Classifying and Segmenting Behavioral Events Using Daily Devices

Group 32

Yen Chang
yec084@ucsd.edu

Xin-ling Bai
xib009@ucsd.edu

Jui-Ting Tsai
jtsai@ucsd.edu

Abstract

In this project, we analyze Extrasensory dataset consisting of data from mobile phone and watch sensors. We trained a neural network showing decent results. Also, we showed that using boosting method improves the result by ensembling multiple neural networks with different input features. Finally, we use LSTM to utilize the dependency of time series data and showing competitive result in the 7-class classification problem.

***Keywords*—** LSTM, RNN, Boosting, In-The-wild Data

1 Introduction

Behavioral Context Recognition has gained more and more importance in bio-medical field such as patient tracking and health status monitoring. Despite the precision, power or system improvement making sensor data more accessible nowadays, collecting data while not affecting their natural behaviors still remain a huge challenge. Therefore, we plan to analyze sensor data from mobile phones and watches and try to classify 7 main behavior with 1 exception class. We continue the study of Extrasensory Dataset [5] in time series modeling to help segment time into meaningful behavioral events.

2 Related Work

Vaizman et al [6] proposed using everyday devices such as smart phones and smart watches to incorporate diverse sensors. They have applied the data that incorporates various sensors, such as accelerometer, gyroscope, and magnetometer, to automatically recognize behavioral context. However, there are some features remains unused. We want to make use of the advantage of neural network and use all features for our training. Moreover, many tasks are still opened as stated on the Extrasensory Dataset web page [5] and one of which is time-series modeling. There are many previous work on time series segmentation [3, 1]. In previous studies, the researchers utilized many algorithms ,including GIR (Global Iterative Replacement) or TCN (Temporal Convolutional Network) to solve the problem. We are considering LSTM to predict activity given previous labels in time series modeling.

3 Dataset and Features

We will make use of "The ExtraSensory Dataset" [5] which contains large scale, over 300k examples(minutes) from 60 users, for our model training. The dataset contain both mobile phone and watch data using a customized app [7] with a wide range of sensors, enabling a variety of features for our training. The dataset is a in-the-wild dataset which is recorded in the daily life. Therefore, the raw data is quite noisy and the sampling rate of different sensors does not match one another creating another issue for our training. Hopefully, Vaizman et al provide a statistical method to pre-process the data, using a variety of statistical parameters such as mean, standard deviation on a minute-based cycle, allowing us to solve the aforementioned problem.

There are 7 different main activities in the dataset: lying down, sitting, standing in place, standing and moving, walking, running, bicycling. Still, there are several missing labels. One cause may be the experimental issue, which the user fail to report their activity. Another possible cause may be an exception to the existing 7 main classes. After inspecting data, we found that the missing labels are quite continuous. Therefore, we try to solve this issue by creating an extra class to accommodate these missing labels as the 8th class. One advantage of this solution relates to the in-the-wild classification. We may encounter situation that does not match our existing classes, and training those exception situation as a new class makes the data of our existing class more consistent and robust.

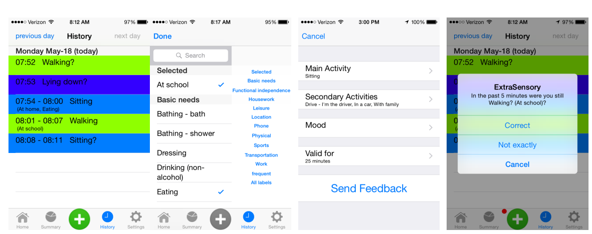

Figure 1: App screenshot for data collection

4 Methods

Neural network came to out first because we are faced with a multi-class classification problem. Then we tried to ensemble multiple models trained by different features to improve the accuracy. Lastly, we add LSTM layer to our neural network to make use of the time dependency.

4.1 Training Framework

We divide the dataset into training, validation and testing set, which contains 48, 6 and 6 users each. In addition, we perform Min-Max scaling before feeding our data into the neural network.
We use the model adjustment framework described in Figure 2 to tune our models. In the first step, we select the parameters such as number of layers, number of hidden nodes. Then we train our model and compute loss according to the validation set to tune our model. Finally, we keep the best result by evaluating the accuracy on the testing set.

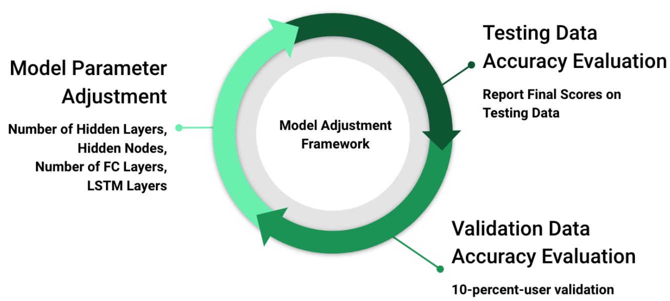

Figure 2: Neural network model adjustment framework.

4.2 Neural Network

There are 11 main groups of features in the dataset, such as gyroscope, location, audio properties, etc.... Therefore, we try to train models based on these main groups, yielding decent result. We also train a model with all the feature together to compare the result with the individual ones.
For the hyper parameter, we use a 2 layer network with batch normalization. Since we are dealing with a multi-class classification problem, we use softmax instead of sigmoid function for our activation function. Also, we found out that shuffling the input helps a lot because there are often consecutive labels in our dataset. Training without shuffling will easily fit to a specific class and yield unsatisfactory result.
After training the network, we found our network overfits to training data pretty much. Therefore, we add dropout layer introduced by Nitish Srivastava et al. [4].And help raise the testing accuracy by over 10 percent.

4.3 Ensemble Method

After training neural network based on individual groups of features, we try to ensemble all 11 groups with a regression layer connected to the output of the separate models, and the structure is shown is Figure 3.

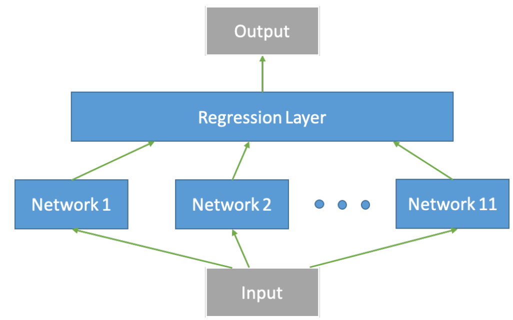

Figure 3: Ensemble boosting method.

4.4 Long Short-Term Memory

Long short term memory (LSTM) [2] have the advantage of remembering state information in the previous data and forgetting information in certain period. By training on large scale data, the model may be able to perform consistence prediction allowing us to segment the behavior according to the output easily.

In the experiment, we attach the LSTM with several layers of fully connected network. The hidden vector (output of LSTM) has 500 dimensions.

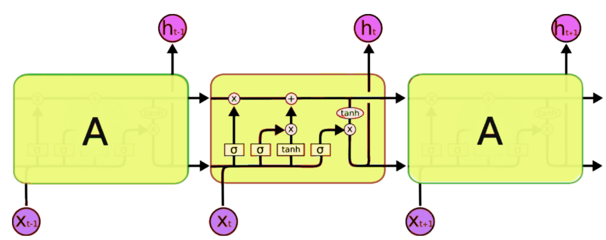

Figure 4: Long short Term Memory

5 Results

5.1 Neural Network

Figure 5 shows the accuracy through epoch while training neural network with all features. We can see that the training accuracy continues to rise smoothly while the validation accuracy fluctuates a bit. The final result is shown in Figure 6. The testing result is surprisingly good at around 0.45 accuracy.
The validation accuracy start out to be larger than that of training, which is not usual. It may arise from fitting to the validation set accidentally in the first few batches.

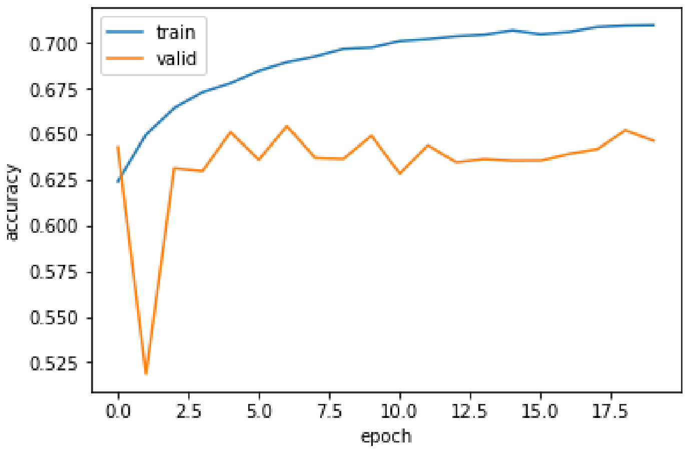

Figure 5: Neural network training accuracy versus epochs.

	Training	Validation	Testing
Accuracy	0.742	0.647	0.450

Figure 6: Neural network result (Training with all features).

5.2 Ensemble Method

The training accuracy of the final ensemble outcome is 0.707. The testing accuracy of individual feature groups and final ensemble model is shown in Figure 7. We can see that the ensemble outcome is better than every feature groups slightly better than a single neural network training with all features. Overall, motion-related feature groups have better accuracy than others, which is reasonable because the class is mostly defined based on body movements. Also, few feature groups yields surprisingly good result such as "discrete" and "audio naive". We may try to train our model with these two features alone in the future.

Feature Group	Testing Accuracy	Feature Group	Testing Accuracy
raw_acc	0.318	location	0.327
proc_gyro	0.336	location_quick_features	0.266
raw_magnet	0.234	audio_naive	0.405
watch_acceleration	0.338	audio_properties	0.334
watch_heading	0.272	discrete	0.446
lf_measurements	0.342	All(ensemble)	0.454

Figure 7: Ensemble Model Result (Compared with individual results from a single feature group).

5.3 LSTM

For LSTM, we first reformed the data into time sequence with 3 time delay. That is every data point contains not only the data in the current timestamp but also the all data within 3 minutes. By feeding in the data sequentially, the network is able to produce cell state and hidden state base on previous information then classify the activity according to it.

However, the sequential data have taken more space then usual and prolong the training time. Therefore, we only train the LSTM up to 10 epochs.

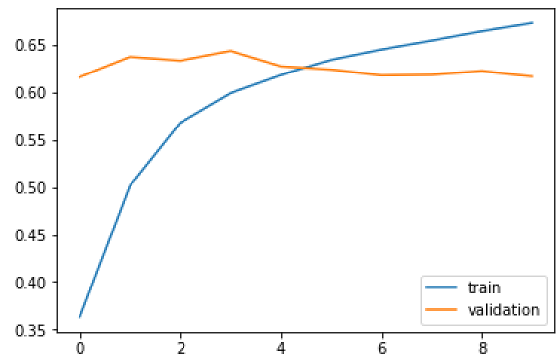

Figure 8: LSTM training accuracy versus epochs

5.4 Model Comparison

Table 1 show the comparison among all the models we used.

	Training Accuracy	Validation Accuracy	Testing Accuracy
Neural Network	0.742	0.647	0.450
Boosting	0.707		0.454
LSTM	0.608	0.615	0.482

Table 1: Model Result

5.5 Confusion Matrix

By generating the confusion matrix, we are able to see which labels are more similar and misunderstanding by the model.

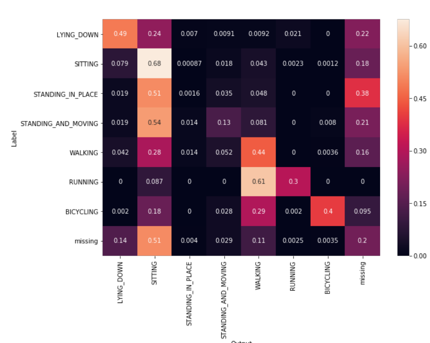

Figure 9: Confusion Matrix

6 Discussion

6.1 Missing Label

According to the explanation of dataset, the main label should be mutually exclusive. In reality, the user can still provide data without setting main activity. As a result, the label may be missing due to the user forget to select main activity or the user is not doing any of the 7 main labels.

We tried to impute the missing label by creating an extra label. However, the missing label may be either situation and the model cannot identify it correctly. This is the reason why missing label in figure 9 have a high level of error.

6.2 Different User Behavior

The users in dataset have very different behavior. Some people are vary active and they want to provide the data as much as they they can. As long as they change the behavior, they report the labels immediately. Although they make a great contribution to the dataset, they might also affect the authenticity of their behavior, since people usually won't interact with the specific application that much in just few minutes. On the other hand, others may provide the data which is a smaller amount and less detailed but more authentic. This situation may cause the training focus on the particular users and jeopardize the performance.

6.3 Data Sample Rate

The amount of each label is somehow unbalance. There are 136356 examples from 60 users belong to label of sitting, but only 83055 examples from 53 users of sleeping. It is harder to predict the labels have less examples than the labels contains more data. When we train the smaller labels, which don't supply enough information to represent the exact pattern behind the curtain or only includes data from specific users, the model may not be able to deal with the data of testing group from different users or subset, since those training labels are not generative sufficiently. Therefore, we can utilize data augment to balance the whole dataset and improve the results.

6.4 Environment Noise

The environment noise would also affect the results of segmentation. The data that is collected from the same user may be fluctuated just due to the influence of environment. These environment noise makes the data is less similar to each other, which may confuse the network and harness the accuracy. To enhance the training and obtain the better result, it is necessary to fix the effect of environment noisy. There are some approaches may be capable of resolving the noisy issue; we can perform the feature extracting methods, such us LDA or PCA, or we can divide the dataset into subsets by which kinds of the environment is when user collect data, in order to lessen impact of the noise as much as we can.

7 Future Work

7.1 Data Preprocessing

Currently our data is minute-to-minute data which consist of many statistical parameters. However, this may lead to some loss of information. Therefore, we may make use of raw data and filtering techniques on the selected features, hoping to yield better result.

7.2 Ensemble Models

For this project, we ensemble neural networks with different input features. We may try to ensemble LSTM model together with LDA or other models as well.

7.3 LSTM

Due to the lack of memory resource and training time, currently we only make use of three consecutive data as a group, meaning that we only take advantage of 3-minute dependency. Though already show performance improvement. We think that for larger data groups and training time, we may come up with better model.

8 Contribution

Jui-Ting Tsai

- Training Neural Network
- Training LSTM Network
- Composing report and poster

Xin-ling Bai

- Data Visualization
- Tuning Parameter
- Composing report and poster

Yen Chang

- Data Preprocessing
- Training Neural Network
- Training LSTM Network
- Composing report and poster

Repository

https://github.com/jason281/ECE228_Group32

References

[1] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmaki, and H. T. Toivonen. Time series segmentation for context recognition in mobile devices. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 203–210. IEEE, 2001.

[2] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[3] C. Lea, R. Vidal, A. Reiter, and G. D. Hager. Temporal convolutional networks: A unified approach to action segmentation. In *European Conference on Computer Vision*, pages 47–54. Springer, 2016.

[4] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[5] Y. Vaizman, K. Ellis, and G. Lanckriet. The ExtraSensory Dataset. http://extrasensory.ucsd.edu/.

[6] Y. Vaizman, K. Ellis, and G. Lanckriet. Recognizing detailed human context in the wild from smartphones and smartwatches. *IEEE Pervasive Computing*, 16(4):62–74, October 2017.

[7] Y. Vaizman, K. Ellis, G. Lanckriet, and N. Weibel. Extrasensory app: Data collection in-the-wild with rich user interface to self-report behavior. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 554:1–554:12, New York, NY, USA, 2018. ACM.