Deep Learning to Detect Heavy Drinking Episodes Using Smartphone Accelerometer Data

Bo Yang
Department of Computer Science
Stanford University
hiyangbo@stanford.edu

Abstract

Harmful use of alcohol is responsible for 5.1% of the global burden of disease. Recent work aimed at promoting healthier drinking habits has shown promise for the effectiveness of just-in-time adaptive interventions delivered on mobile phones just before the onset of heavy drinking episodes. This project used only non-sensitive accelerometer data collected from mobile phones, examined different deep learning architectures and developed a reliable classifier which detected periods of heavy drinking with 96.2% accuracy.

1 Introduction

Harmful use of alcohol is responsible for 5.1% of the global burden of disease. [1]. Thus, social workers have studied how to reduce heavy drinking habits through interventions such as education programs and motivational feedback, and social media campaigns, etc.
With smartphones becoming so popular in today's society, nearly everyone owns one because of the convenience and the wide range of functions they offer. Thus researchers have begun to investigate the effectiveness of mobile interventions.
However, a recent study which delivered hourly mobile interventions to participants during drinking events showed no significant reduction in the amount of alcohol consumed [2], suggesting that overly frequent messaging can reduce the effectiveness of interventions. This highlights the need for accurate, targeted messages to participants during drinking episodes.[3]

Raw accelerometer data are not sensitive and thus will be much easier to be adopted compared to sensitive location, calls, keystrokes which may raise privacy concerns.

For this project I explored 4 state of the art deep learning architectures, including a baseline neuron network, a Convolutional Neural Network (CNN), a Long Short Term Memory (LSTM) and a CNN-LSTM, to detect the heavy drinking episode using mobile accelerometer data. I also examined various hyperparameters and analyzed impacts on model performance. The best model detected heavy drinking with 96.2% accuracy.

2 Related Work

This project used the open source data: Bar Crawl: Detecting Heavy Drinking Data Set in UCI Machine Learning Repository.[6] The dataset was first used by Jackson A Killian et al in their paper Learning to Detect Heavy Drinking Episodes Using Smartphone Accelerometer Data. [3] It extracted features from both the time domain as well as doing Fast Fourier Transform to get the frequency domain features such as spectral_centroid, spectral_spread, spectral_rolloff, avg_power, etc. Then these features were fed to train 4 different classifiers, Multilayer Perceptron Network (MLP) , an SVM with a radial basis function, using LIBSVM, a random forest, and a convolutional neural network (CNN). Random Forest wins among the classifiers with a 77.5% accuracy. However, deep learning networks were yet explored meaningfully in the paper.

Matteo Gadaleta et al use CNN on accelerometer and gyroscope (inertial) signals in their paper IDNet: Smartphone-based Gait Recognition with Convolutional Neural Networks[4]. After a bunch of feature engineering like transforming the accelerometer and gyroscope to orientation independent reference, CNN was then used for automatic feature extraction followed by a SVM layer for final classification.

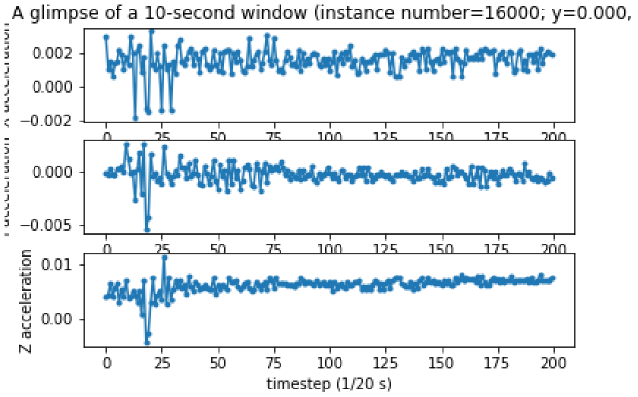
Ming Zeng, et al in their 2014 paper “Convolutional Neural Networks for Human Activity Recognition using Mobile Sensors[5]. use CNN model for accelerometer data, where each axis of the accelerometer data is fed into separate convolutional layers, pooling layers, then concatenated before being interpreted by hidden fully connected layers.

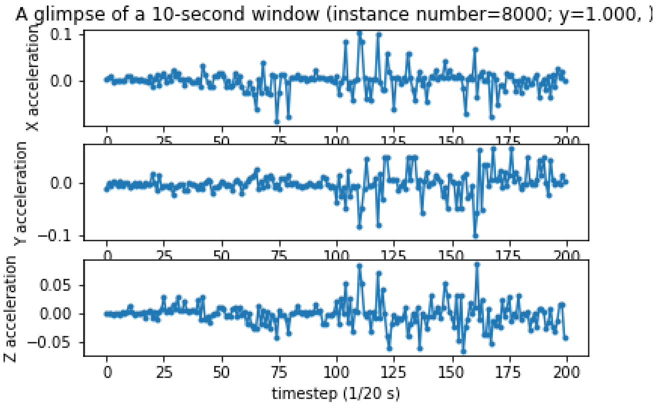
3 Dataset and Features

I used the open source data: Bar Crawl: Detecting Heavy Drinking Data Set in UCI Machine Learning Repository.[6] The data has 14057567 instances, collected from 13 participants. Accelerometer data were collected from a mix of 11 iPhones and 2 Android phones, including 5 columns: a timestamp, a participant ID, and a sample from each axis of the accelerometer. TAC data was collected using SCRAM ankle bracelets and was collected at 30 minute intervals.

There are two parts in the raw dataset, TAC reading, from which the label is extracted from and the accelerometer data, which has four columns, time stamps, x, y and z accelerometer readings.
For TAC reading, the label is extracted as 1 when the reading is equal or larger than the preset threshold (indicating a heavy drinking episode) and 0 otherwise. I use 0.08 as the threshold.
For the accelerometer readings, instead of doing complex feature engineering to extract frequency domain features, I used merely the plain accelerometer data in the x,y,z axis. Its timestamp is recorded at millisecond while each second may have a various number of readings. So to make all seconds have the same number of instances, 20 readings were sampled from each second.   e.g. readings at second T are [x_T, y_T, z_T] a 20 X 3 matrix.
On the other hand, whether there's heavy drinking at second T, is highly correlated to those most recent accelerometer readings, so I created an overlapping sliding window view for every 10 seconds. More specifically, the total features for second T are
[x_{T-9}, y_{T-9}, z_{T-9}
x_{T-8}, y_{T-8}, z_{T-8}
...
x_T, y_T, z_T], a 200 X 3 matrix.
The dimension of the final input is 30726 X 200 X 3.   Note that y=1 accounts for 63.5% of the total population, which serves as the random guess accuracy.

Below is a glimpse of two samples of the 200X3 features with y being 1 and 0 respectively.



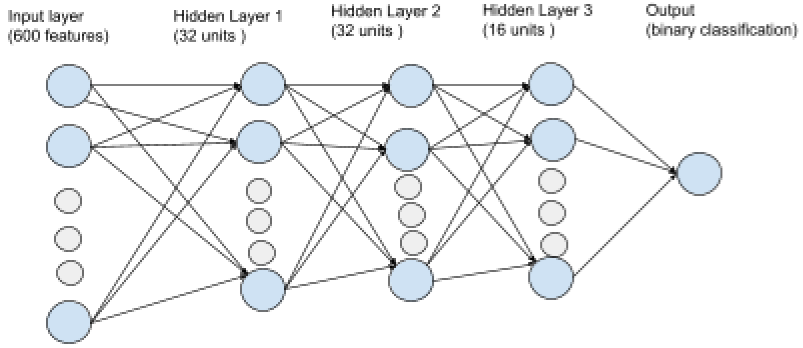


4 Method

I explored 4 different deep learning architectures, including a baseline neuron network, a Convolutional Neural Network (CNN), a Long Short Term Memory (LSTM) Recurrent Neural Network (RNN) and a CNN-LSTM to solve the problem.

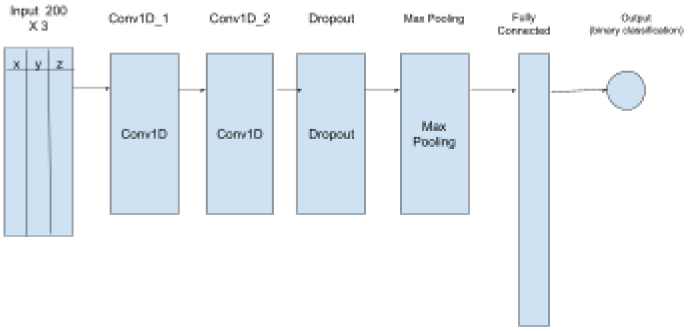
Baseline Neuron Network:

200X3 inputs were flattened before feeding into three hidden layers, with 32,32,16 hidden units respectively. i.e.



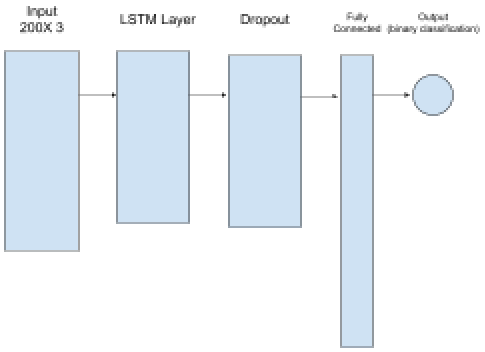
Convolutional Neural Network:

200X3 inputs were fed into two consecutive Conv1D layers, followed by a dropout layer, a max pooling layer as well as a fully connected layer before the final output.



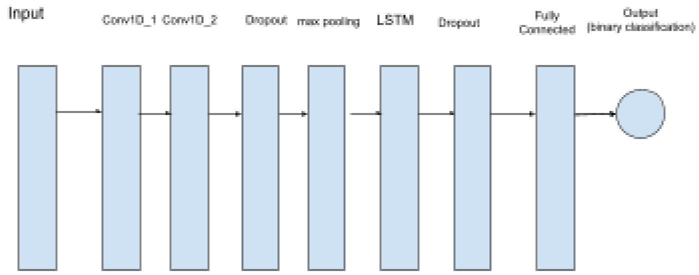
Long Short Term Memory

200X3 inputs were fed into a LSTM layer, followed by a dropout layer and a fully connected layer before the final output



CNN-LSTM

200X3 inputs were reshaped to fit in time distributed and then fed into two consecutive Conv1D layers, followed by a dropout layer, a max pooling layer, the output of which is then fed to the following LSTM layer, followed by a dropout layer and fully connected layer before the final output.



5 Experiments and Results

For the baseline model, I used three hidden layers, with 32 nodes, RELU, 32 nodes, RELU and 16 nodes RELU respectively. The output layer uses sigmoid activation. I use the binary cross entropy as the loss function and Adam as the optimizer.
With epoch=50 and batch_size=32, it gives a model with accuracy of 66% on the test data set, which is similar to the random guess accuracy as y=1 accounts for 63.5% of the population. The training accuracy was above 80%, which indicates an obvious high bias. Thus we need a more advanced network to extract the collelations from the x,y,z axis..

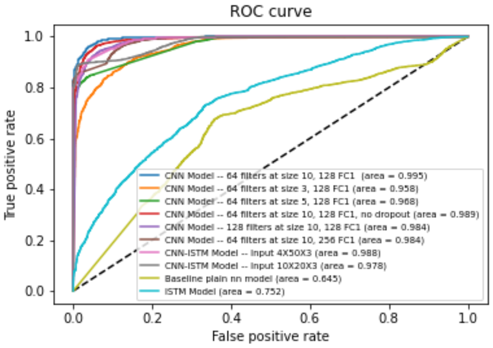
For the CNN models, I started with two consecutive Conv1D layers, each with 64 filters and filter size 3, followed by a dropout layer (p=0.5) and max pooling (pool_size=2), and then a fully connected layer with 128 features before the final output. It got an accurary 87.5% on the test set. Tuning the filter size to 10 further pushed the accuracy to 96.2% which is the best model for this project. Then I removed the dropout layer, the training accuracy increased a bit while the accuracy on the test set dropped from 96.2% to 94.9% which indicated a slight overfitting without dropout and thus the regulation added by the dropout layer is very helpful in this case. I also tried increasing filter size from 64 to 128 as well as increasing hidden units number from 128 to 256 for the last fully connected layers, both of which showed signs of sligh overfitting and failed to improve the model performance.

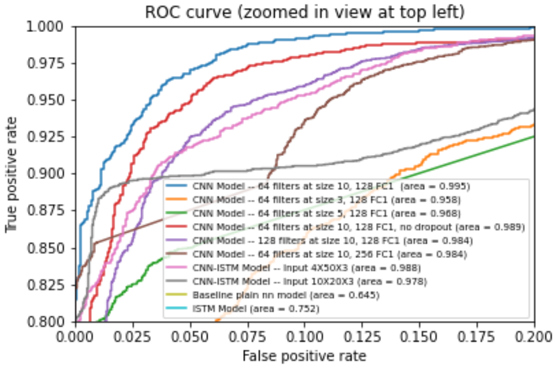
For the LSTM model, I used an LSTM layer with 128 units, as well as a dropout layer (p=0.5) and a fully connected layer with 128 units . It got an accuracy of 70.7% which is much worse than the CNN model. I think this is because the prediction task is to predict drinking episodes and thus the accelerometers in the past 10 seconds all weigh equally, i.e. any inharmonious pattern in the 10 second window may indicate a heavy drinking episode. Therefore CNN does a better job than LSTM on extracting efficient features, the latter of which weigh the last row more than the early rows and therefore may ignore signals in early seconds.

Forr CNN+LSTM, I combined the previous CNN layers and LSTM layers by encapsulating the CNN layers with TimeDistributed. This approach basically used CNN to extract features (the very last max pooling layer) which then served as the input to the following LSTM. When the input was reshaped to 4X50X3, it got an accuracy of 93.1%, while the accuracy dropped to 92.3% with a 10X20X3 input. The result was acceptable but worse than the best

CNN model which had a 96.2% accuracy. It tells that the 128 features extracted from CNN from the sliding window have very accurate characteristics of the drinking episode and adding a LSTM do not add much value there.

Below is the ROC curve graph of all the above models. It shows that the above best CNN model stands out. i.e. the best model is the one used two consecutive Conv1D layers, each with 64 filters and filter size 10, followed by a dropout layer (p=0.5) and max pooling (pool_size=2), and then a fully connected layer with 128 features before the final output





6 Conclusion

I explored 4 deep learning architectures,including a baseline neuron network, a Convolutional Neural Network (CNN), a Long Short Term Memory (LSTM) and a CNN-LSTM, and examined with different hyperparameters, among which CNN got the best result of 96.2% accuracy. The best model used 2 Conv1D layers with 64 filters of size 10 followed by a 2x2 max pooling layer and a 128 features fully connected layer and a final binary output layer.

The best CNN model's accurary 96.2% beats the 77.5% in the original paper which uses random forest. Unlike the previous related works with complex feature engineering to extract features in both time and frequency domain, I used original accelerometer data.

I discussed that the reason that CNN stands out among the four architure is because the prediction task (detect heavy drinking episode) is highly related to any inharmonious pattern detected in a 10second clip of accelerometer data. CNN does the good job of extracting features from the x,y,z correlations with equal weight in that sliding window. LSTM on the other hand weighs more on the last input and thus may overlook the signals in early time in the 10 second clip.

With such great accuracy, it's very promising that we can put the model in real use to help people living a safer and healthier life.

Moreover, the experiments and comparisons done in this project can also be helpful on similar binary classification. i.e. CNN should do well on classifications on inharmonious patterns using pure accelerometer data. E.g. to alert DUI drivers, etc.

7 Contributions

This is my solo project but lots of thanks to my TA Fenglu and Avoy for their advice and pointers to try Conv1D on pure accelerometer data with minimum feature engineering.

Reference

[1] WHO Global status report on alcohol and health_2018
https://apps.who.int/iris/bitstream/handle/10665/274603/9789241565639-eng.pdf?ua=1
[2] Cassandra Wright, Paul M Dietze, Paul A Agius, Emmanuel Kuntsche, Michael Livingston, Oliver C Black, Robin Room, Margaret Hellard, and Megan SC Lim. Mobile phone-based ecological momentary intervention to reduce young adults' alcohol use in the event: A three-armed randomized controlled trial. JMIR mHealth and uHealth, 6(7):e149, 2018.
[3] Killian, J.A., Passino, K.M., Nandi, A., Madden, D.R. and Clapp, J., Learning to Detect Heavy Drinking Episodes Using Smartphone Accelerometer Data. In Proceedings of the 4th International Workshop on Knowledge Discovery in Healthcare Data co-located with the 28th International Joint Conference on Artificial Intelligence (IJCAI 2019) (pp. 35-42).
[4] Matteo Gadaleta∗ , Michele Rossi, IDNet: Smartphone-based Gait Recognition with Convolutional Neural Networks
[5] Ming Zeng; Le T. Nguyen; Bo Yu; Ole J. Mengshoel; Jiang Zhu; Pang Wu; Joy Zhang Convolutional Neural Networks for human activity recognition using mobile sensors
[6] https://archive.ics.uci.edu/ml/datasets/Bar+Crawl%3A+Detecting+Heavy+Drinking