iWrist: Real-Time Continuous Blood Pressure Monitor

Zihao Zou
zihao@wustl.edu
Washington University in St. Louis
St. Louis, Missouri, USA

Ran Xu
ran.x@wustl.edu
Washington University in St. Louis
St. Louis, Missouri, USA

Di Huang
di.huang@wustl.edu
Washington University in St. Louis
St. Louis, Missouri, USA

ABSTRACT

The pressure of flowing blood against the walls of blood arteries is known as blood pressure (BP). The heart pumps blood through the circulatory system, which causes the majority of the pressure. Blood pressure is an important signal for cardiovascular health. This paper presents iWrist, a wearable sensing system for robust BP monitoring on wrist with Photoplethysmography (PPG). We developed a system involved with wristband, smartphone and cloud service, which enables us to continuously monitor the blood pressure. The proposed iWrist wristband will attempt to help doctors better diagnose high blood pressure. It will measure, store, and present BP data on a smartphone. To accomplish this feature, iWrist will use an Raspberry Pi 2W microprocessor and a MAX30102 PPG integrated sensor. This sensor currently has the highest noise rejection ratios in the integrated PPG market. Its high fidelity will allow the microprocessor to process the PPG data more accurately and employ machine learning techniques for diagnosis and prediction. We proofed the portability of our device and demonstrated the accuracy of our model.

CCS CONCEPTS

• **Applied computing** → Consumer health.

KEYWORDS

Smartwatch, mobile sensing, deep learning, respiratory rate

1 INTRODUCTION

According to a report from the World Health Organization, high blood pressure (HBP) is responsible for approximately 7.5 million deaths per year[8]. That is the equivalent of 12.8% of all total deaths that occur worldwide. Unfortunately, most people live with this condition undiagnosed; until it is too late. This is perpetuated both by high-cost barriers to professional medical care and the fact that Hypertension is not easy to detect, even within clinical settings [5]. These two factors combined have earned high blood pressure, also known as hypertension, the notorious reputation of a "silent killer"[7]. In the US alone, approximately 13 million people are unaware of their condition[11].

Traditionally, hypertension has been diagnosed and monitored with cuff-based, mercury sphygmomanometers at doctors' offices. This equipment has been the standard for nearly a century. Recently, the United States Preventive Services Task Force (USPSTF), along with other health groups around the world, have begun the push for diagnosis and home monitoring using ambulatory, 24-hour blood pressure monitors (ABPMs)[10]. ABPMs are portable, cuff-based devices about the size of a walkman that read blood pressure (BP) levels throughout the day, usually every 20 minutes. The extra data produced by these devices help reduce the white-coat effect, or the swings in data incurred due to nervousness, suffered by traditional office-readings. ABPMs allow for more accurate diagnoses and monitoring of HBP.

However, there are three significant issues with current home ABPMs. The first is their price. Even the most affordable, reliable model can run upwards of $1,000. Second, most recommended ABPMs are not readily available for purchase by the general public. They require doctor's prescriptions, which leave out the people who avoid healthcare visits due time and or cost constraints. And third, even if patients manage to access one of these devices, it is difficult to accurately read the data for diagnosis and monitoring without any technical background.

With recent advances in microsensor and low-power transmission, a cuffless blood pressure measurement approach has been proven feasible by utilizing the correlation between photoplethysmography(PPG) and blood pressure[4]. The former terminology refers to a human's optical signal obtained from the human's blood vessel. This signal has been proven to have a correlation with blood pressure[1]. Furthermore, some researchers suggested that machine learning could be employed to better estimate blood pressure[3].

2 GOALS AND REQUIREMENTS

The proposed iWrist is a wristband that comfortably, accurately, and affordably monitors a user's blood pressure and heart rate throughout the day. To make the result accurate, this device utilizes photoplethysmography (PPG) sensors to match the accuracy of the most reputed ABPMs. It will also be paired with access to a mobile phone application that will help diagnose and monitor BP using neural networks. In order to reduce the cost of the device, we chose the Raspberry Pi Zero 2W as the computing unit. The detailed cost of each component is shown in Figure 1. iWrist is not intended to fully replace the good judgement of a doctor, but it will make it easy for anyone to accurately diagnose and control their BP, with an ultimate goal for wide-spread accessibility.

We need mainly three components: a microcontroller that easily fit in human's wrist, an Android application, and a cloud service. We use a microcontroller to drive an analog peripheral to obtain the PPG signal from the wrist. This peripheral contains a photodiode and led source. The data will be collected periodically and transmitted to a smartphone via Bluetooth Low Energy. The smartphone sends the data to the cloud. The cloud will analyze the signal and estimate the blood pressure with the aid of machine learning. Once the cloud detects potential hypertension, it will send an alert to the user's smartphone.

3 DESIGN

iWrist will be designed with the following four key features in mind: portability, convenience, accuracy, and real-time. The entirety of the hardware will be enclosed within a lightweight frame, affording users the ability to wear the wristband without the inconvenience

Raspberry Pi Zero 2W	MAX30102 PPG Sensor	PiSugar2 Portable Battery	Total
15	4	32	51

Figure 1: Detailed Cost for Each Part

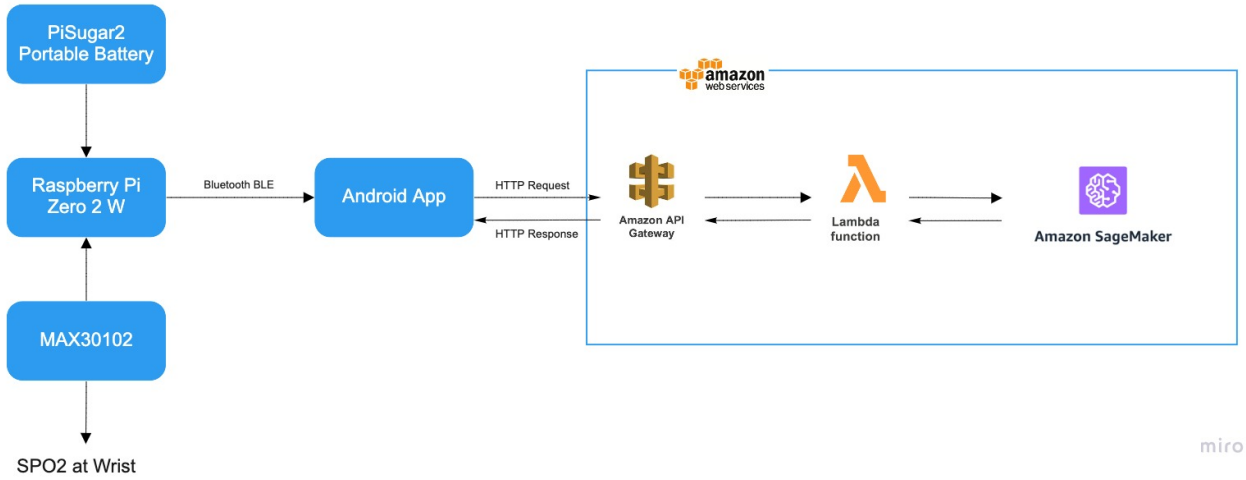

Figure 2: Overall framework of the iWrist

of added weight and discomfort. Individuals will be able to take iWrist anywhere and accurately know their current blood pressure level without having to carry a heavy ABPM.

The wristband has three components: MAX30102 PPG sensor, PiSuger portable battery, and Raspberry Pi Zero 2W. The Max30102 will collect data and send it to Raspberry Pi using the I2C protocol. The Raspberry Pi will receive data from the sensor and send it to a smartphone using Bluetooth Low Energy. The PiSuger battery can supply up to 4-hour battery life.

The Android application provide an user-friendly interface to display the wristband status. It also enables the users to display blood pressure condition in real-time.

Figure 2 depicts how the serverless architecture is used to call the deployed model to make a prediction. A client starts on the application side by calling an Amazon API Gateway API action and passing data. Furthermore, it encrypts the backend, ensuring that AWS Lambda remains within a secure private network. The Lambda function receives the argument values from API Gateway. The value is parsed by the Lambda function and sent to the SageMaker model endpoint. The model performs the prediction and sends Lambda the predicted value. The received value is parsed by the Lambda function and sent back to API Gateway. That value is returned to the client through API Gateway.

4 IMPLEMENTATION

In this section, we first present the design of our wristband which achieves the portability and convenience. We then build a Android application that enables the system receive and review the data in real-time. Finally, we develop a Amazon cloud service that accurately predicts the blood pressure based on transmitted data.

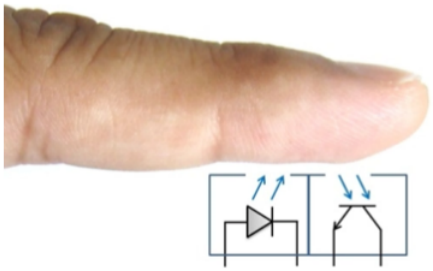

Figure 3: PPG Signal Collection

4.1 Wristband

We utilize a MAX30102 High-Sensitivity Pulse Oximeter and Heart-Rate Sensor for Wearable Health for measuring the PPG signal. As shown in the Figure 3, this sensor has two parts to complete a PPG signal detection. First, there is an LED light in the sensor. This LED light emits light into the user's wrist. The light will go to the user's vessel in the wrist and part of the light will be obtained by human blood. The second part is a light sensor on the other side of the user's wrist. This sensor will receive the light reflected by the vein. However, because part of the light was obtained by the vein, there is a difference between The amount of light emitted and the amount of light received. Furthermore, the user's blood amount will change because of the pulse, leading to the varied difference between light emitted and light received. The difference was sampled by an analog to digital converter with a frequency of 200 Hz.

After the signal was collected, we employ a raspberry pi to communicate with the sensor and request the sensor to upload the data every 20 samples through an I2C pipeline. The Raspberry Pi can automatically detect if there is a finger on top of the sensor by calculating the average signal intensity. Every seven seconds, the Raspberry Pi will pause monitoring and start to upload the collected data to a smartphone with the aid of Bluetooth Low Energy.

In Bluetooth communication, the wristband acts as a server, and the smartphone act as a client. The communication is fully controlled by the client. Before the client notifies the server to transmit data, there is no communication between them, so no power consumption is involved. After the client notifies the server to send data, the wristband will start to collect data and upload it every seven seconds.

4.1.1 Bluetooth Framework. We designed and implemented an application layer Bluetooth protocol in order to communicate between the mobile application and the wristband. As shown in the Figure 4, The protocol is intended to be easy to implement and must effectively resolve the issue in the situation of packet loss. We defined a full data set to be the PPG signals that are collected for a measurement. Each full data set contains 28 packets, and we set the packet size to be 202 bytes, which is acceptable for most of the modern mobile devices. The first two bytes are headers; the first byte is the sequence number, representing the index of complete data sets; the second byte is the packet number, representing the index of each packet within the complete data set. The other 200 bytes contain 50 floating point numbers that are signals collected from the sensor.

The mobile application will save the current sequence number and packet number after receiving a packet from the device and calculate the expected sequence number and packet number for the next packet. After receiving each new packet, the application will compare the expected sequence number and packet number to the ones that are just received. If the numbers don't match, then the data set that is transmitting is considered to be corrupted and will be discarded.

4.2 Android Application

In this section we describe a dedicated Android application that is designed to use along with the blood pressure monitor.The application utilizes Bluetooth Low Energy(BLE) to communicate with the BP monitor in order to conserve energy and elongate battery life. We will also discuss the detailed implementation of the Bluetooth transmission protocol that was crafted for the task of transmitting PPG signal and the network framework that were designed to make requests to the server remotely.

4.2.1 Application Overview. The application is designed to be clean and simple to use. There are three pages in this application, which are the home page, the find device page, and the measurement history page. The home page will display the systolic and diastolic blood pressure measurements and will provide a live update of the blood pressure once the application is connected to the BP monitor. The blood pressure reading will be updated about every 7 seconds depending on the frequency of the PPG sensor. To find the BP monitor, the user could use the find device page. The application will search for nearby Bluetooth devices for 10 seconds and filter all other devices out except the BP monitor. The user can then click on the device in order to connect. After connection, the application will automatically return to the home page and the BP monitor will start to measure without any action required from the user. Once a measurement is completed, the application will automatically save the measured value to local storage and the user can view them in the measurement history page. The application utilizes Android service packet to allow the user to switch to other applications while measuring as long as the user does not force quit the application.

4.2.2 Network Framework. To achieve the goal of making network requests asynchronously, we made a network framework that is able to post data to the server API and receive a response from the server in a non-blocking fashion. The network framework uses HTTP POST as the application layer protocol and sets the headers that are necessary for transmitting data. It contains one static method "makePostRequest" that can be called without creating a new instance of the network framework object, thus saving time and memory. Whenever the application needs to make a request to the server, the network framework will create a new thread and place the network task in this new thread. After the response is received and parsed, the framework will call the callback method that the application passed in to allow further processing.

4.2.3 Bluetooth Communication. As discussed above, the application uses BLE to communicate with the BP monitor. The detailed description of the application layer protocol of BLE transmission can be found in the hardware section of this paper. In this part of the section, we will be describing the mobile end implementation of the Bluetooth framework.

Our Bluetooth framework is extended from an open source asynchronous Bluetooth framework FastBle. Our framework is responsible for initiating a connection, starting the device notification and parsing the received PPG data using the protocol that we described above. The Bluetooth framework follows the singleton design pattern, allowing only one instance of connection to exist at a time which prevents the possibility of errors caused by trying to initiate multiple connections.

4.2.4 Data Processing. Since our model's input layer size is 875 and our sensor does not have a frequency that is divisible by 875, we need to down-sample our raw data of 1400 PPG signals(collected on a frequency of $200Hz$ over a interval of 7 seconds) to 875 signals. To achieve this, we used the fast fourier transform from the Jtransforms library to down-sample the data set in order to fit our model.

4.3 AWS Cloud Service

In this section, the goal is to establish end-to-end communication among three AWS services and allow real-time blood pressure prediction. The three AWS services we have utilized to establish a serverless AWS architecture are Sagemaker, Lambda, and API Gateway. Sagemaker is a fully managed machine learning service. It provides model hosting services for model deployment , and provides an HTTPS endpoint where the trained model is available to provide inferences. It also provides an integrated Jupyter notebook as an entry point for accessing these model related tasks. Lambda is a serverless computing service that will automatically execute code at request. It will invoke the Sagemaker endpoint to handle

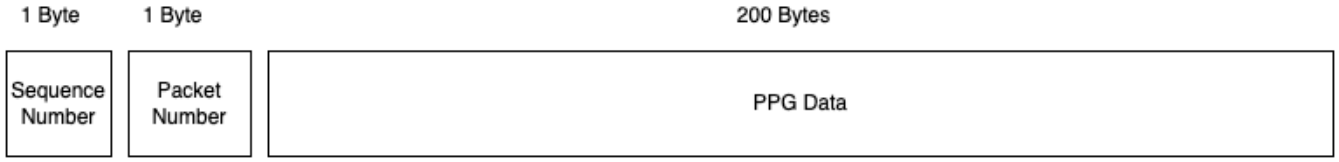

Figure 4: Bluetooth protocol

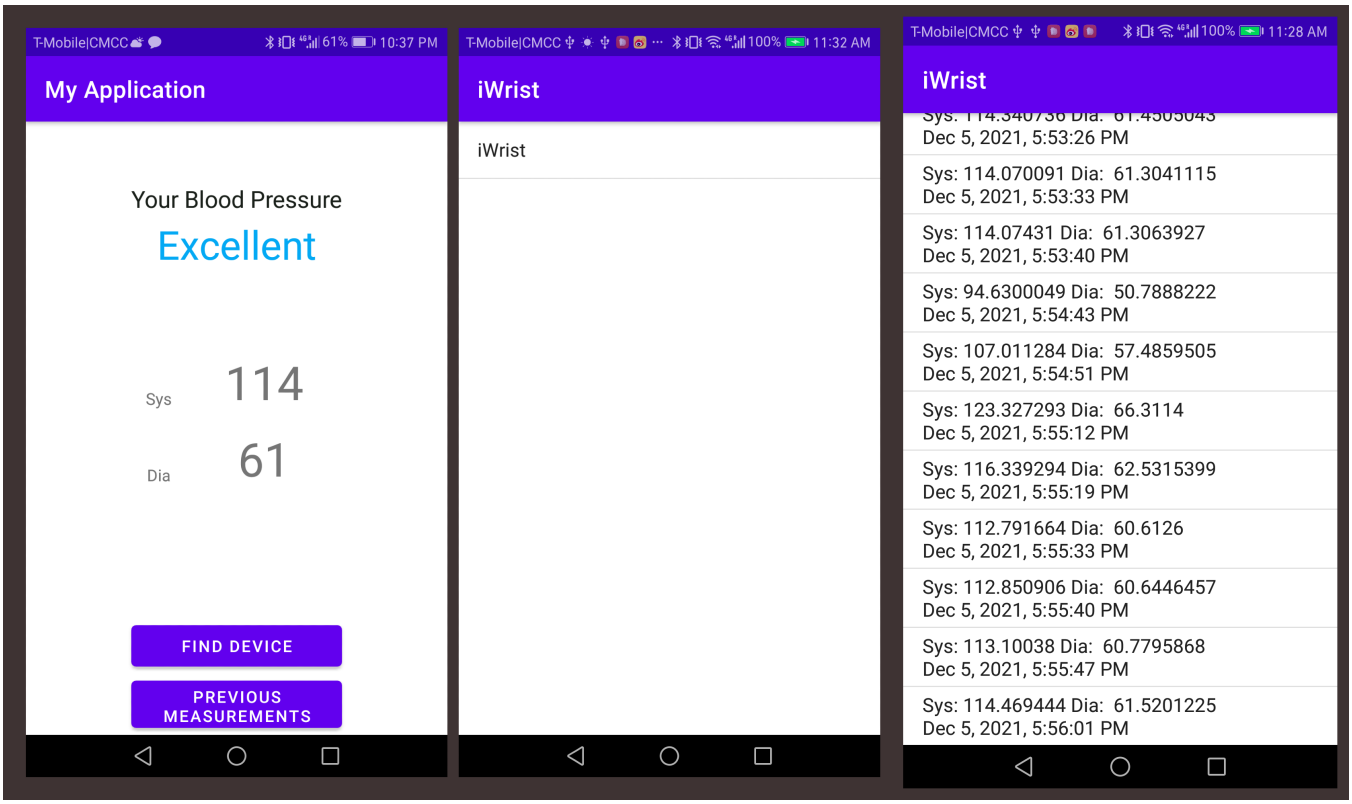

Figure 5: Sample Screenshots of the Application

prediction tasks. API Gateway is a fully managed service that allows developers to create, publish, maintain, monitor, and secure API at any scale.

4.3.1 Model Training. In our task, we train our own model by fitting the collected PPG and BP data by our device. There are 50 PPG samples with corresponding blood pressure measured by three participants. The PPG data are collected by our own device and the BP data are collected by Beurer cuffless blood pressure monitor [6]. Three participants are Asian and at the age of 24 and present normal blood pressure range (SBP < 120, DBP < 80). Based on our hardware design, each sample contains 875 PPG data and 2 BP data (SBP and DBP). We split the 50 samples to 70% training data and 30% testing data. We then build and train a Neural Network locally before deploying to the Sagemaker endpoint as training on Sagemaker is costly. We build a five-layer sequential Neural Network (NN) model with one input layer, three fully connected layers and one output layer. The input layer size is 875 and output size is 2. The nodes of each fully connected layer are 20, 10, and 5 respectively and all using RELU activation function. We compiled the model with Mean Absolute Error loss function and Adam optimizer with learning rate 10^{-3}. In summary, there are 17797 trainable parameters. We fit the model with training data with epoch = 500 and acquire the best MAE loss = 6.67 at training steps and 5.2 seconds training time. We evaluate the model with testing data and acquire MAE = 5.33. The table illustrates some of the testing accuracy.

After training the model, we save the model into two file formats, which are the required formats needed to be deployed to the Sagemaker endpoint. A JSON file contains the model topology and reference to the weights files. A Hierarchical Data Format version 5 (HDF5) file contains the weight value.

4.3.2 Model Deployment. Once acquired the saved model files, upload the model files to the sagemaker jupyter notebook instance.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 20)                17520

 dense_1 (Dense)             (None, 10)                210

 dense_2 (Dense)             (None, 5)                 55

 dense_3 (Dense)             (None, 2)                 12

=================================================================
Total params: 17,797
Trainable params: 17,797
Non-trainable params: 0
_________________________________________________________________
```

Figure 6: Model architecture

Run an ipynb file to load the model files and convert the model to a tar file, which is the required format for model deployment. Upload the model tar file to the S3 bucket and the model is ready for deployment. We then deploy the model to the sagemaker endpoint with the instance type "ml.t2.medium".

4.3.3 Lambda function. We create a Lambda function to invoke the sagemaker endpoint with setting up the endpoint configuration. The Lambda function acknowledges the input as JSON format from application end and transfers the input to Comma-separated values (CSV) format and sends it to the sagemaker endpoint.

4.3.4 API Gateway. Establish a RESTful API in API Gateway and allow it to invoke the Lambda function.

5 EXPERIMENTS

This section demonstrates the experiments that have been conducted. Although we eventually used our own training data and NN model, we have tested on an open-source training dataset and deep recurrent network. However, neither the data or model fit our collected data and generate a blood pressure reading in normal range. We first used a subset of the MIMIC-III [9] dataset used for non-invasive blood pressure prediction. This is a large scale dataset which contains over 9 million PPG and BP data. The PPG measures at 125Hz and collects for 7 seconds for a single sample. The PPG data is distributed in the range of -1 to 1, which is assumed to be normalized already. There is a pre-trained Long Short Term Memory (LSTM) model along with the dataset. We loaded the deep recurrent network and evaluated it on the dataset which generates MAE < 2. However, when evaluated on our collected data, the training loss did not converge. The undesirable performance of the model on our collected data is due to the data normalization. Our collected PPG is in the range of 700 to 2200 with the mean value = 1600. We applied a z-score normalization and attempted to represent the data in the same distribution range. However, the training still did not converge. It is clear that the different PPG measuring devices and normalization approaches lead to the different results. Considering the nature of data acquisition, a similar data distribution is needed. It is considerably difficult to find a suitable open source PPG data and we decided to train on our collected data with a total of 50 samples. We then fit our data to the deep recurrent network. However, training loss is still undesirable with MAE > 100, which implies the deep nn model may not be suitable for a sparse dataset. We started with a simple sequential model and tuned the model step by step. We trained a model with1 fully connected layer to 10 fully connected layers and also trained the model with five learning rates ranging from 10^{-5} to 10^{-1} with Adam optimizer. The best architecture of the model is 3 fully connected layers with learning $= 10^{-3}$ and the MAE = 6.67. After training, we invite two volunteers to participate in our blood pressure test. The first participant is a female caucasian at the age of 51 with hypertension I ($130 <$SBP < 140, $80 <$ DBP < 90). IWrist predicts DBP = 121 and DBP = 82. However, from the Beurer BP monitor, the DBP = 132 and DBP = 86. The second participant is Asian at the age of 24 with normal blood pressure. The predicted SBP = 107 and DBP = 67 while the BP monitor measures the SBP = 112 and DBP = 71.

6 RELATED WORKS

6.1 The use of photoplethysmography

in 2019, npj Digital Medicine published a review article that comprehensively analyzes the use of photoplethysmography in blood pressure estimation[4]. In this paper, the researchers examine the current top of the line and publications on PPG signals acquired by pulse oximeters, various theoretical methodologies used in PPG BP measurement research, and the wearability potential of PPG sensing devices. We used this article as a guideline to design our system.

6.2 RespWatch

Recently, a research team at Washington University in St. Louis announced their research outcome, which utilize a commercial wristband collecting PPG signal and predict the respiratory rate on cloud service[2]. It utilizes the machine learning algorithm to find the correlation between PPG signal and respiratory rate. Based on signal processing and deep learning, the researchers constructed two new RR estimators. In the context of moderate noise, the signal processing estimator gained impressive accuracy and efficiency. however, This research does not explore a low-cost device with minimum functionality.

7 LESSONS LEARNED

During the implementation, we have gone up against a parcel of adversity. At the beginning, we chose STM32WB55CG as the microprocessor. However, due to the low performance of this micro processor, we could not send data to smartphone timely. After struggling from the weak performance of this microprocessor, we switched our plan to use Raspberry Pi, which offered us enough running speed to process our data flow. Thus, we are aware that we need to be more cautious while choosing the suitable device.

During the model training process, we have to use our own collected data and model due to the distinct data distribution. Although we have acquired a high testing accuracy, we only used 50 samples which result in overfitting. As discussed in Experiments section, the blood pressure prediction is worse on participants with hypertension since our training data only contains normal PPG and BP data. In addition, the model itself may only work well in

our data due to the simplicity of our collected data. This shallow network may need to be changed if acquire large data scale, which a deep neural network will fit in this case.

8 CONCLUSION

Hypertension is a health risk that goes largely unnoticed and can be devastating if left unwatched. Nowadays, there are devices that can help measure blood pressure at home, but a sudden rise in blood pressure could still be dangerous and hard to respond immediately. In this paper, we presented an affordable solution to continuously measuring the blood pressure of the patient. By using a simple hardware architecture and a finely tuned machine learning model, we are able to demonstrate that it is possible to provide a live blood pressure monitor. Moreover, the saved history of measurements can help the patient's health care provider to better understand the patient's recent health status and can thus provide a more specific treatment according to the measurements history. In conclusion iWrist aims to provide a solution that is portable, reliable, convenient, and, most importantly, affordable, opening a way for a large population of cost-weary individuals to take cardiac health back into their hands.

9 FUTURE WORK

As for the mobile application, we will add more functionality to the application in future versions. We are currently considering implementing a device memorization feature, which could allow the application to automatically find the BP monitor after the first manual connection. Moreover, we are going to paginate the measurement history by date, allowing the user to find the desired record much faster. Further, we plan to also develop an iOS end application in order to attract more users.

For the model development, We may consider to collect more PPG data to enlarge our training dataset to overcome the overfitting problem. A thorough examination of the data distribution is needed when acquire large dataset and data normalization technique may need to be considered to handle gradient explosion/vanishing. Furthermore, cross validation may improve the testing accuracy when the data scale is larger. Compare the performance among different models with different hyperparameters can improve the model a lot more.

REFERENCES

[1] John Allen. 2007. Photoplethysmography and its application in clinical physiological measurement. *Physiological measurement* 28, 3 (2007), R1.

[2] Ruixuan Dai, Chenyang Lu, Michael Avidan, and Thomas Kannampallil. 2021. RespWatch: Robust Measurement of Respiratory Rate on Smartwatches with Photoplethysmography. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*. 208–220.

[3] Chadi El-Hajj and Panayiotis A Kyriacou. 2020. A review of machine learning techniques in photoplethysmography for the non-invasive cuff-less measurement of blood pressure. *Biomedical Signal Processing and Control* 58 (2020), 101870.

[4] Mohamed Elgendi, Richard Fletcher, Yongbo Liang, Newton Howard, Nigel H Lovell, Derek Abbott, Kenneth Lim, and Rabab Ward. 2019. The use of photoplethysmography for assessing hypertension. *NPJ digital medicine* 2, 1 (2019), 1–11.

[5] Elizabeth B Kirkland, Marc Heincelman, Kinfe G Bishu, Samuel O Schumann, Andrew Schreiner, R Neal Axon, Patrick D Mauldin, and William P Moran. 2018. Trends in healthcare expenditures among US adults with hypertension: national estimates, 2003–2014. *Journal of the American Heart Association* 7, 11 (2018), e008731.

[6] Stephan Lüders, Ralf Krüger, Claudia Zemmrich, Klaus Forstner, Claus-Dieter Sturm, and Peter Bramlage. 2012. Validation of the Beurer BM 44 upper arm blood pressure monitor for home measurement, according to the European Society of Hypertension International Protocol 2002. *Blood pressure monitoring* 17, 6 (2012), 248–252.

[7] EI Obeagu, UO Chijioke, and IS Ekelozie. 2018. Hypertension a great threat to human life. *Int. J. Adv. Res. Biol. Sci* 5, 10 (2018), 159–161.

[8] World Health Organization et al. 2021. World health statistics 2021. (2021).

[9] Fabian Schrumpf, Patrick Frenzel, Christoph Aust, Georg Osterhoff, and Mirco Fuchs. 2021. Assessment of Non-Invasive Blood Pressure Prediction from PPG and rPPG Signals Using Deep Learning. *Sensors* 21, 18 (2021), 6022.

[10] Albert L Siu and US Preventive Services Task Force*. 2015. Screening for high blood pressure in adults: US Preventive Services Task Force recommendation statement. *Annals of internal medicine* 163, 10 (2015), 778–786.

[11] Hilary K Wall, Judy A Hannan, and Janet S Wright. 2014. Patients with undiagnosed hypertension: hiding in plain sight. *Jama* 312, 19 (2014), 1973–1974.