[Team 14] Proj-C2: Terrain Identification from IMU Streams using an Recurrent Neural Network

Vaishnavi Venkat (vvenka23), Sam Hanson (schanson), Caizhi Huang (chuang19)

I. Methodology

Humans instinctively develop the capacity to walk in a way that is energy efficient, stable, and environment adaptive. Lower-limb robotic prostheses can benefit from this kind of context awareness to provide more comfort and safety to amputees. To address this, researchers aim to develop a terrain identification system based on inertial measurement units (IMU) streams collected from the lower limb [1]. In this project, we aim to identify terrains with only the IMU sensor data. With such information, the control of a robotized prosthetic leg can be adapted to changes in its surrounding. A machine learning approach can be used to learn information from the IMU sensors and try to identify different terrains from it. In the second part of this project, we have used recurrent neural network (RNN) with long short term memory (LSTM) cell to handle this task.

A. Dataset Description

The dataset is composed of 29 sessions of IMU data (i.e., 3 accelerometers and 3 gyroscope measurements from the lower limb) from 8 subjects and corresponding labels of terrain type. Each session lasts from 8 to 30 minutes sampled at 40Hz and 10Hz for IMU data and terrain labels respectively. Figure 1 shows an example of the time series IMU data in a 100 second window.

The terrain types are annotated from a synchronized data stream, (0) standing or walking on solid ground, (1) going down the stairs, (2) going up the stairs, and (3) walking on grass. The dataset was highly skewed towards label 0 (standing/walking on solid ground).

B. Data Processing and Model Implementation

The IMU data was sliced into 2-second windows so that each window features 80 time points (40 Hz sampling rate). Thus, the dimension of each window is 80×6. The windows are shifted every 20 time points so that there is a 75% overlap between adjacent windows. There are 20 labels for each window and we obtained a single label for each window by calculating the mode of labels. Additionally, all the IMU data was scaled to give it a mean of 0 and a variance of 1.

We implemented the RNN with LSTM cell using the Keras package from TensorFlow [2]. Figure 2 features the RNN model summary.

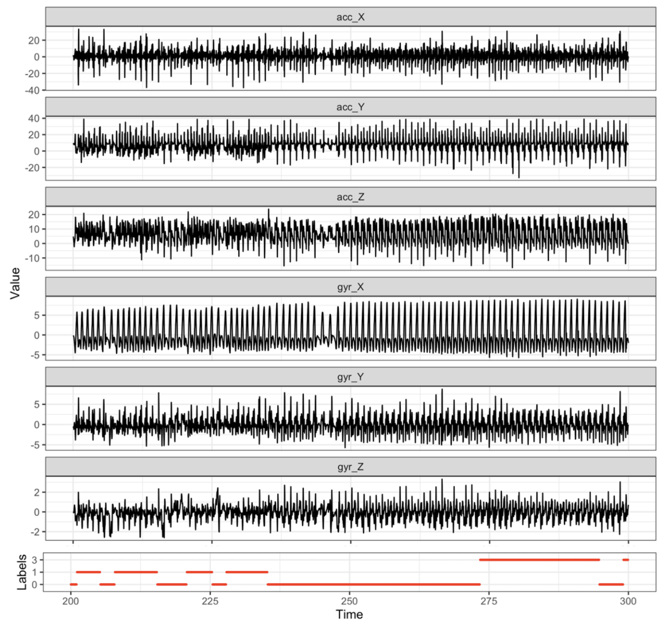

Fig. 1. Example of IMU data and labels using secession 1 of subject 1 from 200 to 300 seconds.

Layer (type)	Output Shape	Param #
lstm_11 (LSTM)	(None, 80, 128)	69120
batch_normalization_16 (BatchNormalization)	(None, 80, 128)	512
dropout_16 (Dropout)	(None, 80, 128)	0
lstm_12 (LSTM)	(None, 128)	131584
batch_normalization_17 (BatchNormalization)	(None, 128)	512
dropout_17 (Dropout)	(None, 128)	0
dense_11 (Dense)	(None, 128)	16512
batch_normalization_18 (BatchNormalization)	(None, 128)	512
dropout_18 (Dropout)	(None, 128)	0
dense_12 (Dense)	(None, 4)	516

Total params: 219,268
Trainable params: 218,500
Non-trainable params: 768

Fig. 2. Summary of the RNN model with LSTM cells.

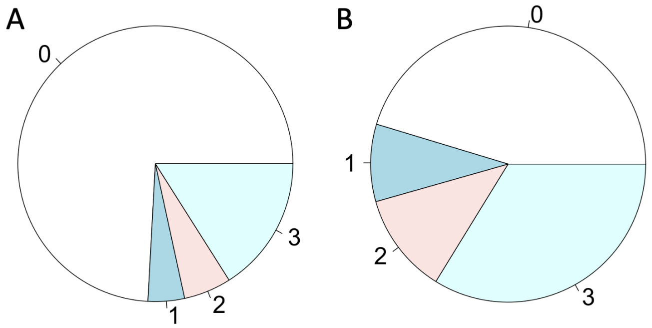

Fig. 3. The pie chart of the number of windows for each label of terrain type. **A**: before downsize label 0; **B**: after downsize label 0.

II. Model Training and Selection

A. Model Training

Several methods of splitting into validation and test sets were attempted. Because the testing data set is from entirely independent subjects, 4 sessions from subjects 6 and 9 were kept separate as validation set. The remaining 25 sessions make up the training set. To handle the imbalance of number of windows for each label, we downsized the label 0 terrain type to match the number of windows for label 3 (Figure 3).

B. Model Selection

The cross-entropy loss and the Adam Optimizer were used for this RNN model. We tried different hyperparameters for number of epochs, batch size and learning rate. Loss and Accuracies of previous generations of the model which experimental hyperparameters are featured in the Appendix, these show higher batch sizes and smaller dropout rates. The final model used 30 epochs, a learning rate of 0.01, and a batch size of 60. We visualized the learning curve using the loss and accuracy and selected the model with highest accuracy (Figure 4).

III. Evaluation

We used the model with the best accuracy in the training at epoch 30. Table I shows the confusion matrices for the training set using 25 sessions. The overall training accuracy is 0.95.

TABLE I
Confusion Matrix for training set

Label	0	1	2	3
0	12062	31	54	69
1	35	2407	0	0
2	54	0	3126	0
3	264	2	0	41259

The overall F1 score obtained on the validation set is 92%. The confusion matrix for the predictions on the validation set are as shown in Table II. A better understanding of the performance of the model can be obtained with other metrics such as precision, recall, F1 scores and balanced accuracy. The results of these metrics for each class in the validation set as shown in Table III.

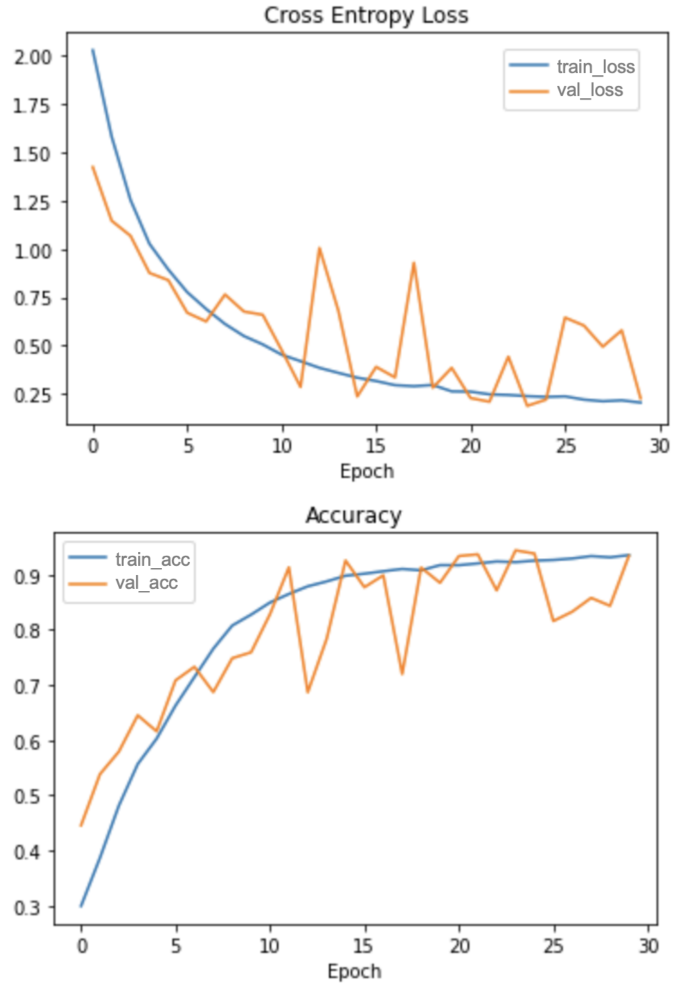

Fig. 4. The learning curve of loss and accuracy by epoch.

TABLE II
Confusion Matrix for validation set

Label	0	1	2	3
0	39697	307	205	785
1	66	1299	0	5
2	27	0	2178	0
3	385	0	3	5282

The best model was also used to obtain the predictions for the test set provided. The distributions of each predicted label on each test sample is as shown in Table IV.

TABLE III
CLASSIFICATION STATISTICS ON VALIDATION SET

	Precision	Recall	F1 Score	Accuracy
Type: 0	0.97	0.99	0.98	0.99
Type: 1	0.95	0.81	0.87	0.80
Type: 2	0.99	0.91	0.95	0.91
Type: 3	0.93	0.87	0.90	0.87
Average:	0.96	0.89	0.92	0.89

TABLE IV
SUMMARY OF PREDICTION FOR TESTING SETS

Label	Subject 09	Subject 10	Subject 11	Subject 12
0	8780	7781	9200	8891
1	398	441	507	642
2	145	449	785	868
3	175	3599	2338	929

REFERENCES

[1] Boxuan Zhong, Rafael Luiz Da Silva, Minhan Li, He Huang, and Edgar Lobaton. Environmental context prediction for lower limb prostheses with uncertainty quantification. *IEEE Transactions on Automation Science and Engineering*, 18(2):458–470, 2020.

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

IV. APPENDIX

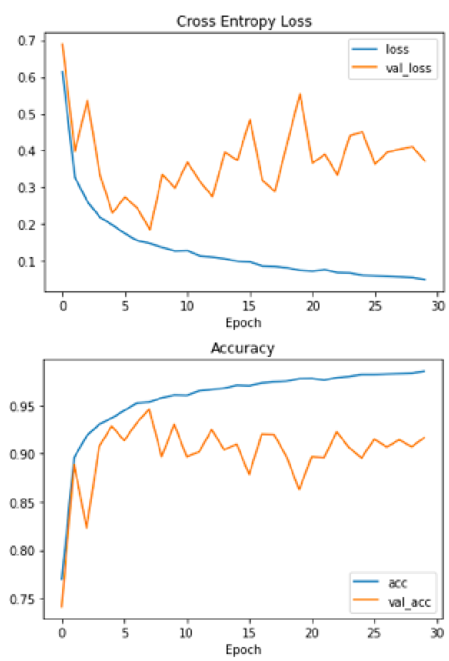

Fig. 5. The learning curve of loss and accuracy by epoch of an earlier model with a higher batch size, lower dropout, and lower learning rate.

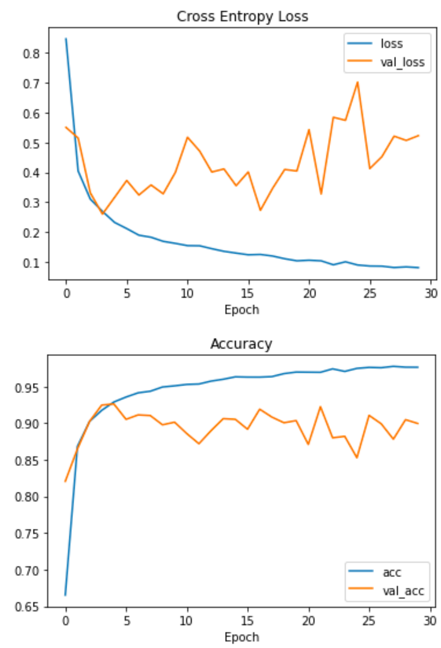

Fig. 6. The learning curve of loss and accuracy by epoch of an earlier model with a higher batch size, and lower learning rate.