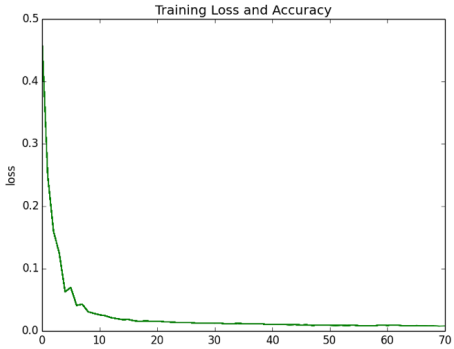

Figure 8: Training curve for ADAS

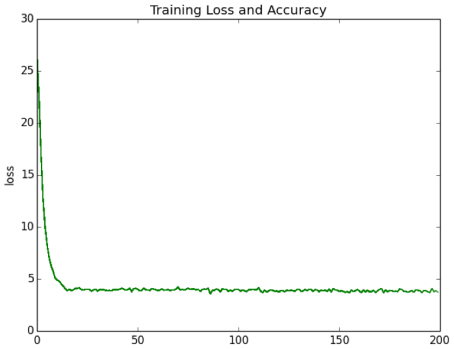

Figure 9: Training curve for MMSE

model performs much better when predicting the diagnosis variables which improve the accuracy from 54.2% to 77.5%. In addition, RNN model performs better when predicts ADAS and Ventricals Norm which decreased the mean absolute error from 7.87 to 4.93, from 0.017 to 0.0084 respectively. However, SVM performs better when predicting MMSE, which achieved 2.66 test MSE when using rbf kernel, while RNN model got 3.99 test MSE. Also we found that although we changed many parameters of our RNN model, the training loss cannot attain very low when the training process converges, and it always converges well. So we think the data for MMSE has not much inner relation and may be more suitable to use SVM than complex models like multilayer LSTMs.

Conclusion and Discussion

Although we adjusted listed parameters in Table 1 to attain best results as we can, the results of our model might still need improvement by considering the following aspects:

1. Both the input data and output data are sequences, so using sequence-to-sequence LSTM model might perform better.

2. As the limitation of time and our knowledge about neural networks, we have not fully investigated the influence of different parameters on the performance of LSTM model. Thus if we find the most suitable parameters of LSTM model for each prediction task, the results might achieve better.

3. Although we used drop out and early stop to prevent overfitting, we might still need more effective methods including regularization, gradient clipping, ensembling, etc.

4. Imputation strategy could be further improved. We have found that avoiding imputation for training data could result in obvious progress in result, more research on imputation strategy may help promote result further.

Contribution

Yanfei Xu (yx427) : responsible for data analysis and preprocessing, baseline model and RNN model parameters tuning and RNN model training and analysis, etc.
Haoyuan Zheng (hz463): responsible for data preprocessing, baseline model and RNN model discussion and optimization, etc.
Jianhua Fan (jf773): responsible for development of SVM model and RNN model, model optimization and analysis, etc.

References

[1] Yu, Wei, et al. "Application of support vector machine modeling for prediction of common diseases: the case of diabetes and pre-diabetes." BMC Medical Informatics and Decision Making 10.1 (2010): 16.
[2] Chen, Yi-Wei, and Chih-Jen Lin. "Combining SVMs with various feature selection strategies." Feature extraction (2006): 315-324.
[3] Sunni, Stacy, et al. ?Multiple imputation techniques in small sample clinic trials?. STATISTICS IN MEDICINE, Statist. Med. 2006; 25:233-245.
[4] LeCun Y, Bengio Y, Hinton G. Deep learning[J]. Nature, 2015, 521(7553): 436-444.