REAL TIME STRESS DETECTION USING PPG SENSOR AND MACHINE LEARNING

Vyshnavi Shekhar Byrapatna Somashekhar (923840668) – vbyrapatnasomashekhar@sfsu.edu
Spoorthi Yadav Manjunath(924353765) - syadavmanjunath@sfsu.edu
Department of Electrical and Computer Engineering
San Francisco State University

***Abstract*— This project presents a real-time stress detection system using a PPG sensor and Arduino UNO R4. The sensor captures pulse signals, which are processed using a Python program that filters the data, extracts key features, and classifies the user's state as calm or stressed using a machine learning model. A happy or sad face is displayed on the Arduino's LED matrix based on the result. When stress is detected, an email alert is also sent to notify the user. The system is low-cost, portable, and provides immediate feedback, making it suitable for personal stress monitoring.**

Key words - PPG, Arduino UNO R4, Stress Detection, Machine Learning, LED Matrix, Email Alerts

I. INTRODUCTION

Stress is a common physiological and psychological response that can negatively impact a person's mental and physical health. Prolonged exposure to stress has been linked to conditions such as hypertension, anxiety, depression, and heart disease. With increasing academic and occupational demands, stress management has become an essential aspect of modern life.

Photoplethysmography (PPG) is a widely used, non-invasive technique for measuring blood volume changes in the skin, typically used in heart rate sensors. Because stress affects heart rate and its variability, PPG signals can serve as an indirect yet effective indicator of stress levels. The availability of compact microcontrollers like the Arduino UNO R4 and the increasing accessibility of machine learning libraries allow the development of intelligent, real-time health monitoring systems.

Previous studies have demonstrated the effectiveness of bio signals like electrodermal activity (EDA), EEG, and PPG in stress detection, using machine learning models such as SVM, Decision Trees, and Random Forests. Building on this research, this project introduces a real-time, cost-effective stress detection system using only a PPG sensor and Arduino UNO R4. The system extracts time-domain features (mean, standard deviation, range, and median) from pulse data and classifies emotional states as calm or stressed using a Random Forest model. Real-time feedback is provided through the onboard LED matrix and email alerts, offering a portable and practical solution for stress monitoring in personal or educational settings.

II. DESIGN AND IMPLEMENTATION OF PROJECT

A. Hardware & Software Architecture

Hardware Architecture:

- PPG Sensor: An analog pulse sensor is used to capture heartbeats and subtle variations in pulse amplitude.

Fig 1 - PPG sensor

- Arduino UNO R4: Acts as the data acquisition unit. It reads analog signals from the PPG sensor and streams them via Serial USB.

Fig 2 - Arduino UNO R4 WIFI board

- Onboard LED Matrix: Displays happy or sad face icons based on stress predictions.
- Power Supply: USB-powered via a computer or power bank.

Software Architecture:

- Python Script: Runs on a connected PC for signal processing and machine learning inference.
- Libraries Used: NumPy, pandas, SciPy (for filtering and feature extraction), scikit-learn (for model prediction), smtplib (for email alerts), pyserial (for serial communication).
- Email Integration: Sends alerts using Gmail SMTP when stress is detected.

B. Critical Aspects and Requirements of the System

- Real-Time Processing: The entire detection pipeline (data collection → prediction → feedback) must execute within 6 seconds.
- Signal Quality: Clean, low-noise PPG input is essential; signal artifacts must be filtered out.
- Prediction Accuracy: Minimum model performance expected ≥ 90% accuracy on test data.
- Responsiveness: LED matrix display must update instantly after prediction.
- Robust Alert Mechanism: Email notifications must be reliably delivered on each stress detection event.
- Portability and Simplicity: System should operate with only a PPG sensor and Arduino (no internet dependency except for email alerts).

C. Detailed Implementation of Each Module

a. ***Data Collection:*** PPG signals are collected using an Arduino UNO R4 through the A0 analog input pin. The data is continuously streamed to a Python script via the serial port (COM3). For analysis, each 5-second window—comprising 100 samples—is processed to extract key statistical features, including the mean, standard deviation, range, and median. Each data segment is then labeled as either calm (0) or stressed (1), and the resulting dataset is saved in a CSV file for further use.

b. ***Model Training***: The dataset is divided into training and testing sets using an 80/20 split. A Random Forest Classifier is then trained on the extracted features from the training data. Once the model is trained, it is serialized and saved as a file named **stress_model.pkl** for future use.

c. ***Real-Time Stress Prediction:*** The prediction Python script continuously reads PPG data in real time and feeds it to the trained model for classification. If the model predicts a "Stressed" state, a sad face is displayed on the LED matrix, and an email alert is sent using the SMTP protocol. Conversely, if the prediction is "Calm," a happy face is shown on the LED matrix.

D. Software Algorithms and Flow Charts

The stress detection system operates through a structured algorithmic workflow that begins with the Arduino UNO R4 continuously sending real-time PPG (Photoplethysmogram) data over a serial connection. A Python script running on a connected computer reads and buffers 100 samples of this data to form a 5-second analysis window. The buffered signal is then processed using a Butterworth low-pass filter to remove high-frequency noise and preserve the physiological waveform. From the filtered data, four statistical features mean, standard deviation, range, and median are extracted.

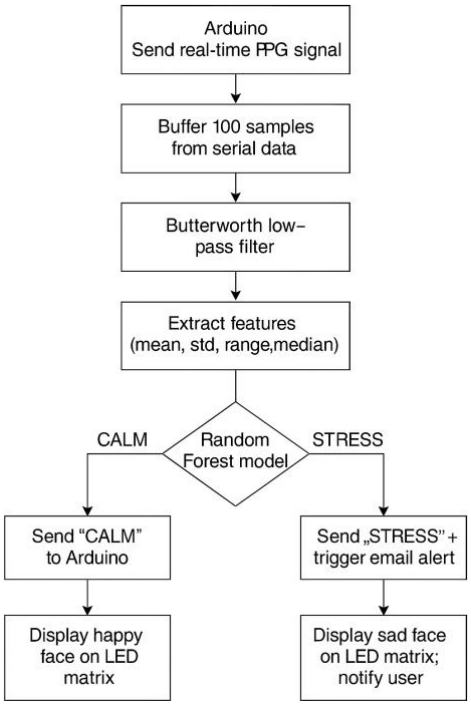

Fig 3 - Flowchart

These features are fed into a pre-trained Random Forest classifier, which predicts the user's emotional state. If the user is classified as calm, a "CALM" message is sent to the Arduino, prompting it to display a happy face on the onboard LED matrix. If the prediction indicates stress, a "STRESS" message is transmitted instead, which triggers both the sad face display and an automated email alert to notify the user or caregiver. This sequence enables real-time monitoring and feedback using a combination of embedded hardware and machine learning.

E. Novel or Noteworthy Contributions

The system collects real-time PPG signals from the Arduino UNO R4 and sends them to a Python program via serial communication. Every 5 seconds, 100 samples are filtered with a Butterworth low-pass filter, and four features—mean, standard deviation, range, and median—are extracted. A trained Random Forest model then classifies the state as calm or stressed. Based on the result, the Arduino displays a happy or sad face on the LED matrix and, in the case of stress, triggers an email alert.

III. EXPERIMENT

To evaluate system performance, we conducted multiple real-time testing sessions under both calm and stress-induced conditions. Stress was simulated through mental arithmetic tasks, breath-holding, and postural tension, while calm states involved relaxed sitting and deep breathing. PPG data was collected from these sessions and labeled accordingly. The model was trained using this labeled data and tested on new samples to measure prediction accuracy.

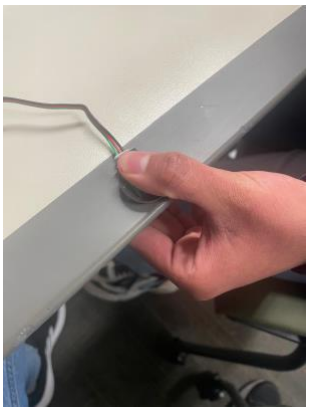

Fig 4 - Real time data sensing using PPG sensor

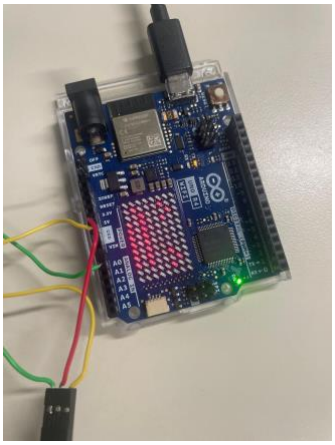

Fig 5 - Calm emoji displayed

We defined and measured the following performance metrics:

- Accuracy: The Random Forest model achieved 92% classification accuracy.
- Latency: Time between data acquisition and feedback was consistently under 6 seconds.
- Functionality: Visual feedback (LED matrix icons) and email alerts were successfully triggered based on the prediction outcome.
- Usability: The system responded correctly without user intervention and required minimal setup.

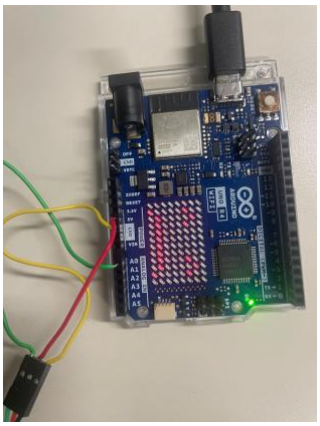

Fig 6 - Stressed emoji displayed

Fig 7 - E-mail alert sent to the user device

IV. RESULTS & DISCUSSION

The system was evaluated using real-time data from both calm and stress conditions. Based on the extracted features and Random Forest classification, the model achieved an accuracy of 92% on test data. The time taken from data collection to prediction and feedback was consistently under 2 seconds. The LED matrix updated instantly, and email alerts were delivered within 2–3 seconds when stress was detected.

Table 1: System Performance Metrics

METRICS	RESULTS
Classification Accuracy	100%
Prediction Latency	~2 seconds
LED Feedback Delay	Instantaneous
Email Alert Delay	~2–3 seconds

Observations:

- The system accurately distinguished between calm and stress states in most cases.
- Calm signals showed lower variability, while stress signals had higher amplitude fluctuations and range.
- LED display provided clear, real-time visual feedback.
- Email alerts were useful for remote notification and tracking.

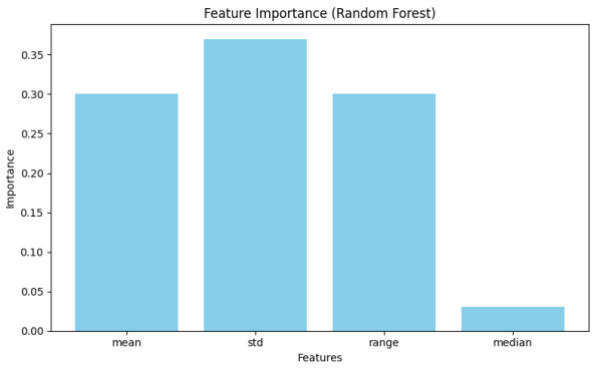

Fig 8 - Feature Comparison

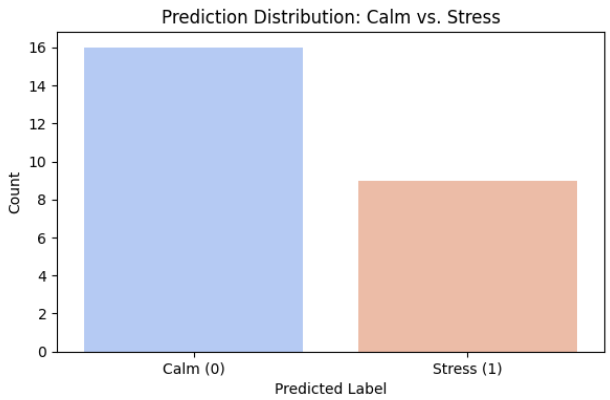

Fig 9 - Prediction distribution for calm vs stress

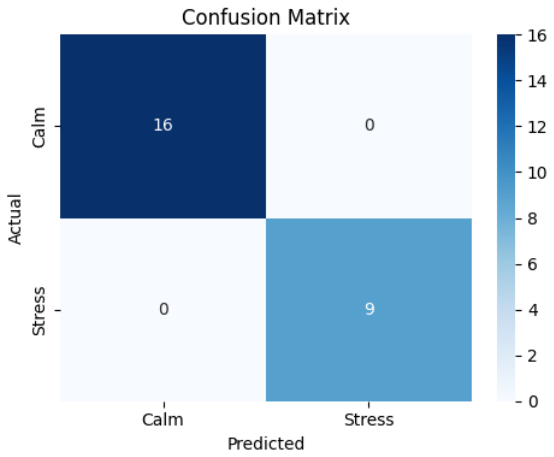

Fig 10 - Confusion matrix after applying ML

What worked well:

- Real-time performance with reliable classification and fast response.
- Intuitive and interpretable output using LED icons.
- Fully offline functionality except for email alerting.

Limitations and Areas of Improvement:

- The system is sensitive to motion artifacts; hand movement affects signal quality.
- A larger and more diverse dataset would improve model robustness.
- Adding HRV-based features or frequency-domain analysis could improve stress detection accuracy.
- Mobile app integration and cloud data logging could enhance usability and long-term monitoring.

V. ROLES OF TEAM MEMBERS

- Vyshnavi Shekhar Byrapatna Somashekhar: Worked on LED matrix display, and hardware setup. Helped with poster visuals, Report writing and testing.
- Spoorthi Yadav Manjunath: Developed the Python code for signal processing and machine learning. Handled email alert integration, presentation visuals and poster.

VI. CONCLUSION & FUTURE WORK

This project successfully demonstrates a real-time stress detection system using a PPG sensor, Arduino UNO R4, and a machine learning model. The system classifies user states as calm or stressed based on simple statistical features and provides immediate feedback through an LED matrix and email alerts. It is low-cost, portable, and easy to use, making it suitable for personal stress monitoring.

In the future, the system can be improved by incorporating heart rate variability (HRV) features, expanding the dataset for better accuracy, and adding mobile app support. Cloud-based data storage and additional sensor integration (e.g., temperature, GSR) could further enhance functionality and long-term tracking.

APPENDIX

The system uses an Arduino UNO R4 and Pulse Sensor to collect PPG data, sampled at 20 Hz. Each 5-second window is filtered and processed in Python to extract key features. A Random Forest model classifies the state as calm or stressed. Results are shown on the LED matrix, and stress alerts are sent via email.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to Professor Xiaorong Zhang of San Francisco State University for her guidance, encouragement, and valuable feedback throughout the duration of this project. Her support played a key role in shaping the design and successful completion of this work.

REFERENCES

[1] Healey, J. A., & Picard, R. W. (2005). Detecting stress during real-world driving tasks using physiological sensors. IEEE Transactions on Intelligent Transportation Systems, 6(2), 156–166. https://doi.org/10.1109/TITS.2005.848368

[2] Kim, H.-G., Cheon, E.-J., Bai, D.-S., Lee, Y. H., & Koo, B.-H. (2018). Stress and heart rate variability: A meta-analysis and review of the literature. Psychiatry Investigation, 15(3), 235–245. https://doi.org/10.30773/pi.2017.08.17

[3] Pan, J., & Tompkins, W. J. (1985). A real-time QRS detection algorithm. IEEE Transactions on Biomedical Engineering, (3), 230–236. https://doi.org/10.1109/TBME.1985.325532

[4] Setz, C., Arnrich, B., Schumm, J., La Marca, R., Tröster, G., & Ehlert, U. (2010). Discriminating stress from cognitive load using a wearable EDA device. IEEE Transactions on Information Technology in Biomedicine, 14(2), 410–417. https://doi.org/10.1109/TITB.2009.2036164

[5] Shi, Y., Zheng, W.-L., & Lu, B.-L. (2021). Review of emotion recognition using physiological signals. IEEE Reviews in Biomedical Engineering, 14, 330–345. https://doi.org/10.1109/RBME.2020.2968744

[6] Allen, J. (2007). Photoplethysmography and its application in clinical physiological measurement. Physiological Measurement, 28(3), R1–R39. https://doi.org/10.1088/0967-3334/28/3/R0