Univerzitet u Sarajevu
Elektrotehnički fakultet
Sarajevo

Final Project Report

Stress-level Management via Smartwatch Monitoring

Course: Machine Learning: Supervised Learning

Professor: Prof. Dr. Amila Akagić

Students: Emin Hadžiabdić, Muhammed Pašić, Armin Memišević

Academic Year: 2025/26

Student ID: 19960, 20109, 20016

Sarajevo, January 18th 2026

Table of Contents:

1 Project Summary and Objectives..... 2
2 The Machine Learning Pipeline..... 3
3 Model Development and Evaluation..... 4
4 Engineering and Implementation..... 5
4.1 Wearable Development..... 5
4.2 Backend and Cloud Infrastructure..... 6
4.3 Web Interface and Survey..... 6
5.1 Optimization and Privacy by Design..... 7
5.2 Other Ethical Safeguards..... 7
5.3 Commercialization Potential..... 8
6 Reflections and Future Work..... 8
6.1 Challenges Overcome..... 8
6.2 Future Roadmap..... 8
7 Conclusion..... 9

1 Project Summary and Objectives

The objective of this project was to create and develop a high-fidelity stress detection system, which is capable of running locally on a wearable device, mainly a smartwatch. Unlike cloud-based solutions, we prioritized privacy, low latency and battery efficiency in this project.

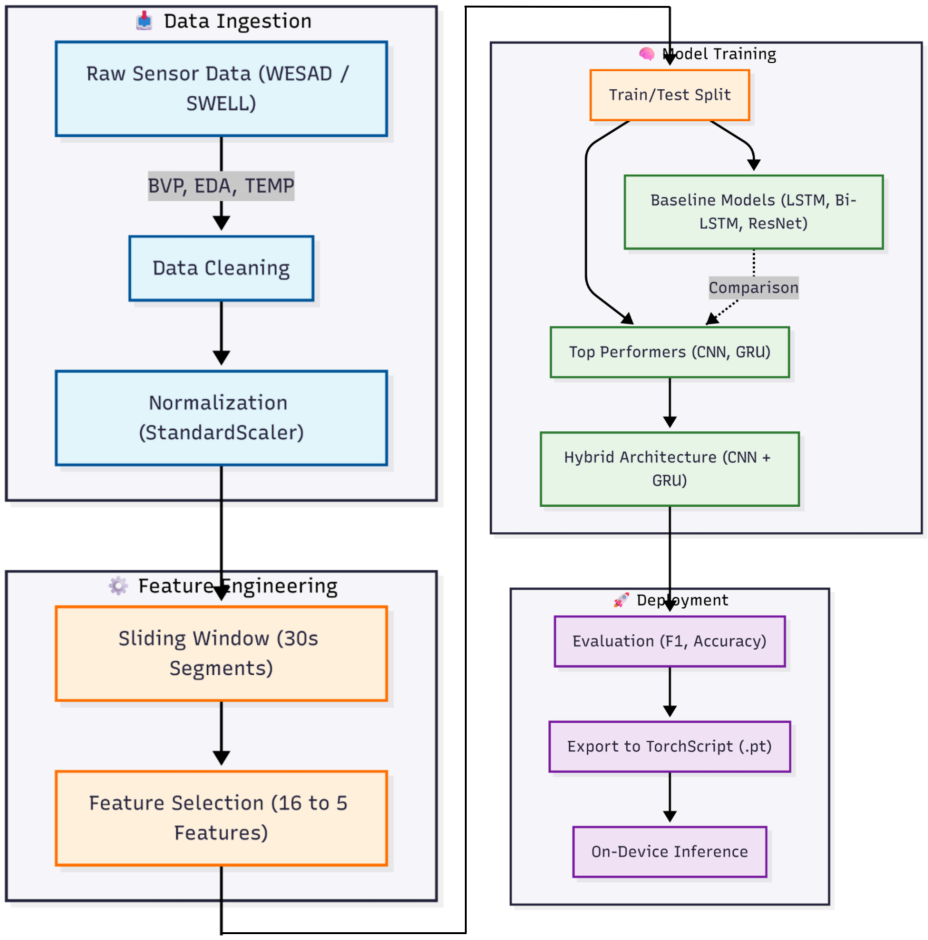
By having utilized a hybrid CNN-GRU architecture, our system achieves a high F1 score of 0.99 and it ensures privacy through on-device inference. Included in the ecosystem are a native application, a Firebase-backed real-time dashboard and a scenario-aware web survey. We decided to implement the web-based survey because of the recognition of the people who do not have access to any wearable hardware.

Our goal is to prove that complex deep learning can be implemented and used both ethically and efficiently in the space of wearable devices without sacrificing any diagnostic accuracy.

2 The Machine Learning Pipeline

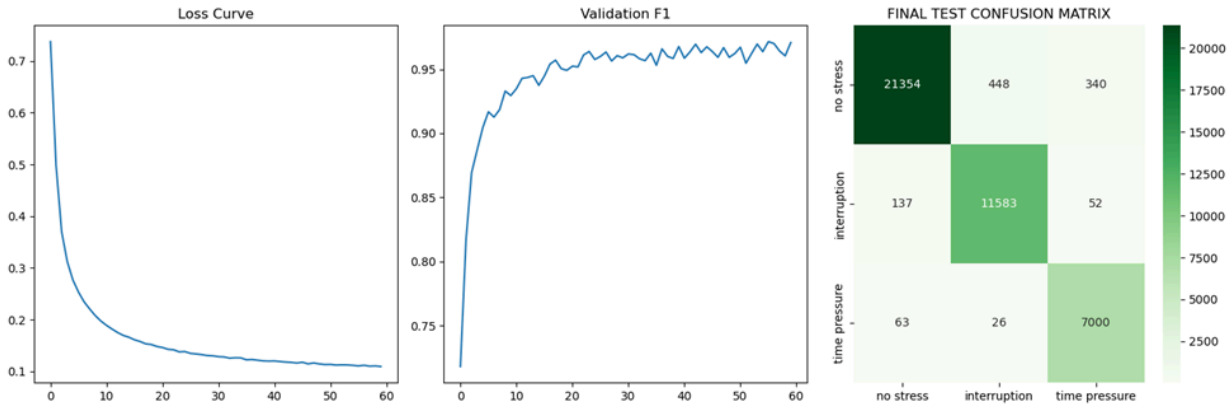
The development followed a rigorous ML pipeline designed for scalable stress monitoring.

- Raw sensor data was sourced from the SWELL and following this comprehensive data cleaning and normalization was done.
- To optimize for smartwatch hardware, we utilized 30s sliding windows and did aggressive feature selection. We reduced the input space from 16 to 5 critical features.
- In the end we ended up comparing the baseline models (LSTM, ResNET) and top-performing architectures, and ending by exporting to TorchScript for on-device inference.



3 Model Development and Evaluation

For us to find the most optimal architecture, we conducted a sort of tournament to compare several deep learning candidates using a subject-wise split to ensure generalizability in the real world. We compared 8 architectures including baseline LSTM, ResNET, pure CNN and pure GRU.



The problem we had was with the overconfidence of the pure GRU models in the reliability diagrams, which would potentially lead to false alarms, and pure CNNs lacked the temporal memory which we needed to track stress recovery. To solve this in the end we picked a hybrid CNN-GRU architecture. The CNN-GRU hybrid was selected for its great precision and mobile efficiency. CNN filters raw signal noise, while GRU has memory of previous physiological states. The hybrid model also fixes the overconfidence issues found in purse GRU models, which ensures that the probability scores are reliable for health monitoring. The result we got meant that with this hybrid approach we got the absolute best of both sides.

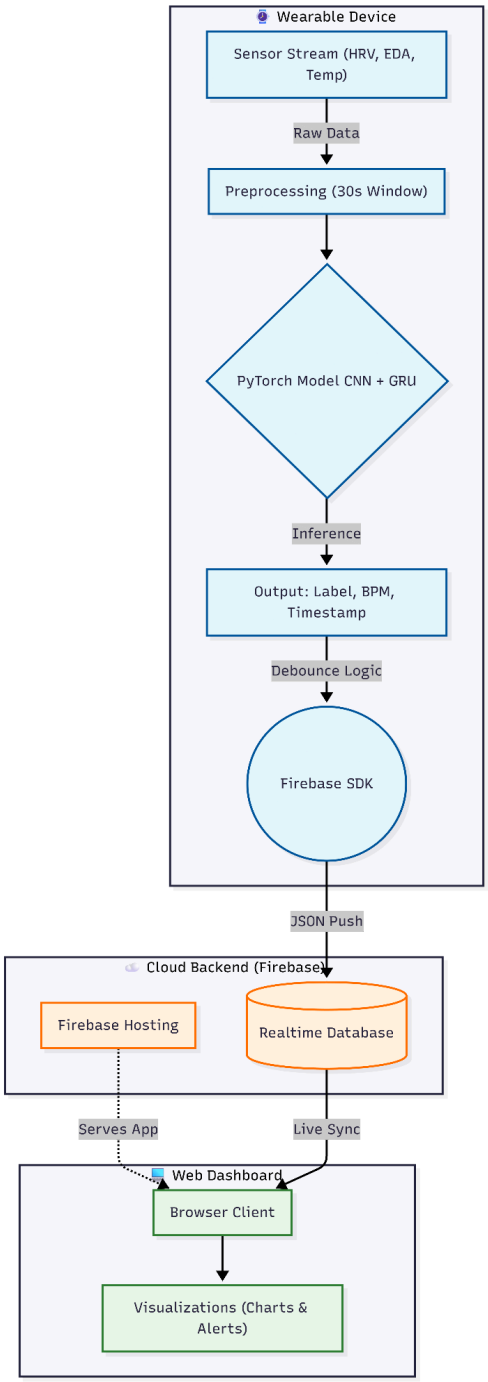
Metric	Score	Interpretation
Precision	0.99	When the watch alerts "Stress," it is almost always correct.
Recall	0.98	The model catches 98% of all actual stress events.
F1-Score	0.99	Excellent balance across all three conditions.

4 Engineering and Implementation

4.1 Wearable Development

The development of the smartwatch component underwent several iterations until it reached stability:

- Early development phases explored Tizen OS. We simulated watch environments in Android Studio; however, technical limitations such as older versions of Tizen led to a final implementation using Native Kotlin for Wear OS.
- The application was built successfully to interface with the smartwatch's hardware sensors which stream data into a specialized processing pipeline.
- To handle outliers and real-world artifacts, we designed the app as such to have a BPM cap (40-180 BPM). This ensures that any heart rate outlier which can be caused by sensor movement doesn't corrupt the model's input.
- The app measures and buffers 30 BPM samples so the model can determine if the user is interrupted, stressed or relaxed.



4.2 Backend and Cloud Infrastructure

- For hosting, we have a centralized server which hosts the model, serving it through a specialized API. The backend accepts GET requests from the connected interfaces which in turn allows the system to host and update the model efficiently.
- The application is synchronized with a Firebase Real-Time Database. It pushes a data packet containing: BPM, Raw Data, Timestamp and Prediction Label.
- Our web dashboard functions as a listener. The UI updates automatically as the dashboard is connected to the Firebase stream. When the user enters a certain stress state, that change pops up on the dashboard without the need for refreshing the page.

4.3 Web Interface and Survey

With the app we also developed a web-based interface to allow users to interact with the backend logic. This interface communicates directly with the centralized server to provide immediate feedback on the user's stress state at the moment. It also includes a scenario-aware algorithm that adjusts based on the user provided survey context.

5 Ethical Consideration and Commercialization

5.1 Optimization and Privacy by Design

One core achievement of our project is the export of the model as a .pt file. By having the deployment stage of our machine learning pipeline directly moved onto the hardware, we achieve a level of privacy which is unheard of with cloud-based inference. By doing this we achieve:

- Privacy - sensitive physiological data is processed locally and the raw heart rate data does not need to be sent to the cloud for classification. This ensures that the most sensitive biometric data will always remain on the hardware.
- Latency - users receive their alerts instantly because the system does not rely on external network speeds for its logic. This is critical for real-time monitoring.

5.2 Other Ethical Safeguards

Deployment of our applications is guided by some ethical considerations that should be looked over:

- Users of the app must be informed that our app cannot distinguish between negative stress (distress) and positive excitement (eustress, eg. exercise).
- Real-time stress readings are carefully designed to avoid inducing additional anxiety with the alerts, which would create a feedback loop that worsens the physiological state of the user.
- While the main processing is local via the .pt model, users must explicitly be informed that classified labels (Relaxed/Stressed), BPM, and timestamps are streamed to a remote dashboard via Firebase. This is for monitoring purposes.

5.3 Commercialization Potential

The system's architecture opens significant market opportunities. Some of these opportunities are:

- For corporate environments to monitor burnout and suggest breaks based on real-time data.
- Because a wearable device is not required for use of our services, because of the web survey, this is a major commercial advantage. This is because it makes the platform accessible to a wider demographic, regardless of their access to wearable technology.

6 Reflections and Future Work

6.1 Challenges Overcome

Transitioning from failed attempts at OS platforms to a working Kotlin application really proved the necessity of native development. Also, the implementation of BPM capping was a critical logical layer that was added to bridge the gap between lab-trained models and real-world noisy sensors.

6.2 Future Roadmap

We are not done with this application and we deem it holds great potential going forward. Some future work we would like to get started on would be:

- Merging the Wear OS dashboard and the web survey platform into a single unified ecosystem
- Refining the model further to reduce the initial 30-sample buffering delay.

7 Conclusion

This project successfully moved from a data-science experiment to a working prototype. The completion of this project really represents an important milestone in bridging the gap between deep learning research and practical, more importantly privacy-centric, wearable technology.

By prioritizing the On-Device Inference, our system addresses the primary ethical hurdle of the wearable industry, and that is data sovereignty. The conversion of the hybrid CNN-GRU model to a .pt format is what allows the smartwatch to act as an independent intelligent node and because the biometric data gets processed locally, the user's sensitive information never leaves the watch, while zero-latency execution guarantees stress alerts are processed instantly.

Moreover, the dual-pathway approach (combining automated wearable tracking and a scenario aware web survey) guarantees that the platform is accessible to all users.

In the end, our project proves that complex AI models can be optimized without sacrificing accuracy, and setting a new standard for ethical physiological monitoring in real time.