Mobile Stress Tracker: The Mobile Application for Stress Detection from Human Speech

Mark Sterling
School of Science and Technology, Computer Science Dep.
Nazarbayev University
Astana, Kazakhstan
mark.sterling@nu.edu.kz

Dias Issa
School of Science and Technology, Computer Science Dep.
Nazarbayev University
Astana, Kazakhstan
dias.issa@nu.edu.kz

Askar Kossymzhan
School of Science and Technology, Computer Science Dep.
Nazarbayev University
Astana, Kazakhstan
askar.kossymzhan@nu.edu.kz

Raiymbek Mustazhapov
School of Science and Technology, Computer Science Dep.
Nazarbayev University
Astana, Kazakhstan
raiymbek.mustazhapov@nu.edu.kz

***Abstract*—The rapid development of mobile technologies has a crucial effect on healthcare. MHealth applications have improved healthcare services in different aspects. In this paper, we introduce the mobile application, based on the deep neural network designed for stress detection using human speech. Microphones, which are embedded in the majority of mobile devices allow a user to track his/her stress state ubiquitously. The paper demonstrates the real-time stress detection from the user's speech using Android-based mobile device. Additionally, the personalized approach of the application allows training on the fly for user-specific models to get better performance. The application achieves the top accuracy of 84.67% for stress detection in the environment without background noise. Our created framework utilizes token-based encryption to secure users' data and prevent unauthorized usage of the framework. To the best of our knowledge, our system is currently the only mobile application for real-time stress detection task using only the human speech with an adaptation feature.**

***Index Terms*—mHealth, stress detection, speech analysis, neural networks, Android**

I. Introduction

The increasing number of the usage of mobile information and communication technologies in health care(also called mHealth or mobile health) has got lots of attention from researchers [1]. In recent year, mobile health is showing active development and the recent report of Vendor markets calculated that in 2017 over 325,000 health apps were available in big app markets [2]. "MHealth delivers healthcare services, overcoming geographical, temporal, and even organizational barriers" [3]. The main purpose of Mobile health to provide solutions to emerging problems in healthcare services [3]. Especially, to reduce the high cost of the healthcare system, promote the awareness of people to self-care and provide direct access to healthcare systems without limitations of location and time [3]. Currently, the majority of mHealth application is designed for consumers [4]. Moreover, only the two categories such as wellness management and chronic disease management take a large part of all consumers facing mHealth applications [4]. In this paper, we would like to introduce our framework for stress detection from human speech. This system consists of 3 parts: the deep neural network for speech stress detection, the database where all users' data is stored and mobile application with user interface to interact with our framework.

The next section provides brief background information on previous works in the field of mobile health and stress emotion recognition. The third section of the paper shows our methods which were used in the creation of the model for stress detection. The fourth section is about the utilized dataset and experiments with our models. The fifth section introduces the created system's back-end details such as design, implementation, and database. The sixth section illustrates the mobile application. The conclusion is in the seventh section.

II. Literature Review

A. Mobile Health framework

There are a promising prediction about the tremendous role of mobile health in future [1], [3]–[5]. Silva M.C et al. [3] made extensive research no state of mobile health in 2015. Authors examined important scientific development in mHealth. They believe that mHealth has started reshaping the structure of the healthcare system. With the increasing popularity of mHealth, there are rising needs for evaluation criteria for mobile health frameworks, applications [1], [4], [5]. Sadegh et al. [5] emphasize the value of stakeholder analysis as the way of preventing "risks of delivering an unwanted health service". They took great framework examples such as Infrastructure Library(ITIL) and Control Objectives for Information and Related Technologies(COBIT) and evaluate existing mHealth framework on following dimensions: technical, service usability, strategic, organizational, social and legal. Sadegh et al. [5] created a framework for the evaluation of mobile health system with special characteristics such as stakeholders' consideration. On the other hand, Chan C.V and Kaufman D.R. [1] build mHealth system evaluation for developing countries. The crucial issue in developing countries ”the lack of the resources to develop technologies by themselves and emphasis is rather on the selection of the most suitable technology and the best means of technology transfer to rapidly achieve economic and social development goals” [1]. Authors create 3 level of consideration where at each level the particular needs of developing country have been chosen.

B. Stress detection and speech emotion recognition

There are very few works in the field of stress detection using only human speech as a reference. One of the most recent researches was conducted by Lu et al. [6]. The authors have done the work similar to ours proposed the framework for stress detection in unconstrained acoustic environments, which utilizes smartphones as a base platform.

In comparison with stress detection from speech, a lot more research work was done in the field of speech emotion recognition. The majority of speech emotion recognition architectures contain convolution neural network layers(CNN), recurrent neural network(RNN) or combination of them. [7]–[10]. The combination of CNN and RNN gives the ability to find out a crucial pattern in audio files(feature extraction) and feature classification. [7], [8]. The main importance in speech emotion recognition which kind of features to use. In paper Trigeorigis et al. [7] utilize the raw audio data as an input for the model. The authors use CNN for preprocessing of the audio data with the purpose of reducing the noise and emphasizing specific regions of the file. On the other hand, Lim et al. [8] convert the raw data into two-dimensional representation by application of Short Time Fourier Transform. After that, according to the paper the produced signal is fed to the models, where first of the layers was CNN. The similar method was used by Badshah et al. [9]. The spectrograms are generated out of the raw audio files from Berlin emotion dataset. After that, these spectrograms are passed to a CNN model. Yanfeng et al [10] focused on preprocessing stage when they offered new data handling algorithm based on imaging behavior of the retina and convex lens(DPARIP) in order to obtain the variety of spectrogram sizes. The architectures based on recurrent neural network, long-term short memory and deep convolutional neural networks was used to train model for feature classification. [7]–[10]. The Trigeorgis et al. [7] model based on LSTM achieved considerably better results than the state-of-the-art models of that time. Badshah et al. [9] model's outcomes show that the fresh CNN model delivers satisfactory outputs for most of the categories besides fear. Some part of their model consisted of the fine-tuned pre-trained AlexNet model. However, the results from the AlexNet model are not satisfactory. The best result was achieved by Lim et al [8] which models use CNN, LSTM and time distributed CNN.

III. Methodology

A. Feature Extraction

The first main phase in implementing a Deep Learning architecture to the problem of emotion recognition from the speech is feature extraction. Feature extraction plays a crucial role in the future success of potential machine learning model: an appropriate feature selection could lead to a better-trained model, while improper features would significantly hinder the training process [7]. For the process of feature extraction, we utilized Librosa audio library [11]. In this paper, we utilized Mel-frequency Cepstral Coefficients (MFCCs), spectral representations of a single recording, as the input data for our deep learning model. MFCCs are widely utilized in the field of sound classification and speech emotion recognition [12]. These features imitate to some extent the pattern of sound frequency reception intrinsic to a human.

In this work, we extract the features described above from a single sound file and stack them into the 1-dimensional array by taking the mean values along the time axis. This array is then fed to the model as an input data.

B. Proposed model

In our work, we utilized Convolutional Neural Network (CNN) for classification of emotion based on features extracted from a sound file. We have done a large number of experiments with different model configurations and class numbers (emotions). Below we describe the model with the best performance that we could be able to get. Our model is comprised of 1-dimensional convolutional layers combined with batch normalization and activation layers. The first layer of our CNN receives 64x1 number array as an input data. The initial layer is comprised of 256 filters with the kernel size of 5x5 and stride 1. After this, a batch normalization is applied, and its output is activated by Rectifier Linear Units layer (ReLU). Th next convolutional layer consists of 128 filters with the same kernel size and stride receives the output of a previous layer as an input. The output of this layer is also activated by ReLU, and then dropout with the rate of 0.1 is applied. Next, batch normalization is implemented, feeding its output to the max-pooling layer with a window size of 8. After that, 3 convolution layers with 128 filters of size 5x5 are located, two of them followed by ReLU activation layers and last followed by batch normalization, ReLU activation and dropout layer with the rate of 0.2. Final convolutional layer with the same parameters is followed by flattening layer. The output of the flattening layer is received by a fully connected layer with different number of units depending on the number of classes that are predicted. After this, batch normalization and softmax activation are applied. Our model utilizes RMSProp optimizer with the learning rate of 0.00001 and the decay rate of 1e-6.

IV. Dataset and Experiments

A. Dataset

The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) was chosen as the dataset for our model

TABLE I
CONVERSION OF 8 EMOTIONS INTO BINARY STRESS CLASSIFICATION

Emotion class	Stress Class
Happy	not stressed
Sad	stressed
Angry	stressed
Fearful	stressed
Calm	not stressed
Surprised	not stressed
Neutral	not stressed
Disgusted	not stressed

due to its great availability. The dataset contains audio and visual recordings of 12 male and 12 female actors pronouncing phrases in English with 8 different emotional expressions [13]. For our task, we utilized only audio part of the database by taking emotions for 8 different emotion classes and two genders. The emotion classes are as follows: sad, happy, angry, calm and fearful, surprised, neutral and disgusted. The overall number of samples was 1400.

For the classification task, we implemented a number of different models in parallel with the purpose of identification of the model with the best performance. Only the two models with the best accuracy are described below.

B. Experiments

1) Model with 4 classes: In the first approach described in this paper, we applied 4 class stress classification directly to the training and testing samples by eliminating multi-class emotion categorization procedure, therefore, this approach has four classes for prediction: female stressed, female not stressed, male stressed, and male not stressed. Table I illustrates the emotion conversion method utilized during the classification. Basically, the idea was to ease the process of classification for our model by decreasing the number of categories that it needs to recognize.

Subsequently, the corresponding model has a topology described in the previous section with 4 units in the final fully connected layer. The architecture of the model is illustrated in Fig. 1.

In order to train our model, we utilized 80% (1157) of samples from the dataset as the training set and 20% (283) of samples as the testing set. After 300 epochs our model performed on the testing set with the accuracy of 83.58%. This clearly illustrates that our model has learned to detect stressful emotions from human speech with reasonable correctness. Furthermore, we also tried to reduce the number of emotion classes with the expectation of growth of the accuracy. However, the accuracy did not increase but actually decreased.

2) Binary model: The next approach was to eliminate the gender of the speaker in order to increase the classification accuracy. Therefore, we have only two categories in this case: stressed, and not stressed. The setup of the model was exactly the same as for the previous one, except that the final fully connected layer had 2 units instead of 4 units, as in the previous case. This approach led us to a slight gain in accuracy by a few percents. The classification accuracy of the

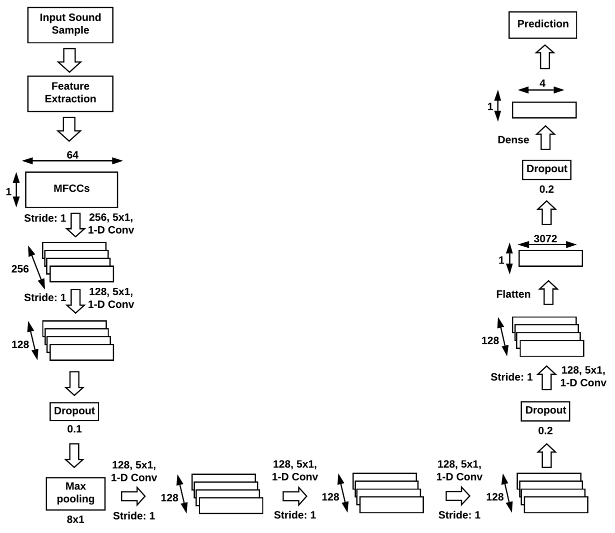


Fig. 1. The architecture of the 4-class speech stress detection model

model was 84.67%. Further experiments were stopped due to upcoming deadlines for field testing of the mobile application. The confusion matrix of the speech stress detection model on the base of 8 emotion classes is depicted in Fig. 2.

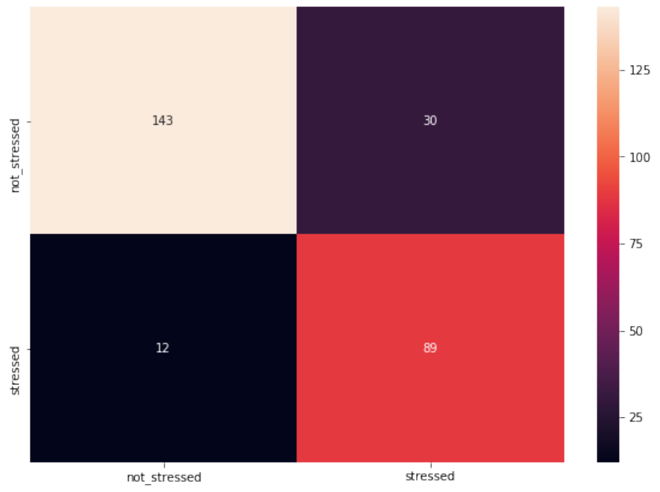


Fig. 2. Confusion matrix of the binary speech stress detection model

C. Analysis

Two models for speech emotion recognition with small accuracy differences were proposed by this paper. Both of the models have outperformed the accuracy of the stress classification in the work of Lu et al., where the authors implemented models that were personally pre-trained for each user [6]. Though the performance of our model is not flawless, it is sufficient for utilization in health monitoring application that provides only an additional diagnosis service.

D. Tuning the model

In order to achieve better results during the field testing of our application, we implemented personalized models similar to Lu et al. [6], which are trained to fit the particular user. The personalizing feature was obtained using the transfer learning technique with the last 4 layers of the network. Therefore, the first time when the result of the prediction is not correct according to the user's feedback, the user's individual clone of the general model is created and retrained using the features extracted from the user's recording that was misclassified. The retrained model is further used as the main model for the user's future predictions. This approach makes the model more adapted to the specific user, so it learns correctly to recognize the particular recording on average with 3-4 tries.

V. Server Side

A. Design

The general architecture of the system is described above (Fig. 6). System consists of server side which includes ML model and API which serves as communication entity between clients. Client side consists of client applications with certain priorities. These priorities define the importance and sequence of development. In this project mobile application has the highest priority and is designed and implemented first.

The server side is a part of a system that persists, manages and analyzes the data received from client applications and allow communication between them. Server side may also function as a major processing segment in the architecture of a client-server systems. In the fields, server side application that communicates through the web is defined as a web service. There exist plethora of web service architectures and many of them share some features. The most popular and well developed in the industry are Simple Object Access Protocol (SOAP) [14] and Representational State Transfer (REST) [15] web services. In the development of this system the REST architecture was used for its simplicity and scalability.

The server side was developed according to REST architecture in the form of Application Programming Interface or API. So that we would be able to integrate server side with different independent clients (Web, Desktop) of diverse platforms these applications, we selected REST API design for the server side. Client and server application communicate through HTTP protocol with messages represented as JSON strings in this design approach. This approach met requirements for this project with minimal effort and absence of any complexity.

B. Implementation

There exist different REST API frameworks developed in different languages. The list was narrowed to Express.JS [16] and Flask [17] by the primary requirements of the project. Both frameworks share the light-weight property, simple and modular structure. On the one hand, Express.JS was superior choice for the effortless communication with mobile client as mobile application was developed in React (see next section). On the other hand, Flask was written on Python and would provide straightforward integration with Machine Learning part of the system. Otherwise, one would have to provide an API for ML to communicate Express API and ML model written on Python. As this approach would significantly grow complexity of the system, Flask framework was chosen to implement server side of the system.

Flask is a lightweight open source [18] microframework for web applications under Python Web Server Gateway Interface (WSGI, PEP 3333) specification [19]. Framework allows express API development with any complexity and is expandable through modules if necessary. The representation diagram of an API is described in Fi. 7 and Fig. 8. These self-explanatory diagrams represent two primary endpoints of an API at current stage of development (other authentication and system specific endpoints are neglected for simplicity). The API was designed with the possibility of augmenting additional functionality in the future as the project is on a initial state now. As a result, API will be hosted on Linux Virtual Private Server (VPS) with dedicated IP address.

C. Database

Persistent data are one of the main requirements of this project as ML analysis requires chronological input and the storage of these inputs is essential part of the system. In fact, from one of the perspectives API is the interface for the database with some logical functionality. For this purpose, the database management functionality was assigned for the open source relational database management system PostgreSQL [20]. This responsibility was delegated to free Database as a Service (DaaS) in the Heroku platform to reduce load from the VPS and eliminate waste (boilerplate SQL server setup, server routes setup, etc) from development process. Fig. 9 illustrates Entity Relationship Diagram (ERD) for the system at current stage of development (again system specific tables are neglected for simplicity). The field 'mood' in the 'fdata' tables represents the output of the ML model and main focus at the current development process.

VI. Mobile application

Mobile application is primary client application of the system. There are two high market share operating systems, iOS and Android, and the aim is to develop mobile client as identical as possible for both platforms. For this purpose, React Native, a platform independent framework for building native mobile applications developed by Facebook, is utilized [21]. The framework provides effortless development of mobile applications abstracting user from platform specific language (Java/Kotlin fro Android and Swift for iOS). This property allows concurrent development of application for both platforms the result which can be mapped to both operating systems.

Mobile application of the our system is called mHealth. The screen shots of the application is provided in the Appendix. The mHealth app is essentially Create Read Update Delete application which maps CRUD methods into HTTP methods to provide data to the API. User interface of the application is designed to be similar to other fitness applications in the market to provide smooth and familiar UX. Apart from

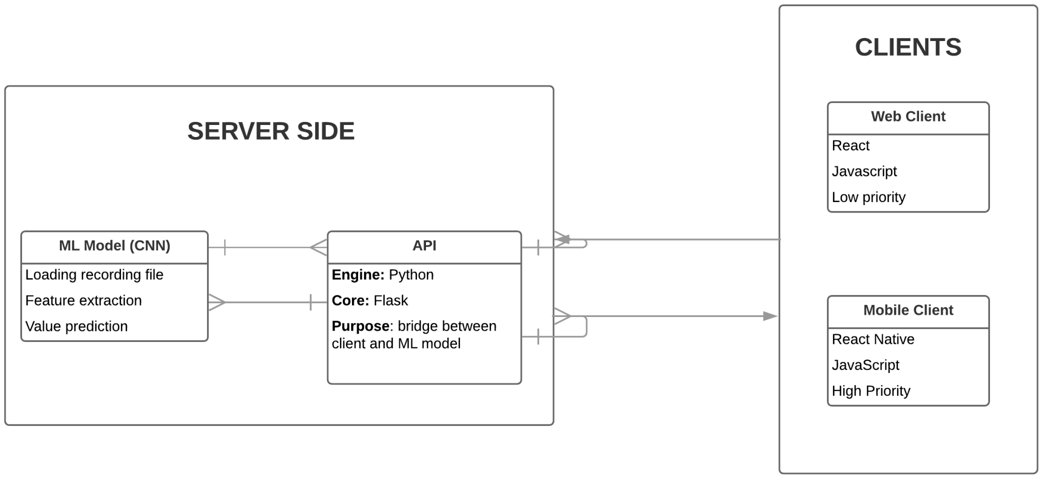
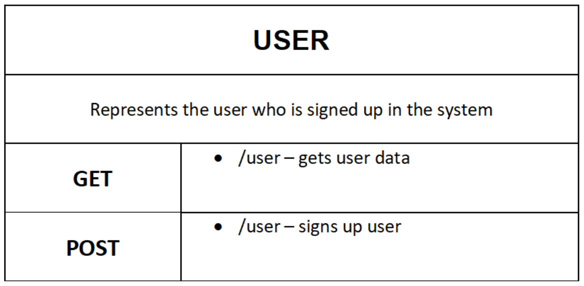
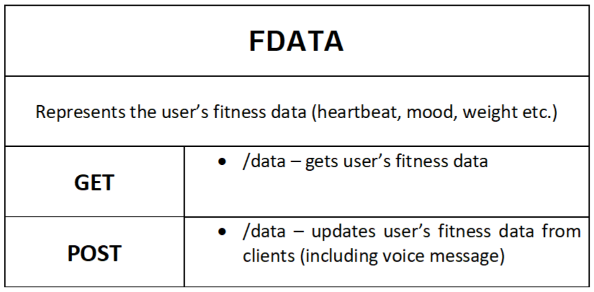


Fig. 3. General architecture of a system



USER	
Represents the user who is signed up in the system	
GET	• /user – gets user data
POST	• /user – signs up user

Fig. 4. API: User endpoints



FDATA	
Represents the user's fitness data (heartbeat, mood, weight etc.)	
GET	• /data – gets user's fitness data
POST	• /data – updates user's fitness data from clients (including voice message)

Fig. 5. API: Fitness data endpoints

retrieving data from API and visualizing it, application also has functionality to record user's voice and feed it to the ML model through the API.

At this stage of development, the data format of the voice recording is chosen to be the audio file. Basically, client sends audio file to the server for the feature extraction and receives result as a response. After this stage, application provides several solutions/suggestions to reduce the stress level. Furthermore, feedback option has been added to the functionality. In other words, user has capability to affect the statistics by providing if the prediction was true or false. It is crucial as the server side has a ability to adapt and train model according to the user's response. In the end, application allows user to adjust application for themselves.

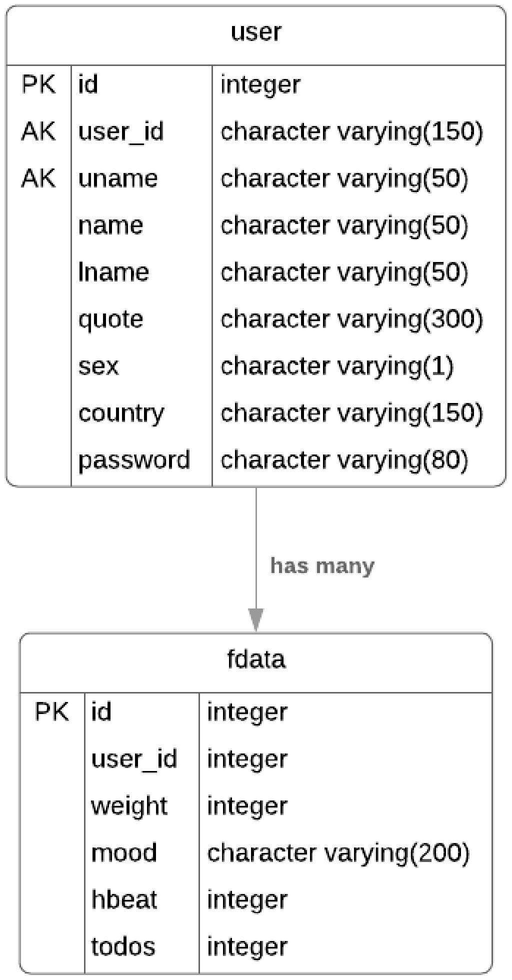
In addition, the other functionality is to track stress record events. In other words, application has two statics: daily and weekly statistics. This statistics represents the ration between stress detected recordings and not stressed recordings. Furthermore, user is provided by the last 10 results of recordings with exact time and date. We believe, this would encourage users to reach the streaks of not-stress days or weeks.

VII. Conclusion

To conclude, this paper proposes the mHealth application for real-time stress detection using human speech. Our deep learning model designed for this task is able to detect stress with the accuracy of 84.67% utilizing samples from the dataset with 8 different emotion classes. Additionally, each user of the application could have own specific stress detection machine learning model, which is additionally trained during the utilization of our mobile application by the user. Further work on the stress detection model involves testing other kinds of extracted sound features, different model architectures and adding noise filter for testing in noisy environments. With the creative way of using sensors in mobile devices, there is a huge potential to promote health care system. In the future, we plan to add other important health data like heart rate, sleep cycles, and diet control.

References

[1] C. V. Chan and D. R. Kaufman, "A technology selection framework for supporting delivery of patient-oriented health interventions in developing countries," *Journal of biomedical informatics*, vol. 43, no. 2, pp. 300–306, 2010.



user

PK	id	integer
AK	user_id	character varying(150)
AK	uname	character varying(50)
	name	character varying(50)
	lname	character varying(50)
	quote	character varying(300)
	sex	character varying(1)
	country	character varying(150)
	password	character varying(80)

fdata

PK	id	integer
	user_id	integer
	weight	integer
	mood	character varying(200)
	hbeat	integer
	todos	integer

Fig. 6. Database ERD

[2] Research2guidance. (2018) mhealth app developer economics 2017 - the state of the art ofmhealth app publishing. [Online]. Available: https://research2guidance.com/product/mhealth-economics-2017-current-status-and-future-trends-in-mobile-health

[3] B. M. Silva, J. J. Rodrigues, I. de la Torre Díez, M. López-Coronado, and K. Saleem, “Mobile-health: A review of current state in 2015,” *Journal of biomedical informatics*, vol. 56, pp. 265–272, 2015.

[4] C.-K. Kao and D. M. Liebovitz, “Consumer mobile health apps: Current state, barriers, and future directions,” *PM&R*, vol. 9, no. 5, pp. S106–S115, 2017.

[5] S. S. Sadegh, P. S. Khakshour, M. M. Sepehri, and V. Assadi, “A framework for m-health service development and success evaluation.” *International journal of medical informatics*, vol. 112, pp. 123–130, 2018.

[6] H. Lu, D. Frauendorfer, M. Rabbi, M. S. Mast, G. T. Chittaranjan, A. T. Campbell, D. Gatica-Perez, and T. Choudhury, “Stresssense: Detecting stress in unconstrained acoustic environments using smartphones,” in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. ACM, 2012, pp. 351–360.

[7] G. Trigeorgis, F. Ringeval, R. Brueckner, E. Marchi, M. A. Nicolaou, B. Schuller, and S. Zafeiriou, “Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network,” in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5200–5204.

[8] W. Lim, D. Jang, and T. Lee, “Speech emotion recognition using convolutional and recurrent neural networks,” in *Signal and information processing association annual summit and conference (APSIPA), 2016 Asia-Pacific*. IEEE, 2016, pp. 1–4.

[9] A. M. Badshah, J. Ahmad, N. Rahim, and S. W. Baik, “Speech emotion recognition from spectrograms with deep convolutional neural network,” in *Platform Technology and Service (PlatCon), 2017 International Conference on*. IEEE, 2017, pp. 1–5.

[10] Y. N. Z. H. H. T. Yangeng Niu, Dongsheng Zou, “Improvement on speech emotion recognition based on deep convolutional neural networks,” in *Proceedings of the 2018 International Conference on Computing and Artificial Intelligence*. ACM, 2018, pp. 13–18.

[11] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, “librosa: Audio and music signal analysis in python,” in *Proceedings of the 14th python in science conference*, 2015, pp. 18–25.

[12] S. S. Stevens, J. Volkmann, and E. B. Newman, “A scale for the measurement of the psychological magnitude pitch,” *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, 1937.

[13] S. R. Livingstone and F. A. Russo, “The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english,” *PloS one*, vol. 13, no. 5, p. e0196391, 2018.

[14] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, and D. Winer, “Simple object access protocol (soap) 1.1,” 2000.

[15] R. T. Fielding and R. N. Taylor, *Architectural styles and the design of network-based software architectures*. University of California, Irvine Irvine, USA, 2000, vol. 7.

[16] A. Mardan, *Express. js Guide: The Comprehensive Book on Express. js*. Azat Mardan, 2014.

[17] M. Grinberg, *Flask web development: developing web applications with python*. ” O’Reilly Media, Inc.”, 2018.

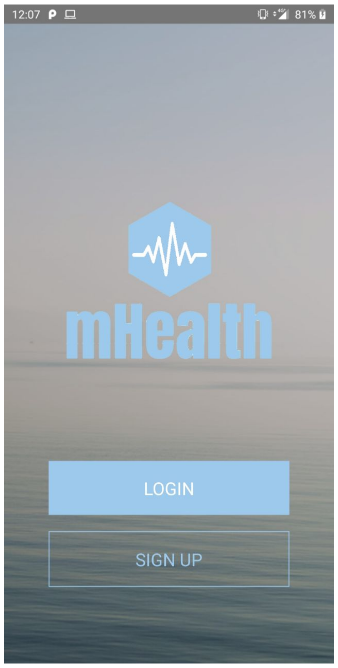
[18] A. Ronacher, “Welcomeflask (a python microframework),” *URL: http://flask. pocoo. org/*, p. 38, 2010.

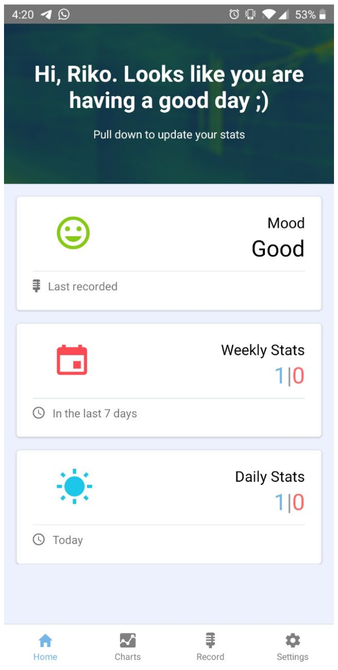
[19] P. Eby, “Python web server gateway interface v1. 0.1 (pep 3333), sep 2010,” *URL https://www. python. org/dev/peps/pep-3333*.

[20] N. Matthew and R. Stones, *Beginning Databases with PostgreSQL*. Apress, 2005.

[21] B. Eisenman, *Learning React Native: Building Native Mobile Apps with JavaScript*. ” O’Reilly Media, Inc.”, 2015.

APPENDIX



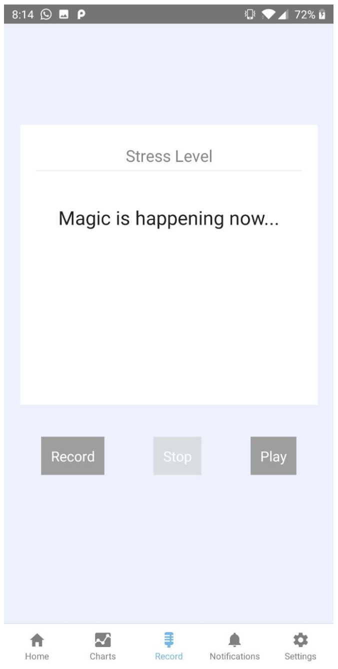

Hi, Riko. Looks like you are having a good day ;)

Pull down to update your stats

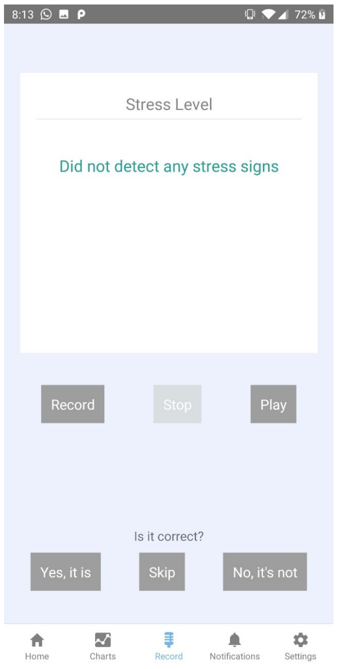
Mood: Good — Last recorded

Weekly Stats: 1|0 — In the last 7 days

Daily Stats: 1|0 — Today

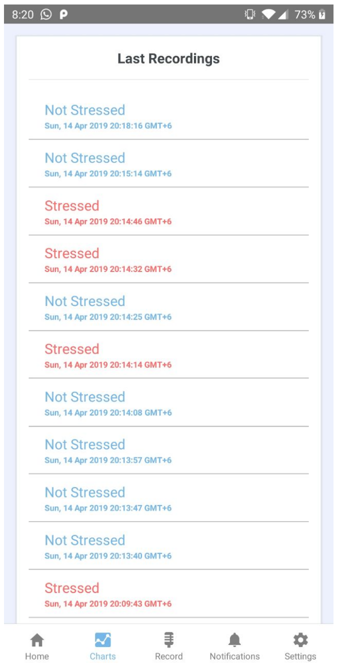

Stress Level

Magic is happening now...


Stress Level

Did not detect any stress signs

Is it correct?


Last Recordings

Not Stressed
Sun, 14 Apr 2019 20:18:16 GMT+6

Not Stressed
Sun, 14 Apr 2019 20:15:14 GMT+6

Stressed
Sun, 14 Apr 2019 20:14:46 GMT+6

Stressed
Sun, 14 Apr 2019 20:14:32 GMT+6

Not Stressed
Sun, 14 Apr 2019 20:14:25 GMT+6

Stressed
Sun, 14 Apr 2019 20:14:14 GMT+6

Not Stressed
Sun, 14 Apr 2019 20:14:08 GMT+6

Not Stressed
Sun, 14 Apr 2019 20:13:57 GMT+6

Not Stressed
Sun, 14 Apr 2019 20:13:47 GMT+6

Not Stressed
Sun, 14 Apr 2019 20:13:40 GMT+6

Stressed
Sun, 14 Apr 2019 20:09:43 GMT+6