# [Team 4] Proj-C1: Terrain Identification from Time Series Data

Kaushik Jadhav (kajadhav), Ajith Kumar Vinayakamoorthy Patchaimayil (avinaya), Aditya Srivastava (asrivas7)

## I. Methodology

Lower limb amputation disrupts the natural, energy efficient walking capability of human beings. Lower-limb robotic prosthetics can benefit from environmental context awareness to provide better comfort to their user [2]. In this work, we develop a terrain identification system based on IMU (inertial measurement units) streams collected from the lower limb. Data will consist of several sessions from 6 different subjects including IMU data from a sensor on the leg of a participant, and the labels come from annotations of terrain type from a synchronized data stream [3]. A machine learning algorithm can be used to classify the samples into different classes like standing or walking in solid ground, going down the stairs, going up the stairs, and walking on grass.

The dataset is highly imbalanced and contains 4 types of files: The ”_x” files contain the xyz accelerometers and xyz gyroscope measurements from the lower limb. The ”_x_time” files contain the time stamps for the accelerometer and gyroscope measurements. The units are in seconds and the sampling rate is 40 Hz. The ”_y” files contain the labels. (0) indicates standing or walking in solid ground, (1) indicates going down the stairs, (2) indicates going up the stairs, and (3) indicates walking on grass. The ”_y_time” files contain the time stamps for the labels. The units are in seconds and the sampling rates is 10 Hz.

The model chosen by us for the collaboration phase of the project is a Random Forest Classifier. But before training, the dataset has many problems. Firstly, we observe that there are separate files for data, timestamps and labels and that only 1/4 of the data is labelled. We cater to this issue by conctanetaing dataframes, then performing full outer join. And then finally, interpolating missing labels with existing values. However, even after interpolation, the data becomes imbalanced. So we take care of this using the SMOTE (Synthetic Minority Oversampling TEchnique) oversampling. Then we split the data into training and validation sets with an 80:20 ratio and perform training and prediction and generate our metrics. These procedures are described in detail in the next sections.

## II. Data Pre-processing

### A. Concatenation

The ”_x” files contain our training features and ”_x_time” files contain their respective timestamps. Similarly, the ”_y” files contain our training labels and ”_y_time” files contain their timestamps. So we first merge the x and _x_time dataframes to get X dataframe and y and _y_time dataframes to get Y dataframe. So, the dataframe X has columns acc_x, acc_y, acc_z, gyro_x, gyro_y, gyro_z and time and Y has labels and time. However, we see that X has shape (48138, 7) and Y has shape (12035, 2). To resolve this and get a consistent shape of data of (48138, 7), we perform Full Outer Join based on the time column.

### B. Full Outer Join based on time

We perform full outer join of the X and Y dataframes based on their common attribute of timestamps. However, even after this step the data is not completely pre-processed as we see that only 1/4 of the data has labels. To handle these missing labels, we interpolate the missing labels using the DataFrame.interpolate() function of pandas. We set interpolation method to ‘pad’: Fill in NaNs using existing values.

### C. Handling missing labels

After performing full outer join, we still observe that the labels column of the merged dataframe has a lot of missing values. As the sampling rate for the X input stream and y input stream attributes is different, only 1/4 of the data is actually labelled. So, we interpolate the missing labels with existing values using the df.interpolate(’pad’) method provided of pandas. So after performing interpolation, we don’t have any rows with non-numeric or otherwise invalid labels. However, as shown in Fig 1 below, now we notice that the class labels are not equally distributed and so the data now needs to be balanced.

### D. Handling imbalanced data

Even after full outer join and interpolation, the data was found to have unbalanced labels. As shown in the below figure 1, class 0 had many more samples compared to other classes. We used SMOTE (Synthetic Minority Oversampling TEchnique) oversampling to balance this after which the data contains balanced labels and looks as shown in figure 2.

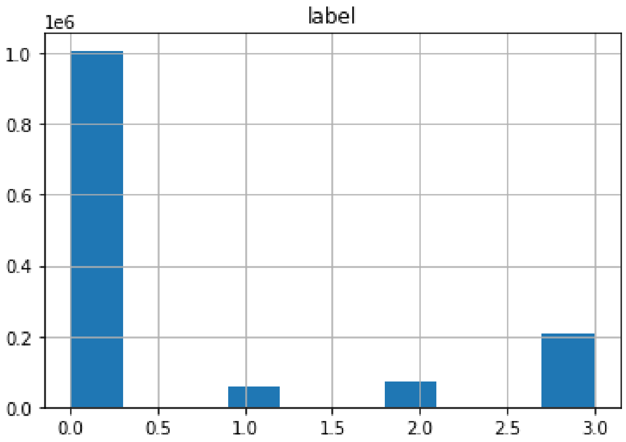


Fig. 1. Data before SMOTE oversampling

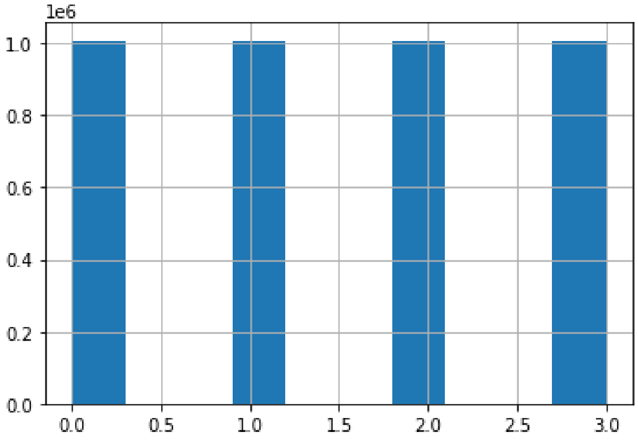


Fig. 2. Data after SMOTE oversampling

## III. Model Training & Selection

After the initial merging, joining, interpolation and oversampling, we perform model training and selection for our data followed by validation testing and ultimately the final prediction generation. First, we split the data into training and validation sets, then choose a suitable model and train it on the data. Then we perform hyperparameter tuning for the model and finally evaluate the model's performance based on metrics like classification report, confusion matrix, etc.

### A. Model Training

We split the final merged and preprocessed dataset into training and validation sets. 80% of the data is used for training and 20% for validation. The model chosen by us for the collaboration phase is a classical model of RandomForestClassifier. We train the RandomForestClassifier on a total of 100 estimators, that is 100 trees on the runtime provided by Google Colaboratory. Colaboratory provides a Tesla K80 GPU and a 2 core Intel Xeon processor with 25 GB of RAM and 12GB of VRAM [1]. This allowed for fast data processing and fast training times. We train the model on the training set and perform hyperparameter tuning for certain parameters towards the end. This is discussed in more detail in the next section.

### B. Model Selection

Next, the data was validated on the 20% split that was kept as validation dataset. The results of validation are summarized in the evalution section. Although the initial model performance on the validation set was nice, we managed to improve it further by hyperparameter tuning. We used GridSearchCV for tuning the hyperparameters of our RandomForestClassifier. The parameters that we tweaked using GridSearchCV were n_estimators, min_samples_leaf, min_samples_split, max_features, max_depth and bootstrap. We got an improvement of 2% on the validation set accuracy after hyperparameter tuning. These improved results along with other metrics and their significance are all discussed in the next section.

## IV. Evaluation

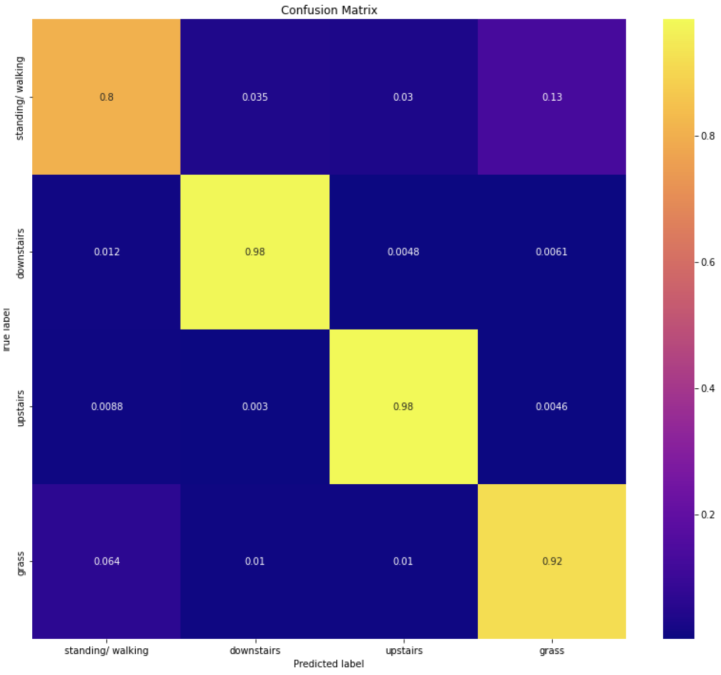
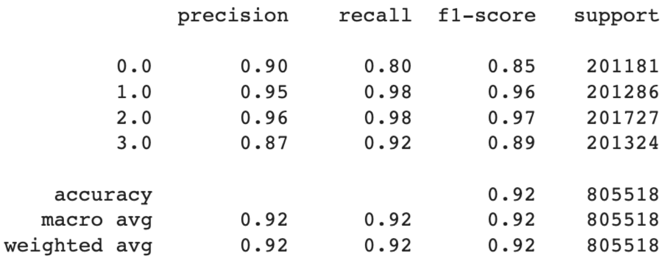


Fig. 3. Confusion Matrix



```
              precision    recall  f1-score   support

         0.0       0.90      0.80      0.85    201181
         1.0       0.95      0.98      0.96    201286
         2.0       0.96      0.98      0.97    201727
         3.0       0.87      0.92      0.89    201324

    accuracy                           0.92    805518
   macro avg       0.92      0.92      0.92    805518
weighted avg       0.92      0.92      0.92    805518
```

Fig. 4. Classification Report

The model is finally evaluated on the 20% data kept as validation set. We got a final validation set accuracy of 91%. To better evaluate the performance of this model, we examined the precision, recall, and F1 scores of each model classification type. Additionally, we generated a confusion matrix to show more explicitly how the model performed with each class type. Figures 3 and 4 show how well the model performed for each class. We can conclued that the model is having a decent accuracy despite it being a classical approach and we hope to built on top of this and improve above metrics even further in the competition phase of the project.

## References

[1] B. Zhong, R. L. d. Silva, M. Li, H. Huang and E. Lobaton, "Environmental Context Prediction for Lower Limb Prostheses With Uncertainty Quantification," in IEEE Transactions on Automation Science and Engineering, vol. 18, no. 2, pp. 458-470, April 2021, doi: 10.1109/TASE.2020.2993399.

[2] Zhu, L.; Wang, Z.; Ning, Z.; Zhang, Y.; Liu, Y.; Cao, W.; Wu, X.; Chen, C. A Novel Motion Intention Recognition Approach for Soft Exoskeleton via IMU. Electronics 2020, 9, 2176. https://doi.org/10.3390/electronics9122176

[3] https://docs.google.com/document/d/1DA-0_IYSno_OREDATejhvXCsjscPQaDK

[4] Rashid, Khandakar M., and Joseph Louis. "Window-warping: a time series data augmentation of IMU data for construction equipment activity identification." ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction. Vol. 36. IAARC Publications, 2019.

[5] Wang, Zhuo, et al. "Terrain recognition and gait cycle prediction using IMU." 2021 IEEE International Conference on Real-time Computing and Robotics (RCAR). IEEE, 2021.