Mehrgan Khoshpasand

July 30, 2018

Mr. Alireza Manashty, Dr. Cook
CS4997 Supervisor, Course Coordinator,
Faculty of computer science.

Dear Messrs. Manashty and Cook:

Here you can find the final thesis for the CS4997, Honors Thesis. The estimated work for the project was 160 person-hours but it took about 185 person-hours. For additional information please contact me at mkhoshpa@unb.ca.

Best regards,

Mehrgan Khoshpasand

University of New Brunswick, Faculty of Computer Science
CS4997 SUMMARY SHEET
Summer 2018

Student Name: Mehrgan Khoshpasand

(signature): Khoshpasand

Student ID# :: 3540117

Email ID:: mkhoshpa@unb.ca

Mailing address for this term:: 20 Dunns Crossing Rd.
Street, Apartmentt #14

City, Province, Postal Code Fredericton NB E3B2A4

Phone (day) : ( )(506)4712778

Phone (night): ( )

Thesis Title: A cloud-based mobile human fall forecasting system using recurrent neural networks

Computer Science Supervisor (name): Alireza Manashty

Date Submitted: July 30th 2018

Phase Title	**Estimate**		**Actual**	
	Person-Hours	Completion Date	Person-Hours	Completion Date
Background Research	40	June 10th	40	June 10
Training the Model	40	June 22nd	60	July14
Deploying the model	25	July 2nd	20	July14
Writing the Draft	35	July 16	40	July16
Finalizing the Thesis	10	July 30	15	July 30
			10	
Preparing for the Presentation	10	TBD		
	Total: 160		**Total:** 185	

Keep this Summary Sheet updated during the life of the thesis project. Submit a copy of the updated Summary Sheet with your Plan and attach the final version of this Summary Sheet as an Appendix in your Thesis.

Towards A Smart-Phone-Based Fall Forecasting System using Machine Learning

Mehrgan Khoshpasand Foumani

July 2018

Abstract

A forecasting system to not only detect but to predict human falls can prevent major injuries. Human fall can be life-threatening for the elderly people. In this project, a fall forecasting system is proposed using smart-phones sensors, cloud IoT infrastructure, and machine learning. To enable fall forecasting, first along short-term memory (LSTM) recurrent neural network (RNN) model is trained using URFD fall dataset. The mobile accelerometer sensor data is sent to the deployed cloud-based model to predict falls 20 time-steps before happening. An iOS mobile app is used to collect the data and deliver the forecasting results to the user. The promising results indicate that given enough training data, the model and system can be used to forecast falls in real-time to prevent serious fall-based injuries.

Contents

1 Introduction

Human falls are one of the significant safety hazards, especially among the elderly. A fall forecasting and preventing system can be very beneficial to seniors and can help them to live a better life. Preventing falls also can help societies by reducing the cost of health and social services[1]; especially for Canada that now has an aging population and seniors outnumber children. In the past, some fall detection methods have been proposed that make use of motion sensors. In recent years, the number of seniors with smart-phones has increased, and all the smart-phones have motion sensors inside them. Now more than 4-in-ten seniors are smart-phone users in the U.S, and this number has increased in recent years.

While there have been earlier works that used smart-phones to detect a fall, very little research has been done to predict or forecast a fall on time that the fall can be predicted [2]. For Instance, the work by Yang et al. (2013) predicts a fall 0.4 seconds before happening[3]. This project aims to forecast a fall when the fall can be prevented by sending an alert to the user.

This document presents a Fall Forecasting System on Smart-phones using Machine Learning as the project for Honors Thesis (CS4997) in summer 2018. The system uses the raw accelerometer data coming from smart-phone sensors and a neural network to forecast falls and alert the user. Finally, the usefulness of such a system in predicting falls is investigated.

The report is formed as follow; section 2 contains a literature review of related works; a state-of-the-art survey of fall detection-prevention systems that use the smart-phones' sensors. Also, section 2 presents a background of materials used and discussed in the system, section 3 presents an overview of the architecture of the fall forecasting system, section 4 evaluates the system, and finally, future works are discussed.

The proposed project provides a state-of-the-art survey of fall detection-prevention systems that use the smart-phones' sensors. Besides, the report contains a detailed description of the architecture of the proposed model and a comparison of the model with the existing methods. The report also provides in-depth information about some of the possible machine learning algorithms that can use the smart-phones motion sensors' raw data to forecast falls before happening.

2 Literature review

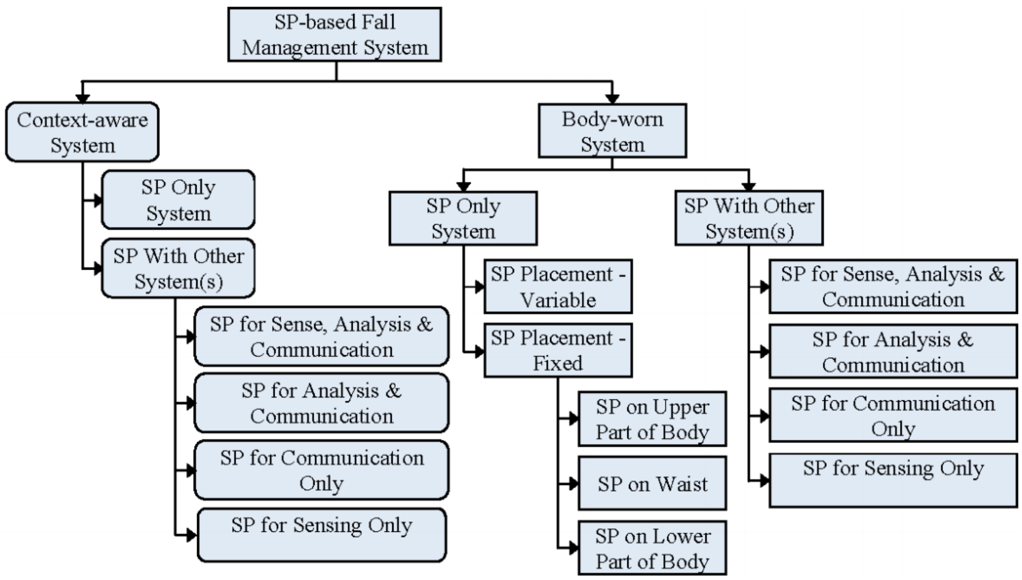
2.1 Fall Detection and Prevention systems

As was discussed in section 1, human falls are one of the most significant health problems among the elderly. According to The World Health Organization, 28%-35% of people aged over 65 falls at least once a year [2]. Moreover, falls can be more dangerous for people having high-risk factors or living alone. Falls also reduce the independence of elderly and increase the health care cost[2]. Therefore, having a system that can predict or detect a fall at the early stage can be very beneficial to society and individuals. In recent years, academia and private companies proposed many smart-phone based fall detection/prediction systems. The following subsection contains information about these systems.

2.1.1 Smartphone-Based Fall Detection and Prevention systems

In general, all the smart-phone based fall detection-prevention systems follow the same architecture. These systems use smart-phone sensors to collect data, analyze this data to detect the difference between falls and other activities of daily living, and finally communicate the result to the user or others using

Figure 1: Taxonomy of smartphone-based systems regarding sensing mechanism and position of the sensor.[2]



real-time messaging, email, SMS and other means. [2].

The work by Habib et al.[2] presents a complete taxonomy and comparative analysis of smart-phone-Based solutions for fall detection and prevention system. In summary, fall detection systems can be categorized with regards to the type of sensors used in the system; the analysis was done on the data, and communicate performed. The following figures are adapted from the work by Habib et al. to show the taxonomy of smart-phone based fall detection-prevention systems.[2]

Figure 2: Taxonomy of smartphone-based systems regarding communication

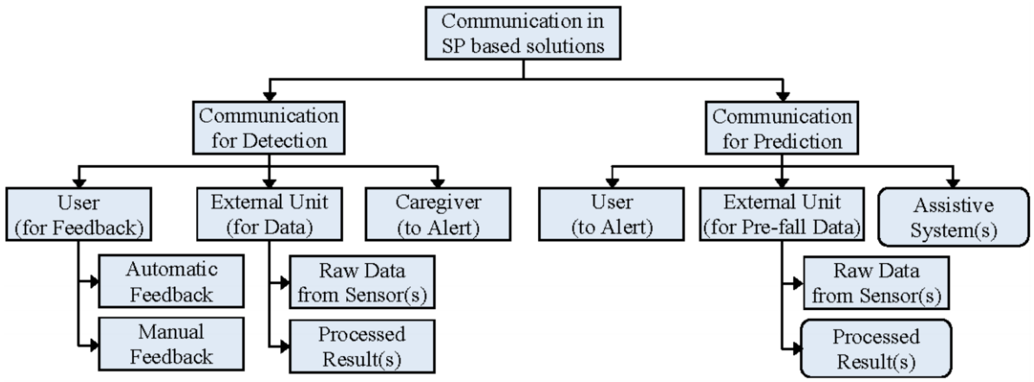
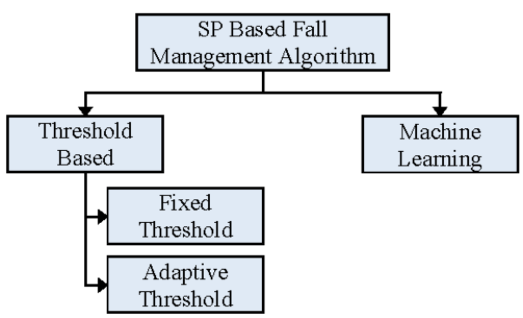


Figure 3: Taxonomy of smartphone-based systems regarding analysis algorithm



Tsinganos et al. [10] proposed a fall detection system that uses android accelerometer sensor with a threshold algorithm to distinguish falls and activities of daily living. In addition, They enhanced the accuracy by a k Nearest Neighbor (kNN) classifier.

Shen et al. [6] proposed a system that can predict falls with a high success rate by recording humans' gate data. The system analyzes the data with fuzzy Petri net model to identify the human's actions.

2.1.2 Other Fall Detection and Prevention systems

Although smart-phones are great tools for fall detection and prevention systems, these systems use a wide range of tools. Many of this tools are wearable devices that are designed for this purpose and require the user to wear them on a fixed part of the human body. The wearable device can provide more accurate data but wearing it on a daily basis can be difficult especially for the elderly.

Yang et al. [3] proposed a fall prediction algorithm that can predict human falls 0.4 seconds before happening. Their fall prediction algorithm adopted a neural network which predicts falls from accelerometer data. The accelerometer data comes from a wearable device which must be worn by the users.

Tong et al. [9] designed a cheap wearable device that collects tri-axial accelerometer data. They proposed a hidden Markov model (HMM)-based method that uses the data from the accelerometer. They claim their method predicted 100% of falls 200ms-400ms before happening.

2.2 Deep learning and Long Term Short Memory(LSTM)

Deep learning is a sub-field of machine learning, where many layers of information-processing stages are exploited inside a Neural Network (NN). The goal of deep learning is to automatically discover the representations needed for detection or classification of raw data with minimal engineering and domain expertise needed. Deep Learning achieves this goal by feeding the raw data to the first layer. Each layer consists of numbers of units. Each unit applies a simple non-linear function to the weighted sum of inputs that are coming from the previous layer. A layer transform representation of the previous layer into a slightly higher level of abstraction and pass it to the next layer. During the training, to minimize some error function, DL algorithms adjust some internal parameters, called weight. The weights are adjusted by calculating the gradient vector of the error for each weight. Gradient vector shows how much the error changes by changing the weight by a tiny amount. This method of finding the local minimum of a function is called Gradient Descent(GD). Since in GD method weights are adjusted for each sample; it is very costly for big training datasets. Instead, Stochastic Gradient Descent (SGD) is often used. In SGD, average gradient vector is computed for a small number of samples from the training set, and weights are adjusted accordingly [11]. Surprisingly, this simple procedure usually finds a good set of weights[11]. Numerically evaluating an analytical expression of the gradient can be costly. Hence, most DL algorithms use the back-propagation algorithm to compute the gradient. Back-propagation is an inexpensive procedure that allows information to flow backward through the network. [5] In a nutshell, backpropagation is an application of chain rule; computing the gradient of the function in respect of input by working backward from the gradient of function in respect of output. Convolutional Neural Network (CNN) is a class of deep neural network that has been applied with great success to the detection, segmentation, and recognition of objects and regions in images. The computer-vision community has recently widely adopted CNNS.[11]

Recurrent Neural Networks(RNNs) are a family of neural network that

Figure 4: Adapted from [11] A. Shows equations for the forward pass in a multi-layer neural network with two hidden layer. At each unit, a nonlinear function is applied to the weighted sum of the outputs of the units in the layer below. Then the unit passes the its output to the layer. B. Shows the backpropagation procedure.

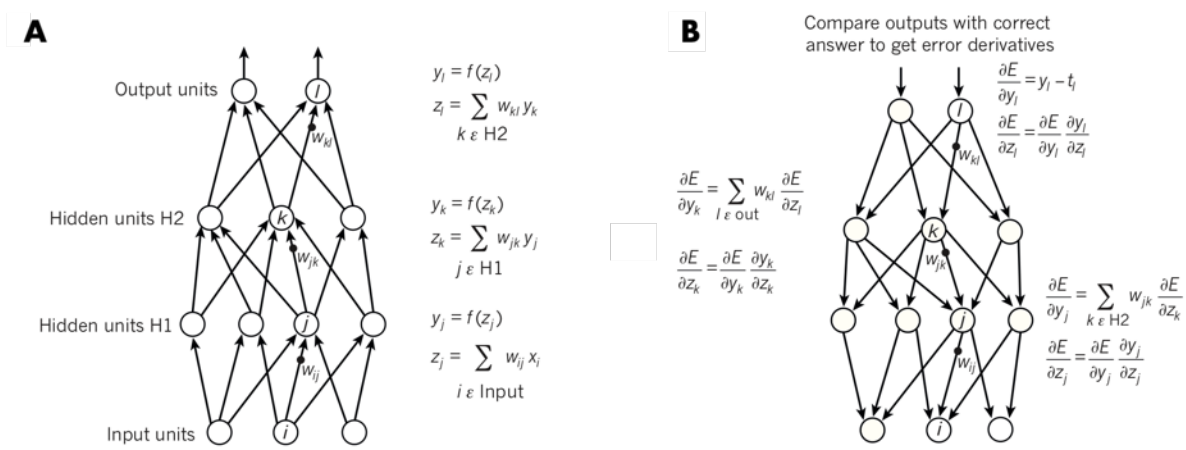
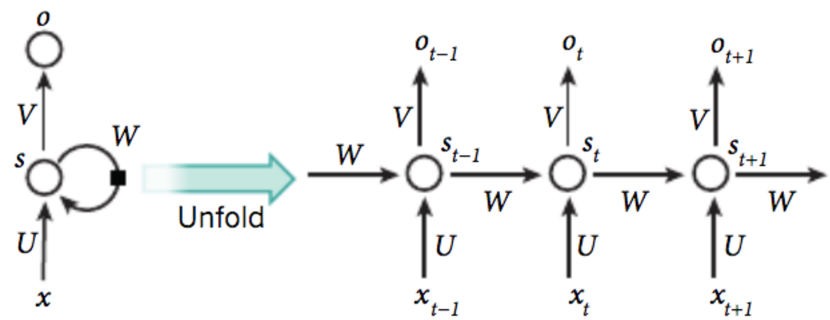


Figure 5: Adopted from [11] A Recurrent Neural network Unfolded over time



is specialized in processing sequential data.[5] At the time of processing an element of the sequence, RNNs maintain information about the history of the sequence in a 'state vector', in their hidden units.[11]. Many RNNs use the following equation to define their hidden units[5].

$$h^{(t)} = f(h^{(t-1)}, x^{(t)}, \theta) \tag{1}$$

In the equation 1, h is used to represent the state of the network.

RNNs can be seen as feedforward networks that all the layers share the same weights. Although RNNs' purpose is to learn sequential dependencies, in practice, it is difficult to store information for very long.[11]

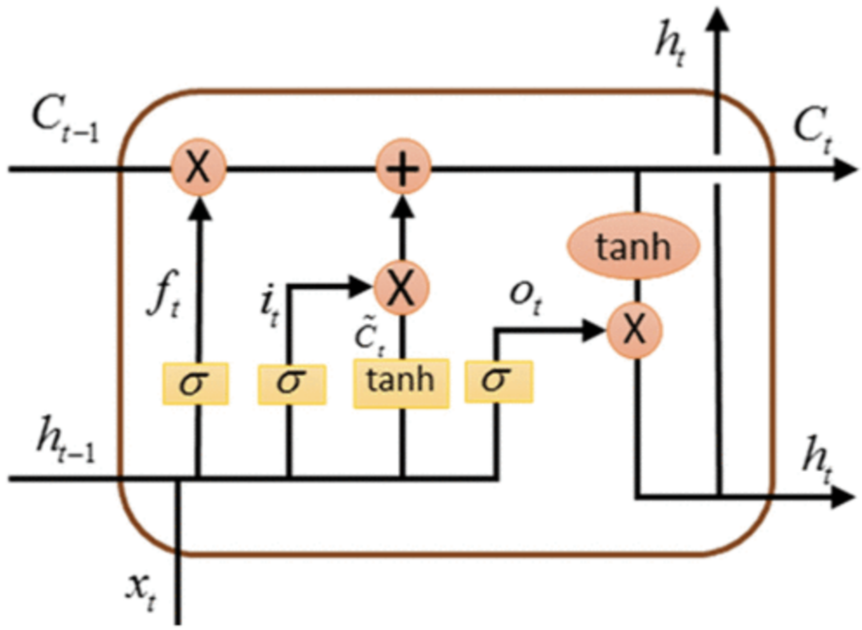
One of the critical problem in training RNNs is backpropagated gradients usually explode or vanish over many time steps. This problem is the result of growing or shrinking the backpropagated gradients at each time step.

Hochreiter and Schmidhuber introduced Long Short-Term Memory (LSTM) [13], which is one of the most important advances in recurrent networks since it is not affected by vanishing and exploding gradient problems. The core contribution of LSTM is self-loops to produce paths where the gradient can flow for a long period.[5]. LSTMs are good at learning long-range dependencies [12]. LSTM uses a new structure called a memory cell. Cells are connected to each other recursively to replace the common hidden units. In a cell, weights are gated which means weights can change dynamically based on the input sequence.[5] A memory cell is composed of four main elements: an input gate, a forget gate, an output gate, and cell activation vectors. The input gate allows the incoming signal to change or block the state of the memory cell. The output gate can shut off the output of the memory cell. The forget gate controls the weight of its previous state [12][5].

3 Proposed fall forecasting system

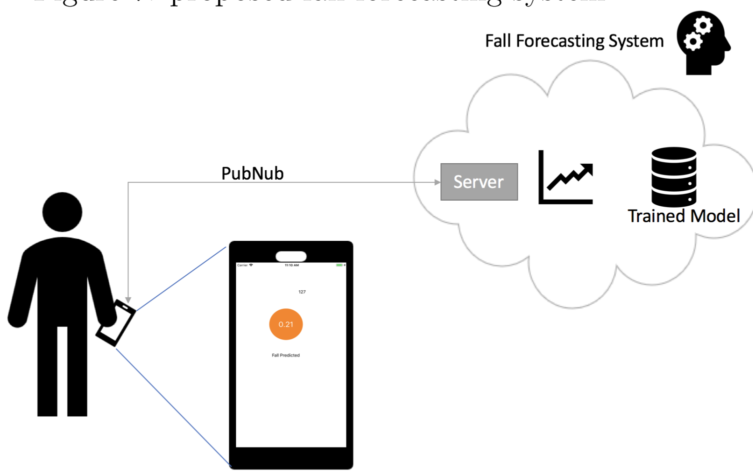
At the time of designing our fall forecasting system, the property that was considered necessary was easiness to use. The only requirement for the users

Figure 6: Adopted from [12] Structure of a LSTM memory cell



is to have a smartphone and carry it with themselves as most people do in their daily life. The problem that this requirement raised was finding a fall dataset based on the variable placement of the dataset. Since there is not a perfect dataset for this purpose, it was decided to use UR fall dataset [14]. UR fall dataset contains 70 sequences, which consists of 30 mocked falls and 40 activities of daily living. Fall events are recorded with 2 Microsoft Kinect cameras and corresponding accelerometric data. ADL events are recorded with only one device (camera 0) and accelerometer. In the proposed fall forecasting system, only accelerometer data is used. Accelerometer data were

Figure 7: proposed fall forecasting system



collected using PS Move at 60Hz. Each fall or ADL event consists of between 150-1000 time step with the corresponding tri-axial accelerometer data. The iOS application is responsible for getting the accelerometer data from the user. The app gets the Sensor data from OS and sends the data to cloud in real-time using IoT infrastructure that PubNub provides. Accelerometer data is captured at 60Hz to be consistent with the dataset.

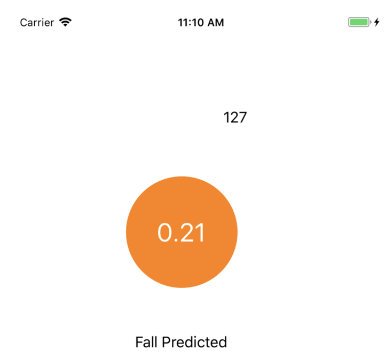
The reason iOS was chosen since I was already familiar with Android development and developing iOS app provided a great learning opportunity. An Android application can also provide accelerometer data. The following is part of the code of the application that sends accelerometer to the server. The code is in Swift.

```
motionManager.startAccelerometerUpdates(to: OperationQueue.
    current!, withHandler: {
            (accelerData:CMAccelerometerData?, error: Error?) in
            if (error != nil ) {
                print("Error")
            } else {
                let accelX = accelerData?.acceleration.x
                let accelY = accelerData?.acceleration.y
                let accelZ = accelerData?.acceleration.z
                var messageDictionary = [
                    "x":accelX,
                    "y":accelY,
                    "z":accelZ
                    ] as [String : Any]
                let jsonData = try! JSONSerialization.data(
    withJSONObject: messageDictionary)
                let jsonString = NSString(data: jsonData,
    encoding: String.Encoding.utf8.rawValue)
                self.client.publish(jsonString, toChannel:
    targetChannel,
                                        compressed: false,
    withCompletion: { (publishStatus) -> Void in
                                            if !publishStatus.
```

```
isError {
                                        // Message
successfully published to specified channel.
                                    }
                                    else {
                                        //error
                                    }
                }})
            }
        })
```

The server, which is written in NodeJS, make the prediction based on the received sensor data and a Keras model that is imported into TensorFlowJS.The model acts as a prediction engine for the fall forecasting system and is an essential part of the system. However, the model can easily be replaced by an improved model whenever it is available. The model predicts if a fall is going to happen in next 20 time-steps. After making the prediction, The server sends the results back to the smartphone in real-time. Finally, if a fall is forecasted, the application alerts the user on the screen. The following is the screenshot of the application that forecasted a fall.

Figure 8: A screenshot of application when a fall is forecasted. the Overall delay of 127 ms is taken into the account so the fall is going to happen in 0.21 second. This delay is the results of communication between phone and server and the time takes for the server to make the prediction based on the inputs.



4 Experimental results

The Keras model is an LSTM network that acts as a binary sequence classifier for fall or non-fall classification. The input of LSTM network is 20 past accelerometer data. A two-layer neural network with 20 LSTM cells as input layer and one dense layer as output is used. To convert the time-series to supervised learning, the column in the data frame that containing information about fall/no-fall is shifted 20 rows using pandas shift() function. In the training process, after transforming the time-series to supervised learning, the dataframe is given to the model and model gets trained. The model is trained using 33 ADL time-series and 25 fall time-series. During the testing process, five fall time-series and 7 ADL time-series are randomly chosen and given to the model that was not given in the training process. The model successfully labeled all 7 ADL as ADL but only forecasted 3 of 5 falls 20 time-steps before happening.

Activity	train	test	predicted
fall	25	5	3
ADL	33	7	7

The following chart compares the proposed system to the relative studies by Shen et al.[6], Yang et al.[3], and Tong et al.[9].

System	test precision rate	test recall rate	time of prediction
Proposed	100%	60%	333ms
Shen	79.69%	91.60%	vary
Yang	46.66%	70.02	400ms
Tong	100%	100%	200-400ms

5 Conclusion

Due to the unpredictability of the events that can cause some of the falls; forecasting falls can be a challenge in real life. In this document, a fall forecasting system is proposed that make the prediction based on accelerometer data collected by smart-phones. Although the model was good at predicting Activity of Daily life, it only was able to predict 60% of the falls. The low accuracy in predicting falls could be the result of many factors. The most crucial factor can be the size of the dataset which is consist of only 30 falls. Traditionally, a more significant dataset is needed to have a good Neural Network.

Moreover, human falls of UR dataset are mocked and can be different from real falls. Although the proposed system is not producing the desired result at this time, it can be improved by training a better model or using a different dataset. It also is concluded that having a cloud-based fall forecasting system can harm the overall performance by adding communication lag.

6 Future works

The accuracy of the model can be enhanced by trying some techniques other than Although the recurrent neural network has shown promising results in working with time-series, the results can be enhanced by trying other techniques. In the future, we also can eliminate the communication lag by embedding the forecasting element inside smart-phones. Embedding might also lead to better battery consumption especially when cellular is being used. I hope by open sourcing this system in the future; this project can take a small step in helping the research community and people with high risk of falls.

7 Acknowledgment

I would like to thank my supervisor, Alireza Manashty, for his support. Also, I want to thank the course coordinator, Dr.Cook for providing useful feedback. Finally, I'd want to thank the University of New Brunswick for providing the resources for this research.

References

[1] The Frailty. Epidemiology of Falls (world report).pdf. pages 3–7, 2001.

[2] Mohammad Habib, Mas Mohktar, Shahrul Kamaruzzaman, Kheng Lim, Tan Pin, and Fatimah Ibrahim. Smartphone-Based Solutions for Fall Detection and Prevention: Challenges and Open Issues. *Sensors*, 14(12):7181–7208, 2014.

[3] Shih Hung Yang, Wenlong Zhang, Yizou Wang, and Masayoshi Tomizuka. Fall-prediction algorithm using a neural network for safety enhancement of elderly. *2013 CACS International Automatic Control Conference, CACS 2013 - Conference Digest*, pages 245–249, 2013.

[4] Christopher M Bishop. *Pattern recognition and machine learning, 5th Edition*. Information science and statistics. Springer, 2007.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[6] Rong Kuan Shen, Cheng Ying Yang, Victor R.L. Shen, and Wei Cheng Chen. A Novel Fall Prediction System on Smartphones. *IEEE Sensors Journal*, 17(6):1865–1871, 2017.

[7] Anne Ngu, Yeahuay Wu, Habil Zare, and Andrew Polican B. Fall Detection Using Smartwatch Sensor Data with Accessor Architecture Anne. 10347(June), 2017.

[8] Ramesh Rajagopalan, Irene Litvan, and Tzyy-Ping Jung. Fall Prediction and Prevention Systems: Recent Trends, Challenges, and Future Research Directions. *Sensors*, 17(11):2509, 2017.

[9] Lina Tong, Quanjun Song, Yunjian Ge, and Ming Liu. HMM-based human fall detection and prediction method using tri-axial accelerometer. *IEEE Sensors Journal*, 13(5):1849–1856, 2013.

[10] Panagiotis Tsinganos and Athanassios Skodras. A smartphone-based fall detection system for the elderly. *International Symposium on Image and Signal Processing and Analysis, ISPA*, (September):53–58, 2017.

[11] Yann LeCun, Yoshua Bengio, Hinton Geoffrey, Geoffrey Hinton, Lecun Y., Bengio Y., Hinton G., and Hinton Geoffrey. Deep learning. *Nature*, 521(7553):436–444, 2015.

[12] Xu Tao and Zhou Yun. Fall prediction based on biomechanics equilibrium using kinect. *International Journal of Distributed Sensor Networks*, 13(4), 2017.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long Short Term Memory. *Memory*, (1993):1–28, 1996.

[14] Bogdan Kwolek and Michal Kepski. Human fall detection on embedded platform using depth maps and wireless accelerometer. *Computer Methods and Programs in Biomedicine*, 117(3):489–501, 2014.