Extracting kinetic features from wearable tech for clinical symptoms of Parkinsons Disease

Alexandra Bourdillon[1], **Kartik Sawhney**[1], **Rishab Mehra**[1], **Patrick O'Grady**[1], **Thomas Liu**[1]

[1] School of Engineering – Stanford University, Stanford, CA

{abourdil, kartiks2, rishab, pogrady, tliu623} @ stanford.edu

Abstract. *The diagnostic application of sensor enabled wearable technologies holds great potential for the clinical management of Parkinson's Disease, a progressive disorder of the nervous system characterized by severe degradation of movement and uncontrolled tremor. Here, we present a deep learning approach to extract digital biomarkers of accelerometer data for predicting clinically-relevant metrics of tremor. In this study we compare and contrast a convolutional neural network approach to long-short-term memory recurrent neural network approaches to offer insights about digital features and propose areas of future study.*

1. Introduction

The application of sensor enabled wearable technologies for enhancing clinical understanding and treatment of motor-affecting conditions holds great potential for widespread medical impact. Parkinson's Disease (PD) is a progressive disorder of the nervous system characterized by severe degradation of movement and disruptive uncontrolled trembling [Jankovic 2008]. Accelerometer data allowed by advanced sensor-based technologies offer continuous and remote assessments and insights that supplement health-professionals' evaluations for diagnosis, prognosis and disease management [S. Patel and Rodgers 2012, S. Del Din and Rochester 2016]. Deep learning offers promising tools for pattern recognition of salient clinical features of PD for automated assessment [I. M. Pires and Flórez-Revuelta 2017]. Our goal is to capitalize on such techniques for feature extraction to identify key objective digital traits from accelerometer sensor data to identify optimal predictors of symptoms of PD. We hope this work will help improve longitudinal tracking of impairments and delivery or personalized treatment, both outlined as great needs for the advancement of PD therapy by the Movement Disorder Society Task Force [A. J. Espay 2016].

Deep Learning signifies one of the key advances of machine learning over the past decade [N. Hammerla 2016]. Its diversity of application and steady development has led to impressive new streamlined pipelines for high-dimensional, non-linear modeling by way of multi-layer interconnected computational nodes, for example through deep feed-forward neural networks (DNNs). To better model translational invariance and temporal dependencies within movement, convolutional neural networks (CNNs) and Long-short-term memory recurrent neural networks (LSTMs), respectively, offer alternative architectures, which have demonstrated impressive utility and accuracy in modeling movement [N. Hammerla 2016, Gamboa 2017, P. Malhotra and Agarwal 2015] as well as a robust framework for feature extraction [Ordonez and Roggen 2016]. Futhermore, the motivation for using CNNs and LSTMs is to produce an effective model for training on the

nature of basic continuous movements as well as the salience of the combination of basic movements [Y. B. Yang and Krishnaswamy 2015].

Such architectures have been applied to pattern recognition of human activities and movements with variety in scope and functionality [Ordonez and Roggen 2016, Y. B. Yang and Krishnaswamy 2015, I. M. Pires and Flórez-Revuelta 2017]. For the purposes of supporting treatment in PD, we are particularly interested in identifying a digital biomarker for estimating the severity of tremor. Tremor, or involuntary shaking movement is a characteristic symptom of PD, which, as it progress, poses increasingly debilitating effects on patients. By defining objective digital biomarkers with convenient reproducibility, clinicians can reliably and accurately track critical milestones in disease progression or symptom reduction and provide additional metrics for evaluating the efficacy of interventions [G. Ebersbach and Deuschl 2006].

Accelerometer data collected with a wearable wristwatch technology on PD patients undertaking various upper-body tasks made available by the Parkinson's Disease Digital Biomarker DREAM challenge allows us to test a number of experiments exploiting internally-developed CNN and LSTM architectures for the purposes of extracting key digital features and enhancing intuition for future progress in this area of research.

2. Related Work

The rise of real-time and robust wearable technologies embedded with motion-tracking sensors have given way to substantial efforts in efficient and computationally complex approaches to pattern recognition. Implementations and utility vary widely but results appear to converge on CNN and LSTM architectures that demonstrate impressive reliability and accuracy [I. M. Pires and Flórez-Revuelta 2017, Ordonez and Roggen 2016, N. Hammerla 2016].

A number of efforts have been directed towards processing data acquired through mobile devices, especially for the detection of Activities of Daily Living (ADL), including tasks such as standing, walking, running, ascending stairs and descending stairs. Accuracy ranges between 74% and 88%, and as high as 98.69% have been reported using decision tree methodologies on various momentum statistics such as mean, median, variance, standard deviation, maximum, minimum, range, RMS and FFT coefficients [Fan, Ling]. Other groups demonstrated effective pipelines drawing from accelerometer data skewness, kurtosis and the slope for each axis [I. M. Pires and Flórez-Revuelta 2017] . Pires and colleagues reported 85.89% accuracy when training a deep learning neural network on various statistical metrics about the maximum peaks in combination with similar metrics of the raw signal - including 5 greatest distances between maximum peaks, the average, the standard deviation, the variance, the maximum, the minimum and the median [I. M. Pires and Flórez-Revuelta 2017].

Hammerla and colleagues published extensive work comparing and contrasting deep, convolutional and recurrent approaches to human activity recognition (HAR) using wearable sensor technology [N. Hammerla 2016]. They found that LSTM-powered recurrent neural networks (RNNs) outperformed both deep and convolutional approaches, in part due to the sophisticated modeling of temporal dependencies on the movement data, although CNNs performed well for recognizing repetitive tasks such as walking and running. The researchers reported that among bi-directional RNNs, the number of units per

layer produced the largest effect on performance. Further statistical analysis of that hyper-parameters revealed that DNNs incurred the widest spread in performance compared to CNNs and RNNs, indicating that DNNs may require more iterations and tuning to achieve similar performance levels [N. Hammerla 2016].

Yang and colleagues have published developments on hand gesture-specific HAR trained on multichannel time series signals acquired from on-body sensors (typically a three-axis accelerometer and a two-axis gyroscope at a sampling rate of 32 samples per second). Their research examined pattern recognition for a number of common upper-limb behaviors: inactivity, opening a window, closing a window, watering a plant, turning a page, drinking from a bottle, cutting with a knife, chopping with a knife and stirring in a bowl. Their internally developed CNN outperformed four baseline models using various methodologies including: support vector machine (SVM), kth nearest neighbor (KNN), means and variance (MV) and deep belief network (DBN). On hand-specific gestures, the CNN achieved accuracy between 94.1% and 96.0% [Y. B. Yang and Krishnaswamy 2015]. Their work affirms that CNNs are promising models for perceiving salient patterns of signals, and not simply their positions or scales.

The combination of CNNs and LSTMs for the purposes of HAR has proved promising as well. Ordonez and colleagues introduced a new framework for wearable activity recognition called "DeepConvLSTM," an eight-layer network, which combines convolutional and recurrent layers. The CNN operates via a sliding window approach which examines several sensor channels over time. Their convolutional layers act as feature extractors and are able to abstract away prominent digital patterns of the input sensor data. Compared to a non-recurrent deep CNN architecture with four convolutional layers and three dense layers (mirroring that of DeepConvLSTM), DeepConvLSTM achieved higher accuracy levels of 1.7% for locomotion activities and 3.2% for hand gestures [Ordonez and Roggen 2016].

In the application of computational analyses of wearable technology data to Parkinson's Disease, Silva de Lima and colleagues conducted a systematic review on this intersection with respect to fall risk assessment. They evaluated over 27 peer-reviewed articles, 23 of which focused on freezing of gait (FOG) and four specifically on falls. Studies varied in setting and sensor location, but most relied on tri-axial accelerometer data. For FOG, the article reported impressive sensitivity, ranging from 73% to 100%, and specificity, ranging from 67% to 100%, although the reviewers pointed to the need for computational optimization to allow for feasible portable devices and efficient algorithms for producing clinically-relevant and substantial results [A. Silva de Lima and Faber 2017].

3. Data & Pre-Processing

3.1. Dataset

The dataset was made available through the DREAM Challenge and hosted through Synapse (ID#: syn8717496). Motion data (through a GeneActiv wristwatch and a Pebble smartwatch) and accelerometer on patients while undertaking various upper-body tasks.

The data was collected over two collaborating research institutions on 19 patients over 2 separate clinic visits in which a series of 12 tasks were repeated over 6-8 to cycles (sessions). These tasks include: finger-to-nose for 15s (twice with each arm), alternating

hand movements for 15s (twice with each arm), opening a bottle and pouring water (three times), arranging sheets of paper in a folder (twice), assembling nuts and bolts for 30s, folding a towel three times. Each patient wore a GeneActiv wristwatch on the more symptomatic hand and a Pebble smartwatch on the other side.

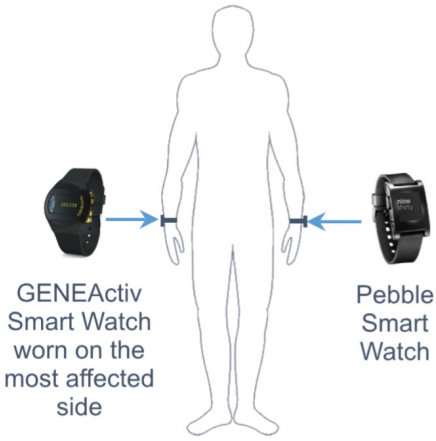

Figure 1. The dataset contains accelerometer data from two types of smartwatches. Graphic courtesy of the DREAM CHALLENGE.

The data can be accessed through a large table hosted by Synapse in which metadata about each subject's task recording is offered (patient ID, session #, visit #, task, device, device side and tremor score) as well as accelerometer data from the worn device itself (in the form of a tsv file with a timestamp, acceleration in x, acceleration in y, acceleration in z, and the magnitude of the acceleration).

Data analysis is typically a good first step in any data-driven project. In this project, this means understanding the timeseries motion data from a statistical perspective. As it is complicated to visualize acceleration in 3 dimensions over time, our initial modeling revolved around the magnitude of acceleration at each time step. We sought to understand if there are significant statistical differences between high and low tremor score timeseries across the different activities mentioned previously. To mitigate the effect of data anomalies on our visualization, we removed nan values from the timeseries and removed outliers in magnitude which we, for the time being, considered to be any datapoint with a magnitude of acceleration over 40.

First, we consider the drinking activity and how it varied between the GeneActive wristwatch and Pebble smartwatch.

As seen in the box plots, higher tremor scores in this activity coincide with a higher mean magnitude acceleration, which provides a simple sanity check that our formulation of the data, even a simple form, provides interesting insights. This loose correlation between tremor scores is much more explicit with the GeneActiv device than the Pebble device, indicating that a Parkinson's patient can experience extremely different symptoms between the more and less afflicted sides.

While the box plots for this activity make the data simple to understand, the data from other activities does not provide the same clarity of thought. In the following figures, the mean acceleration magnitude actually decreases with a higher tremor score, reversing

the correlation depicted by figure 1 and figure 2.

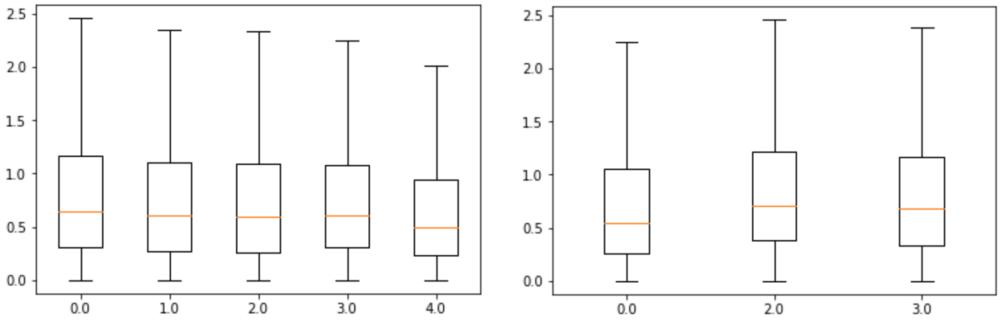

Figure 2. The first box plot is of the magnitude of acceleration data captured using a GeneActiv wristwatch while performing the ramr2 (rapid alternating hand movements - right handed) activity. The second box plot of the magnitude of acceleration data captured using a Pebble smartwatch while performing the ramr2 activity.

Because of this ambiguity in correlation using statistical methods, we have chosen to rely on advanced machine learning techniques (as outlined below) to generate pertinent features.

3.2. Pre-processing Acceleration Data

Our pre-processing step for the acceleration data required sculpting the data into a (n x 5) matrix where n was the length of the data points taken for a single read, and 5 corresponds to the 5 attributes: time stamp, xacceleration y-acceleration, z-acceleration, and the magnitude. The values were normalized and padded with zeros to create a consistently sized input (n_max x 5, where n_max is the maximum length of the data) into the CNN.

3.3. Pre-processing Meta Data

Metadata about the specific read were compiled into a 'one-hot vector' which was combined with the output of the CNN before channeling into a fully-connected neural network. We encoded 4 types of features about the task at hand:

1. The handedness of the read with respect to the task
   (a) '2' if reading was from the active hand in a one-handed task
   (b) '1' if reading was from either hand in a two-handed task
   (c) '0' if reading was from the inactive hand in a one-handed task
2. The handedness of the read with respect to clinical sided-ness
   (a) '2' if the one-handed task relied on the hand with the GeneActiv wristwatch (therefore more symptomatic side)
   (b) '1' if the task was two-handed
   (c) '0' if the one-handed task relied on the hand with the Pebble smartwatch (therefore less symptomatic side)
3. The device itself
   (a) '1' if reading was from the GeneActive device
   (b) '0' if reading was from the Pebble smartwatch
4. The medication regiment (as prescribed by the study coordinators)

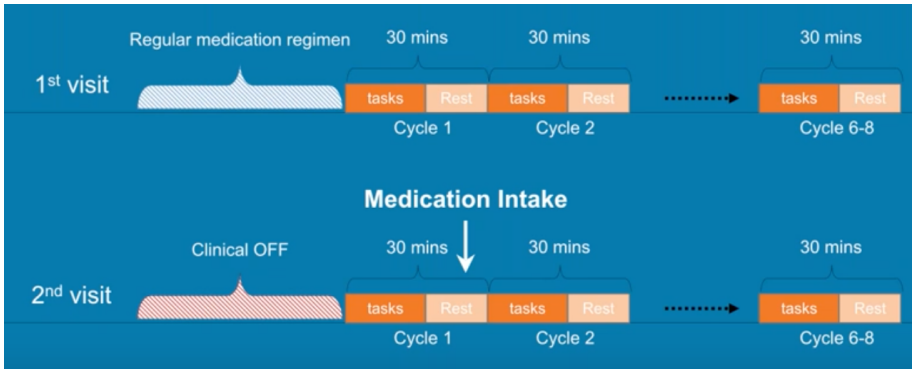

Figure 3. This is the medication cycle during the first and second visits, as provided by the organizers of the challenge.

(a) '2' if the patient was on their normal medication regiment
(b) '1' if the patient had just recently taken their medication
(c) '0' if the patient had not taken their medication

The medication cycle can be seen in Figure 3.

3.4. Balancing Training and Test Sets

The training data for tremor originally had 1661 data points with a tremor score of 0, 816 datapoints with a tremor score of 1, 407 datapoints for a tremor score 2, 38 datapoints for a tremor score of 3, and 12 datapoints with a tremor score of 4. To create a better balance, we first doubled the number of datapoints for non-zero tremor scores. Further, for our initial experiments on our LSTM [Hochreiter and Schmidhuber 1997] model, described below, we converted all non-zero tremor scores to a single value (positive), and performed binary classification instead. This gave us 1661 negative points, and 2546 positive points. We then randomly samples 1661 points from the positive set, giving us a balanced dataset.

We performed similar augmentation for the test set and ended up with 68 positive and 68 negative datapoints.

4. Methods

We developed, trained and tested three architectures that combined both the accelerometer data and the metadata features descried above. We began with a CNN that convolved over the time-series accelerometer data to extract patterns of the movements that could offer insights into the motion characteristics of tremor. We flattened and concatenated these results with the metadata and predicted tremor using a fully-connected network. Secondly, we leveraged LSTM tools (2-layered) to interpret movements at the micro level as well as the contextual one. We added the metadata as an initial input as well as a set of frames into each node of the LSTM. We developed a third architecture with attention by initializing the hidden states with an LSTM encoding from the full time-series. See the image below for reference.

4.1. CNN Implementation

Our early attempts to tackle this problem involve a model that consists of a three-stage convolutional neural network (CNN) which is then fed into a deep neural network. The

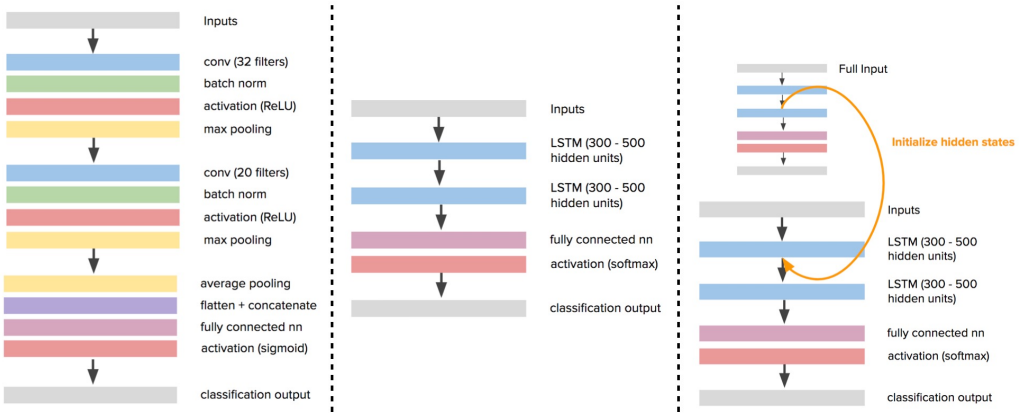

Figure 4. A graphic depicting the 3 architectures that we developed, trained and tested: CNN, LSTM and LSTM + Attention.

full architecture is delinated in further detail below:

CNN Stage 1:

- A 1-dimensional convolution with 10 filters with a kernel size of 5 and a stride of 2, as well as 'valid' padding.
- BatchNorm is applied to the channels axis of the input.
- ReLU activation function is applied to the inputs.
- MaxPooling in 1-dimension in uses a window of size 4 and a stride of 2.

CNN Stage 2:

- A 1-dimensional convolution with 20 filters with a kernel size of 10 and a stride of 2, as well as 'valid' padding.
- BatchNorm is applied to the channels axis of the input.
- ReLU activation function is applied to the inputs.
- MaxPooling in 1-dimension in uses a window of size 4 and a stride of 2.

CNN Stage 3:

- A 1-dimensional Average Pooling with a window size of 64 and a stride of 2.
- The outputs of the convolutional stages are then flattened.
- Finally, the flattened outputs are fed into densely connected neural network (DNN) layer with an output space of 1, using sigmoid function for binary classification, and a Xavier normal initializer (glorot uniform).

Our model utilizes Adam Optimization with a Mean Squared Error loss, and is trained over 10 epochs with a batch size of 10. Results from our tests of this model are reported towards the end of the notebook.

4.2. Vanilla LSTM Implementation

Long Short Term Memory [Hochreiter and Schmidhuber 1997], is a variant of recurrent neural networks. These are ideal for capturing temporal information provided from the accelerator data of the smart watches.

We fed the accelerometer data through a 2 layer LSTM to get features from the data. Then we sent the output of the LSTM through a fully connected layer to get our

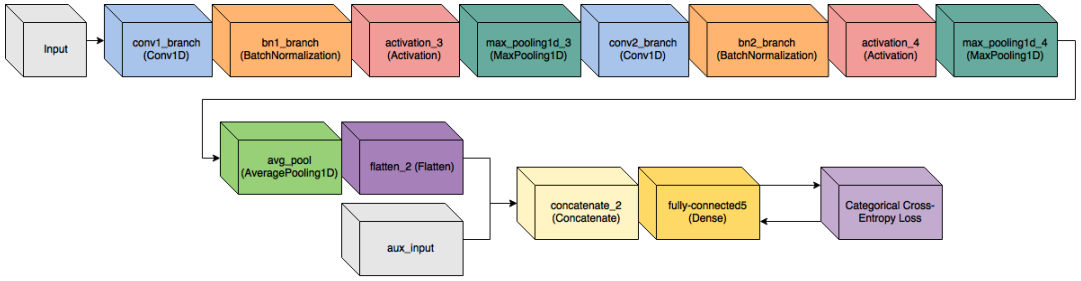

Figure 5. A visualization of the layers in the Convolutional Neural Network model.

final tremor score predictions. We performed a binary classification on tremor vs no tremor. We used the meta data available described above as the hidden state of the first layer of the LSTM - we copied the meta data as many times as required to reach the size of the hidden state. We performed experiments by sending a different number of data points from different positions in the accelerometer data. For example we sent the second hundred data points of every example to the accelerometer data.

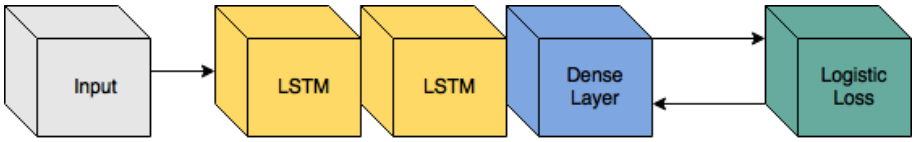

Figure 6. A visualization of the layers in the LSTM model.

For training the LSTM we used learning rates of 1e-4, hidden layers of 500, Adam Optimization function with betas of 0.9 and 0.99 and a batch size of 60.

4.3. LSTM with Attention Implementation

Attention modules [Jason and Brownlee 2017] are similar to vanilla LSTM modules, except that we first pass the entire sequence through the LSTM, and average the hidden states all the hidden states received. Thus, we encode the entire context. This is useful in situations where movement in the future is relevant to the current point we are analyzing. This was of extracting features, and finding the most important data points, and the best length and position of the input is more relevant, since none of the sequences have an inherent disadvantage of not having information from the future data points.

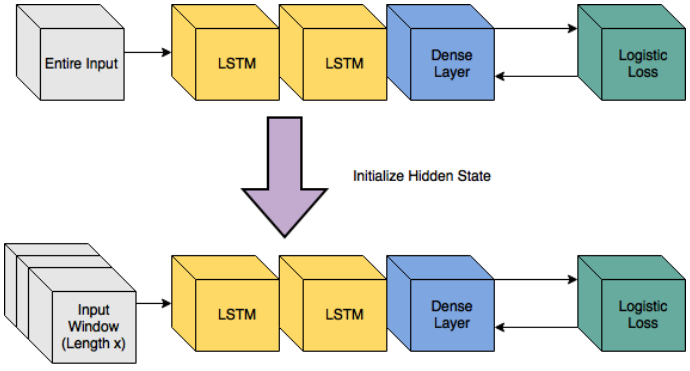

Figure 7. A visualization of the layers in the LSTM + Attention model.

For training this LSTM we followed the same hyper-parameters as our vanilla implementation.

5. Results

5.1. CNN Results

Our initial results with CNN achieved 58% accuracy on our test set, with similar precision on our training set. We arrived at this after manually tuning hyper-parameters over many iterations. In some versions of our hyper-parameters we achieved as high as 80% in our training set, although with diminishing accuracy on our testing set, which was counter-intuitive. Further exploration about the features that could create this discrepancy might reveal more intuition about optimal featurses for extraction.

Table 1. Results Comparing CNN and Vanilla LSTM accuracy on the test set.

Model	Test Set Accuracy
CNN	58%
LSTM	68%

5.2. LSTM Results

Table 2. The first table shows results on using different temporal lengths from the accelerometer data, and the second table shows results on using different temporal locations on the accelerometer data.

Number of Frames	LSTM	LSTM with Attention
100 Frames	61.67%	66.22%
200 Frames	65.00%	69.30%
300 Frames	68.33%	68.33%
400 Frames	68.33%	70.12%
500 Frames	67.50%	70.91%
Average	66.167%	68.97%

Temporal Location of Frames	LSTM	LSTM with attention
First 100 Frames	61.67%	66.22%
Second 100 Frames	57.62%	65.00%
Third 100 Frames	60.83%	62.13%
Average	60.04%	64.45%

The first experiment we performed was to see how many frames, from the beginning, are the best features for the LSTM. The results can be seen in Table 1.As we can see the accuracy increases when the frames increase from the initial 100, but stagnates around 300 frames, showing no more information is received afterwards, and the first 300 frames are good features.

The next experiment we performed was seeing which temporal part of the action matters the most. For this we compared results on the first 100 frames vs the second 100 frames vs the third 100 frames. The results of the experiment can be seen in Table 1. As we can see there are no clear patterns, but the first 100 frames are the best features out of the three experimented temporal locations. We are planning on doing more experiments by using the first 10% of the accelerometer data, second 10%, etc and seeing which part of the given task matters the most.

Our attention-enhanced LSTM always achieved the same or better results as the vanilla LSTM model (see table) indicating promising development for the integration of contextual information provided by the initialization of hidden units in our architecture.

6. Discussion & Future Work

6.1. Current performance

As it is evident from the CNN and LSTM results above, our model can identify interesting features or predict tremors with some accuracy. This can be ascribed to both basic features and models. Our attention to preprocessing, balancing training and testing sets and overview of architectures offer a strong foundation with which we hope to continue exploring other techniques and alternatives as discussed in this section.

6.2. Feature engineering

Currently, we use different time intervals and mathematical manipulation of data as features. However, we would like to extend this further, and investigate other features that can give us better results. For instance, using traditional techniques such as random forests can help us get weights of different features. This information can then be used to select more meaningful features for our CNN and LSTM, changing weights accordingly. Similarly, a classifier to filter noise or anomalies before the input is fed to our models can help boost the performance of the model. Given an extremely unbalanced dataset, we would also like to explore popular rebalancing techniques such as modifying the weight contributions/datapoint such that it is inversely proportional with the class frequency. We are also working on a visualization framework that will help track the impact of individual features and obtain a more concrete understanding of the combination that works best. Besides these techniques, we are experimenting with using power spectrum of the acceleration data, and range and standard deviation within specific windows in addition to just changes in acceleration.

Python package tsfresh helps to automatically extract hundreds of features from time series. These include basic features such as the number of peaks, the average or maximal value, or more complex features such as the time reversal symmetry statistic. We have already started thinking about the relevance of some of these features for this task, and will be exploring the package in much more detail. We have also considered other algorithms that help analyze sensor data (e.g. pedestrian dead-reckoning algorithms to detect footsteps). While we do not observe a direct application of these algorithms to our models, it seems that some key ideas might be useful.

6.3. Next Steps

For both the CNN and LSTM approaches, one important future step would be to continue to tackle the variance in the accuracies between train and test. In both the CNN, Vanilla

LSTM, and LSTM with attention, we observed higher accuracies on our training data than on our test data. This problem suggests that we are likely overfitting on the training data. To work on generalizing our model, we included two layers of dropout in our CNN model (with a dropout ratio of 0.3). However, this strategy didn't produce significant differences in our accuracy variance, and looking ahead, this could be the focus of future optimization. Another technique we tried was data augmentation. This was not only to balance our training data, but also served to increase the generalizability of our model. Nonetheless, this is a weakeness of our model that we would seek to improve. We could also look to adopt more complex regularization techniques, as well as other strategies to prevent overfitting, such as early stopping.

The most recent results of the Dream Challenge reveal almost 75% accuracy on a 5-class tremor score. Some of the most highest-scoring contestants were able to do so by optimizing on meta-data (information about the task and individual), sometimes without any machine learning integration and simple statistical models. One weakness of this method is that it may over-fit to these particular individuals and may not scale well to other situations. We would hope to learn from these methods by further developing our integration of metadata, perhaps through a robust and learned embedding model.

Furthermore, there is reason to suggest that a combination of CNN and LSTM model could leverage both micro digital patterns as well as contextual features in a seamless way. This intuition is enhanced by the work of [Ordonez and Roggen 2016] who developed such an architecture for their purposes. We propose this as another one of several future directions.

7. Acknowledgements

This study and data exploration was made possible by the Parkinson's Disease Digital Biomarker DREAM Challenge, which is funded by the Michael J. Fox Foundation for Parkinson's Research and the Robert Woods Johnson Foundation. We would also like to thank the Stanford Vision and Learning Lab to provide us with GPU resources to work on this project.

We owe special gratitude to the reviewers from the CS 273B course who offered insightful feedback about the description and clarity of our model as well as additional feedback about our graphs and figures, which we updated in this version. We also modified our introduction and description of the figures so that they more seamlessly integrated with the overall arc of our research. A critical component of our model that we fixed in this revision was debugging our loss function for the CNN model which was once producing "NaN" as well as attempting numerous tyeps of experiments with the LSTM model. Both of these enhancements were very effective and clearly outlined in our final product.

References

A. J. Espay, P. Bonato, F. B. N. W. M. J. M. D. J. K. B. M. E. A. M. F. H. A. E. L. R. R. J. G. A. N. M. H. M. A. L. I. L. T. S. E. R. D. M. A. B. K. K. A. K. C. G. J. F. D. G. M. L. K. J. M. H. B. R. B. S. P. (2016). Technology in parkinson's disease: Challenges and opportunities. *Mov Disord*, 31:1272–1282.

A. Silva de Lima, L. Evers, T. H. L. B. J. H. M. L. Y. O. B. B. and Faber, M. (2017). Freezing of gait and fall detection in parkinson's disease using wearable sensors: a systematic review. *264*, page 1642–1654.

G. Ebersbach, H. Baas, I. C. M. M. and Deuschl, G. (2006). Scales in parkinson's disease. *J Neurol*, 253(4):iv32–iv35.

Gamboa, J. (2017). Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887*.

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

I. M. Pires, N. M. Garcia, N. P. and Flórez-Revuelta, F. (2017). Pattern recognition techniques for the identification of activities of daily living using mobile device accelerometer. *In Review*.

Jankovic, J. (2008). Parkinson's disease: clinical features and diagnosis. *J Neurol Neurosurg Psychiatry*, 79(4):368–376.

Jason and Brownlee (2017). Attention in long short-term memory recurrent neural networks. *Machine Learning Mastery*.

N. Hammerla, S. Halloran, T. P. (2016). Deep, convolutional, and recurrent models for human activity recognition using wearables. *In Proc. IJCAI*.

Ordonez, F. J. and Roggen, D. (2016). Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *sensors*, 16:115–140.

P. Malhotra, L. Vig, G. S. and Agarwal, P. (2015). Long short term memory networks for anomaly detection in time series. *ESANN 2015 proceedings*.

S. Del Din, A. Godfrey, C. M. S. L. and Rochester, L. (2016). Free-living monitoring of parkinson's disease: lessons from the field. *Mov Disord*, 31(9):1293–1313.

S. Patel, H. Park, P. B. L. C. and Rodgers, M. (2012). A review of wearable sensors and systems with application in rehabilitation. *J Neuroeng Rehabil*, 9(1):1.

Y. B. Yang, M. N Nguyen, P. P. S. X. L. L. and Krishnaswamy, S. (2015). Deep convolutional neural networks on multichannel time series for human activity recognition. *IJCAI*.