Classification of Body-Rocking Behavior

Ziad Ali
Electrical Engineering
North Carolina State University
Raleigh, US
zaali@ncsu.edu

Anusha Manur
Computer Science
North Carolina State University
Raleigh, US
amanur@ncsu.edu

Shashank Shekhar
Computer Science
North Carolina State University
Raleigh, US
sshekha4@ncsu.edu

I. Methodology

Our goal was to implement an RNN-based machine learning framework that could be used to detect body-rocking behavior from blind subjects. The data was obtained with inertial sensors on the subjects' wrists and arms using both accelerometers and gyroscopes. An LSTM (long short-term memory) network was specifically used due to its ability to capture long-term data [1].

Previous work investigated the classification of body-rocking behavior using non-sequential machine learning frameworks (e.g. random forest, k-nearest neighbor, multi-layer perceptron). In this work, the efficacy of the LSTM model will be compared against the efficacy of the optimal non-sequential model developed previously.

To create the LSTM, the LSTM class within the Layers module of the Keras library was used. A Sequential model was created with all of the LSTM layers added first, followed by a dropout layer and then a dense layer with a single output (sigmoid activation). The binary cross entropy loss function was used with an Adam optimizer.

Initially, the LSTM network was trained by passing in individual timesteps of the arm and wrist accelerometer/gyroscope data. However, it quickly became apparent that this method was not effective (maximum validation F1 scores of approximately 0.78); instead, certain features of the recorded data (mean, covariance, skewness, kurtosis, frequency response) were calculated over "windows" of the original data and fed into the LSTM network as the input data. The length of the window over which these features were calculated was determined according to the "Window Length" hyper-parameter. The variation in the window's position for each iteration of training was determined by the "Slide Length" hyper-parameter. Since single predictions were generated for entire windows of data, prediction arrays were "expanded" such that each prediction was repeated (an amount of times equivalent to the slide length) to have a one-to-one correspondence with the ground truth data.

II. Model Training and Hyper-parameter Selection

To develop the LSTM model, the hyper-parameters shown in Table I were used. The 10 sessions of provided data were split into 8 sessions of training data and 2 sessions of validation data. While a grid search or random search would have been optimal to select the best combination of hyper-parameters, a lack of training time and the amount of hyper-parameters possible made these options infeasible. Rather, each hyper-parameter in the list was individually tuned to select for the highest F1 score. Thus, the LSTM units were the first hyper-parameters evaluated, then the LSTM layers were evaluated, then the batch size, and so forth. The initial hyper-parameter settings are shown in Table II.

TABLE I
Model Hyper-Parameter Values

Hyper-Parameter	**Values**
LSTM Units	50, 100, 200, 400
LSTM Layers	1, 2, 3, 4
Batch Size	8, 16, 32, 64, 128, 256
Dropout	0.3, 0.5, 0.8
Window Length	100, 200, 300, 400
Slide Length	10, 25, 50

TABLE II
Model Hyper-Parameter Values

Units	Layers	Batch Size	Dropout	Window	Slide
200	1	16	0.5	300	50

III. Evaluation

The results of the LSTM units hyper-parameter testing are shown in Fig. 1. The F1 score was the metric selected for - thus, 200 units was selected as the optimal LSTM unit size.

Next, the remaining hyper-parameters were evaluated in order - the layers evaluation is shown in Fig. 2, the batch size evaluation is shown in Fig. 3, the dropout evaluation is shown in Fig. 4, the window length evaluation is shown in Fig. 5, and the slide length evaluation is shown in Fig. 6. The final hyper-parameters selected are shown in Table III.

TABLE III
Model Hyper-Parameter Values

Units	Layers	Batch Size	Dropout	Window	Slide
200	2	128	0.5	300	50

When the final model was evaluated on the validation set, the following metrics were obtained: accuracy = 83.64%, recall = 77.29%, precision = 93.17%, and F1 score = 84.49%.

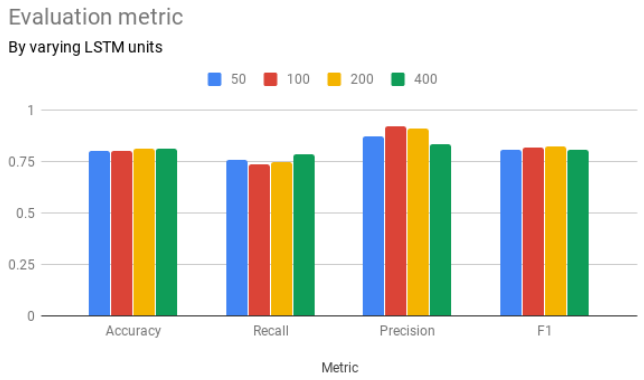

Fig. 1. Confusion matrix derivations (accuracy, recall, precision, and F1 score) shown for different numbers of units in LSTMs in model architecture. LSTMs with 200 units performed best with regards to F1 score.

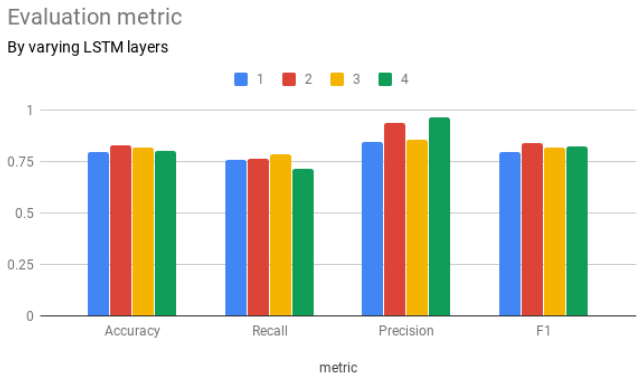

Fig. 2. Confusion matrix derivations (accuracy, recall, precision, and F1 score) shown for different numbers of layers in the model architecture. Models with 2 LSTM layers performed best with regards to F1 score.

Previously, the best model examined (which used the random forest framework) achieved an accuracy of 95.60%, a recall of 94.06%, a precision of 82.06%, and an F1 score of 87.65% on the validation set. The primary difference between these two models is that the random forest model had a substantially higher recall and accuracy, albeit a lower precision. This suggests that the random forest model currently outperforms the LSTM model.

Since LSTM models are optimized to make classifications for sequential data, it is likely that more work remains to be done to improve the model used. Possible improvements in the future include eschewing the method of extracting features and instead returning to training the LSTM on the raw data and adding in convolutional neural networks to extract features automatically.

The performance of the LSTM model for a section of the validation set is shown in Fig. 7 (a slight vertical offset is added to the ground truth values so that both graphs can be observed). For this section of the validation set, the LSTM model precisely tracks the real detections.

The performance of the LSTM model for a separate section of the validation set is shown in Fig. 8. Here, the tendency for the model to make false positive predictions is demonstrated. The key to improving the model in the future will be to decrease the amount of false positive predictions while maintaining the currently high rate of true positive predictions.

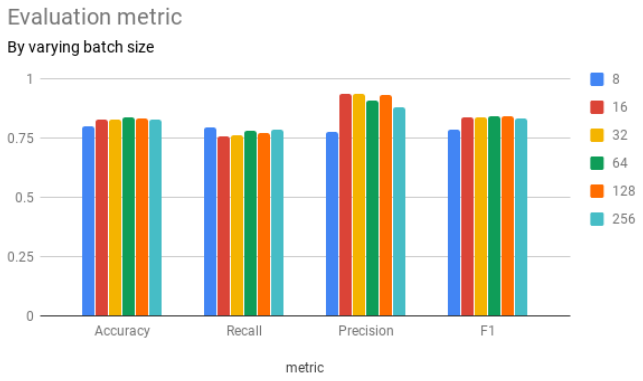

Fig. 3. Confusion matrix derivations (accuracy, recall, precision, and F1 score) shown for different batch sizes in model architecture. Models with a batch size of 128 performed best with regards to F1 score.

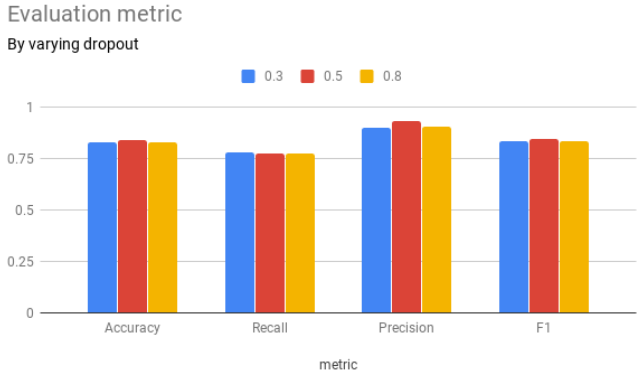

Fig. 4. Confusion matrix derivations (accuracy, recall, precision, and F1 score) shown for different dropout rates in model architecture. Models with a dropout rate of 0.5 before the final dense layer performed best with regards to F1 score.

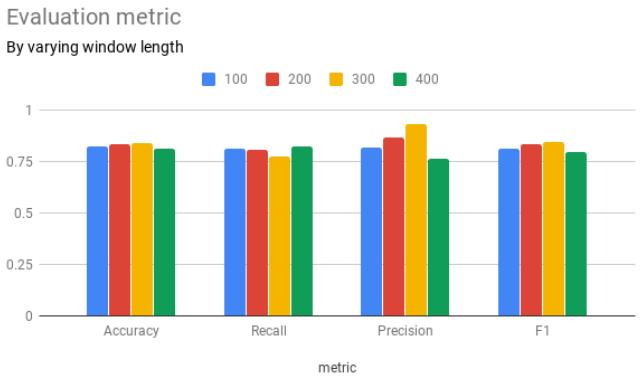

Fig. 5. Confusion matrix derivations (accuracy, recall, precision, and F1 score) shown for different window lengths for the model architecture. Models with windows of 300 units performed best with regards to F1 score.

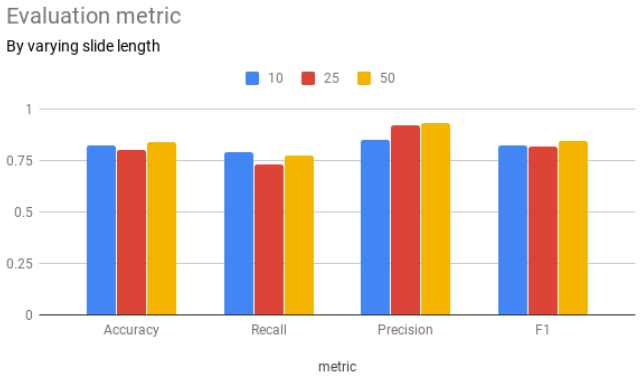

Fig. 6. Confusion matrix derivations (accuracy, recall, precision, and F1 score) shown for different slide lengths for the model architecture. Models with slide lengths of 50 units performed best with regards to F1 score.

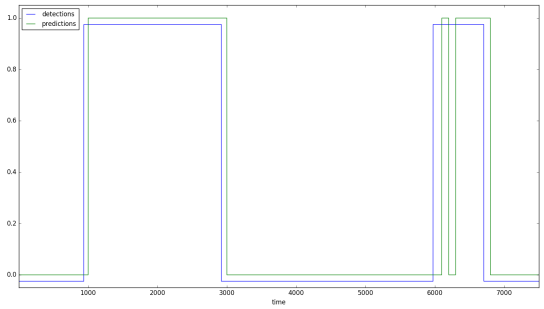

Fig. 7. Ground truth values (classification of body rocking behavior) plotted in blue and predicted values plotted in green. In this scenario, the predictions track the real values closely, indicating the model is precise.

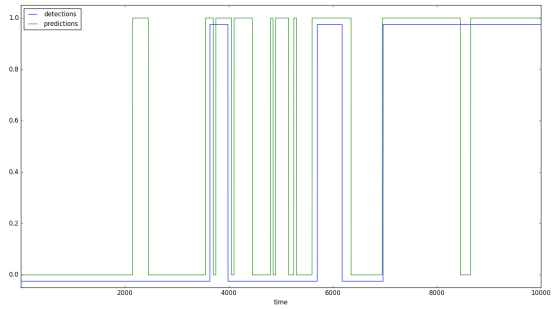

Fig. 8. Ground truth values (classification of body rocking behavior) plotted in blue and predicted values plotted in green. In this scenario, there are many false positive predictions, indicating the model needs to be improved in the future.

REFERENCES

[1] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, LSTM: A Search Space Odyssey, in IEEE Transactions on Neural Networks and Learning Systems, vol. 28 (10), 2017, 2222-2232.