[Team 114]Proj-C2: Terrain Identification from Time Series Data using a Basic Neural Net classifier.

Ramachandran Sekanipuram Srikanthan
Unity Id: rsekani

Chinmayee Dande
Unity Id: cpdande

Mayur Sanap
Unity Id: msanap

I. METHODOLOGY

Our aim in the initial phase of the project is to employ simple neural networks to perform terrain identification through the classification of time series data. To accomplish this task, we propose the utilization of an Artificial Neural Networks (ANN). This model will serve as the baseline for the project's second phase. The data required for this phase was collected by NCSU's Active Robotic Sensing (ARoS) Laboratory, using inertial sensors on the subjects' arms and wrists, which included both accelerometers and gyroscopes [2]. The purpose of collecting this data was to identify the type of terrain on which a person is walking. The data will be used to develop prosthetic designs that can detect the walking surface and adjust the gait accordingly. Each timestamp in the data consists of six values, with three values from an accelerometer and three from a gyroscope, denoted as x, y, and z values. The sensor data was sampled at a rate of 40 Hertz, while the data labels (type of terrains) were obtained at a rate of 10 Hertz. Table 1 contains the annotated labels for the training data.

TABLE I
TERRAIN LABELLING IN DATASET

Label	Terrain
0	Standing or Walking on solid ground
1	Going Down stairs
2	Going Up stairs
3	Walking on grass

The artificial neural network (ANN) is a machine learning model that simulates the functioning of a human brain in processing and analyzing complex data. ANN is composed of a series of interconnected processing nodes, called neurons, arranged in layers. Each neuron takes input values, processes them through an activation function, and produces an output value, which is then transmitted to the next layer of neurons. ANN is capable of learning complex nonlinear relationships between inputs and outputs through a process called training, which involves adjusting the weights of the connections between neurons to minimize a given error metric through the process of optimization. In this project, we have used a feed-forward neural network architecture with multiple hidden layers, and a soft-max output layer for multi-class classification. The neural network was implemented using the Pytorch deep learning library.

A. Data preprocessing

First, we separate the dataset files into training, validation and testing respectively. This will ensure that the model learns the data pattern for each subject.We divided the data into training data and validation data. The training data consisted of continuous sessions of data for multiple subjects ranging from subject-001 to subject-008[1]. This is being further split into training and validation data with ratio 67% - 33%. Our test data consists of data for multiple subjects ranging from subject-009 to subject-012. The data distribution over training data set can be seen in Fig 1.[1] The sensor data is being sampled

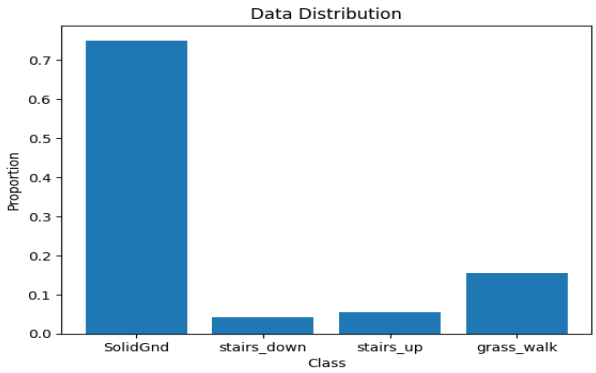

Fig. 1. Data distribution

at 40 Hz and the corresponding terrain is being identified at 10 Hz sampling rate. So for each 4 time-steps in x data, the corresponding y data (label) is being obtained. This data is being pre-processed before passing it to the model. The sensor data from 4 time steps are being merged together thus producing 24 features (4 time-steps with 6 values from both sensors in each time steps) for each label.PyTorch DataLoader object is being used to load the data in batches during training or validation.

II. MODEL TRAINING

A. Model Architecture

After preprocessing the datasets, we perform the following steps to feed the data into our classification model[3]:-

1) Basic (Artificial Neural Network) ANN Classifier:: The 24 input features are being passed to the ANN classifier and the output of this model is passed to softmax layer to perform the classification. The model contains 2 hidden layers with 256 hidden neurons and 128 neurons followed by ReLU activation layer. The final layer contains 4 neurons which are fully connected to produce the classification result. The architecture of the Simple ANN can be seen in Fig 2.

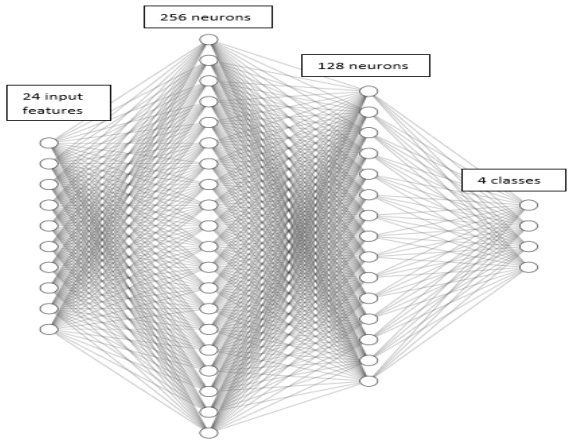

Fig. 2. Simple ANN Architecture

2) Basic (Convolutional Neural Network) CNN Classifier:: We trained using a CNN with a single convolutional layer of kernel size 3x3 and stride 1, a pooling layer with kernel size 2 and stride 2, and two fully connected layers of size 10 neurons with ReLU activation and a 4 neuron classifier. The architecture of the Basic CNN can be seen in Fig 3.

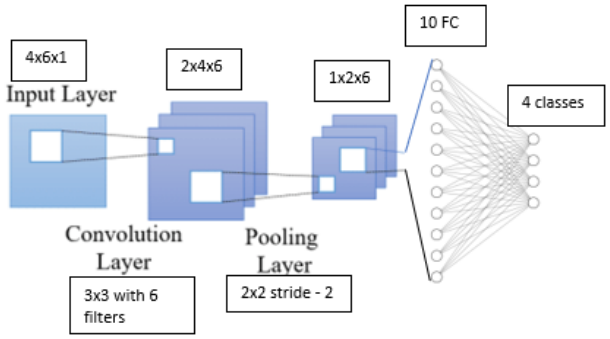

Fig. 3. Simple CNN Architecture

B. Hyperparameters

The model is being trained for 50 epochs, with a batch size of 16. The weighted cross entropy loss function is being used as the cost function. The weights are being computed with the number of samples of each classes from the training set. An Adam optimizer is being used for the optimization with an initial learning rate of 10e-4.The stratify parameter is used to ensure that the class distribution of the input data is maintained in the validation sets.

III. EVALUATION

A. Loss and Accuracy

The ANN model achieved a train accuracy of 83.11% and validation accuracy of 81.36%. On the other hand, the CNN model achieved a train accuracy of 75.30% and validation accuracy of 75.29%. The above graphs show the variation of the model loss and accuracy with number of epochs. The graph of train and test accuracy with respect to epochs is being shown in Fig 4. The graph of train and test loss with respect to epochs is being shown in Fig 5.

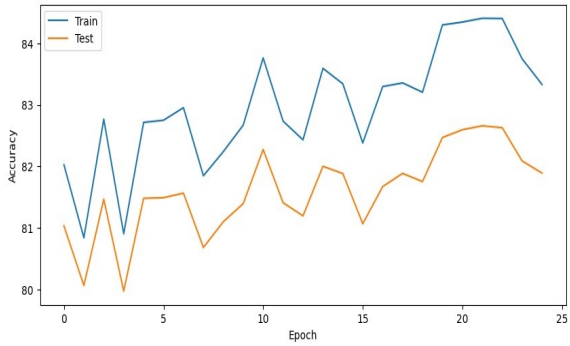

Fig. 4. Accuracy Vs Epochs

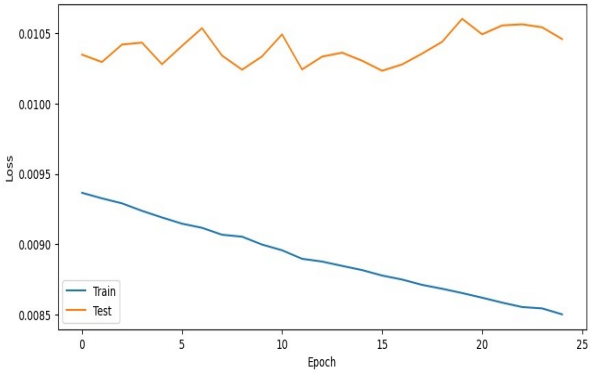

Fig. 5. Loss Vs Epochs

B. Confusion Matrix

The confusion matrix is a summary of classification results. The number of correct and incorrect predictions are summarized with count values and are broken down for each classes. The confusion matrix for the train set is shown in Fig 6. and the confusion matrix for the validation set is shown in Fig 7.

The model was assessed by examining the subsequent metrics such as accuracy, precision, recall, F1- score, and mean absolute error on training and validation sets.The test data (i.e subject-09 to subject-12) was tested for predictions using the ANN model.

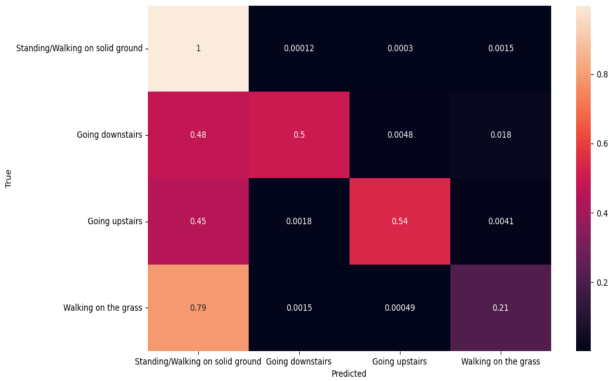

Fig. 6. Train Confusion Matrix

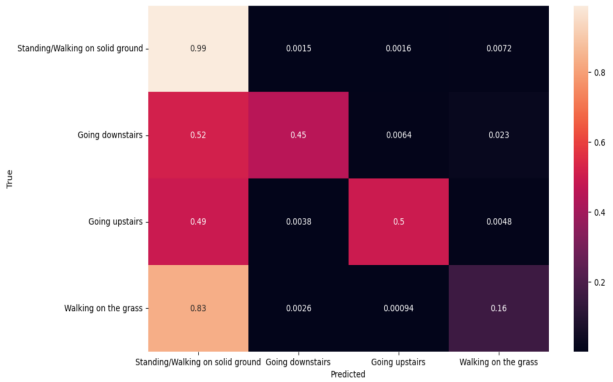

Fig. 7. Validation Confusion Matrix

REFERENCES

[1] B. Zhong, R. L. da Silva, M. Li, H. Huang, and E. Lobaton, "Lower limb prostheses environmental context dataset," 2020. [Online]. Available: https://dx.doi.org/10.21227/d3z5-n231

[2] Dixon, Philippe Schutte, K.H. Vanwanseele, Benedicte Jacobs, Jesse Dennerlein, Jack Schiffman, Jeffrey Fournier, P-A Hu, Boyi. (2019). Machine learning algorithms can classify outdoor terrain types during running using accelerometry data. Gait Posture. 74. 10.1016/j.gaitpost.2019.09.005.

[3] Kelly Shen, Michael Kelly, Simon Le Cleac'h, Terrain Classification for Off-Road Driving. Stanford University, 2017.