**Final Project Proposal**

**Topic: A Human Activity Recognition-Based Safety System**

**Pattern Recognition and Machine Learning CSC 588 - U17**

Lalitha Priya Bijja – 101168225[1]  
Department of Computer Science  
University of South Dakota

**I. Introduction**

**Research question**

How can human activity recognition (HAR) using smartphone sensor data be leveraged to develop an automated safety system for women, which detects abnormal activities and triggers real-time SOS alerts with location tracking, without the need for manual intervention?

**II. Background and Motivation**

In recent years, **Human Activity Recognition (HAR)** using smartphone sensors has gained significant attention due to its wide-ranging applications in healthcare, security, fitness tracking, and human-computer interaction. Improving personal safety is one of the most important uses, particularly for women and other vulnerable groups. As smartphone usage increases, using built-in sensors like gyroscopes and accelerometers to track physical activity in real time offers a practical and affordable way to ensure safety.

Several studies have successfully applied machine learning models to recognize activities like walking, running, standing, and sitting. For example, the **UCI HAR dataset** has been widely used to train classifiers on basic human movements. Models such as **Random Forests**, **Support Vector Machines (SVMs)**, and **Convolutional Neural Networks (CNNs)** have been used to classify human activities based on smartphone data with high accuracy. However, there has been limited research on using HAR systems for **real-time safety applications** involving emergency scenarios like sudden falls or erratic movements.

Traditional safety apps, particularly those designed for women, rely on **manual activation** of emergency features (e.g., pressing a panic button or sending a distress signal). These apps are effective but have a major limitation: in high-stress or incapacitating situations, the individual may not be able to initiate the alert. There is a critical need for an **automated system** that can recognize abnormal or distress-related activities and trigger an SOS without user intervention

A major challenge in this domain is the **detection of abnormal activities**, such as sudden falls or erratic movements, which could indicate distress or danger. Traditional HAR models classify routine activities (e.g., walking, sitting, running), but they often lack the ability to recognize and respond to **unexpected, emergency behaviors**. Women, in particular, face the risk of harassment or accidents in public spaces, and there is a pressing need for systems that can automatically detect such situations and **trigger an emergency response**.

**Justification for the Chosen Problem:**

Safety concerns, particularly for women in public spaces, remain a pressing global issue. Women may face harassment, accidents, or other forms of physical threats, making it crucial to develop proactive safety mechanisms. While many apps offer location tracking or panic buttons, these solutions fail when users are unable to manually activate them. Incorporating automated abnormal activity detection into existing systems can bridge this gap.

The proposed solution leverages the same smartphone sensors used in HAR to identify abnormal behaviors like falls, sudden sprints, or erratic movements, which may indicate distress. By doing so, the system can trigger an SOS alert, sharing real-time location data with emergency contacts, thus providing timely assistance.

This approach not only addresses a critical gap in women’s safety but also has broader applications for vulnerable populations, such as the elderly or individuals with medical conditions who may experience falls or emergencies. With the advancement in deep learning and real-time data processing, developing a reliable, automated system for safety alerts is now more feasible than ever.

**Literature Review**

Previous solutions for human activity recognition focused primarily on **fitness tracking** and **health monitoring**, where machine learning algorithms were applied to predict activities like walking, running, or sitting based on sensor data. In some safety applications, apps used GPS tracking or user-triggered emergency buttons to alert contacts, but few have combined **HAR** with real-time, sensor-based **automated safety responses**.

In the research domain, **machine learning models** (such as Random Forests, Support Vector Machines, and LSTMs) have been employed to classify user movements into predefined categories. More recently, **deep learning approaches**, particularly **Recurrent Neural Networks (RNNs)** and **Long Short-Term Memory (LSTM)** networks, have shown promise in processing time-series sensor data to achieve more accurate HAR.

However, few projects have integrated HAR with **safety systems**, and there is limited research on using abnormal activity detection for **real-time safety alerts** tailored specifically for women.

**III. Objectives**

This project aims to extend **HAR** with a **safety system** for women, capable of detecting abnormal activities such as **falls, sudden running, or erratic movement patterns**. The system will automatically trigger an **SOS alert** and share the user's real-time location with emergency contacts. By leveraging **LSTM networks** for HAR, the system will monitor regular activities while flagging potentially dangerous ones.

Key aspects of the solution include:

1. Collecting and preprocessing smartphone sensor data (accelerometer and gyroscope).
2. Building a robust LSTM-based HAR model for real-time activity classification.
3. Implementing a detection mechanism for abnormal behavior, triggering a safety protocol when such behaviors are detected.
4. Sending SOS messages with real-time GPS data to predefined contacts when danger is detected.

**Dataset: UCI Human Activity Recognition Dataset:**

* **Source**: [UCI HAR Dataset](https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones)

**Description**: The experiments involved a group of 30 volunteers aged between 19 and 48 years. Each participant performed six activities: walking, walking upstairs, walking downstairs, sitting, standing, and laying. These activities were recorded using a Samsung Galaxy S II smartphone worn at the waist. The phone's embedded accelerometer and gyroscope captured 3-axis linear acceleration and angular velocity at a rate of 50Hz. To ensure accurate labeling, the activities were video-recorded.

The collected dataset was divided randomly into training and testing sets, with 70% of the participants contributing to the training data and the remaining 30% forming the test data.

Before model training, the sensor signals underwent preprocessing. Noise was filtered out, and the data was segmented into fixed-width sliding windows of 2.56 seconds with a 50% overlap, resulting in 128 readings per window. The sensor's acceleration signal, which includes both gravitational and body motion components, was separated using a Butterworth low-pass filter. This filter assumed that gravitational forces consist of low-frequency components and used a cutoff frequency of 0.3 Hz. For each window, feature vectors were created by extracting variables from both the time and frequency domains.

**Features:** Accelerometer and Gyroscope Data:Sensor readings captured in three axes (x, y, z).

**Timesteps**:Each activity is recorded over multiple timesteps.

**Frequency**:The data is sampled at a constant rate, ensuring consistency.

**Labels:**

The dataset consists of six activity classes:

1. Walking

2. Walking Upstairs

3. Walking Downstairs

4. Sitting

5. Standing

6. Laying

**Files:**

train/X\_train.txt`: Training features

train/y\_train.txt`: Training labels

test/X\_test.txt`: Test features

test/y\_test.txt`: Test labels

**Size:**

Training set: 7352 samples

- Test set: 2947 samples

**IV. Methodology**

**Exploratory Data Analysis**

**Class Distribution**

The dataset was analyzed to determine the distribution of activity classes. This provides insights into potential class imbalances, which could affect model performance. A count plot of activities revealed that all classes are equally represented, ensuring a balanced dataset.

A screenshot of a computer

Description automatically generated  
**Fig:** Training data distribution

class distribution of training data. X axis labelled as activity while y axis is count of each activity. Here the bar plot shows sitting count as 1250 approx, running 1200, standing 800, jumping 1000 approx

**Outlier Detection**

A screenshot of a computer code

Description automatically generated  
Fig: Outlier Detection

Number of outliers detected in training data is 368 whereas Number of outliers detected in test data is 148.

**Checking for Missing values**

A screenshot of a computer code

Description automatically generated  
Fig: Checking for missing values

**Feature Correlation**

A correlation matrix was computed to identify relationships between features. Strong correlations might indicate redundancy, while weak correlations could signify the independence of features. This step helps in understanding the data structure and identifying any preprocessing needs.

A screen shot of a graph

Description automatically generated

**Fig:** Correlation Heatmap

**Temporal Analysis**

The sensor data was plotted over time to observe patterns and distinguishable characteristics for each activity. This highlighted how activities like "Walking" and "Running" showed periodic signals, whereas "Standing" and "Sitting" had stable readings.

**Data Preprocessing**

A screenshot of a computer code

Description automatically generated

**1. Standardization:** Sensor readings were standardized to ensure consistent scaling. This is essential for time-series data, as deep learning models are sensitive to varying data magnitudes.

**2. Label Encoding**: Activity labels were mapped to numeric values and converted into one-hot encoded vectors for multi-class classification.

**3. Reshaping for LSTM/RNN**: The input data was reshaped to fit the specific requirements of sequential models. LSTMs and RNNs require inputs in the format of **(samples, timesteps, features).**

**4. Train-Test Split:** The dataset was already split into training and testing sets, facilitating straightforward model training and evaluation.

**V. Model Development**

**LSTM Model**

The Long Short-Term Memory (LSTM) model was chosen for its ability to capture long-term dependencies in time-series data. The architecture consisted of stacked LSTM layers with dropout for regularization. The output layer used a softmax activation function to produce probability distributions over activity classes.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

The first LSTM layer has 128 units. The parameter count for this layer is 353,280, which includes the weights, biases, and recurrent connections.

The second LSTM layer has 64 units. This layer does not return sequences, and the parameter count is 49,408.

The total number of parameters in the model is 403,078

The high number of parameters (over 400,000) indicates a complex model that can potentially capture intricate patterns in the data.

The log presents the training process over 20 epochs displaying accuracy, loss, validation accuracy, validation loss metrics.

The model starts with good accuracy of 94.02% suggesting that the model performs well.

**RNN Model**

A SimpleRNN model was developed as an alternative to the LSTM model. While RNNs are simpler, they are often less effective for long sequences due to the vanishing gradient problem. However, the comparison helped in understanding the trade-offs between model complexity and performance.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

The log represents the training process over 30 epochs.

The model starts with a relatively high training accuracy of 58% in the first epoch, indicating a good initial fit.

Validation accuracy improves over time, reaching 84% by the 30th epoch, suggesting the model performs consistently well on the validation set.

**VI. Evaluation Metrics**

**Confusion Matrix**

A confusion matrix was used to evaluate the classification performance of the models. It provided a detailed breakdown of true positives, true negatives, false positives, and false negatives for each class.

Accuracy : Accuracy was computed as the proportion of correctly classified samples to the total number of samples. This metric provided an overall sense of model performance but did not account for class imbalances.

F1 Score : The F1 score was calculated to balance precision and recall, offering a more nuanced measure of model performance, particularly for imbalanced classes.

**Evaluation for LSTM Model**

A screenshot of a computer

Description automatically generated

Model Accuracy: The overall accuracy of the LSTM model is 94.13%, which is an excellent performance metric indicating that the model correctly classifies a high percentage of the test data.

High Accuracy Classes: The highest number of correct predictions is for class 6, with 529 correct predictions. This indicates that the model is particularly good at identifying this class.

Moderate Accuracy Classes: Other classes such as class 1 and class 2 also show high correct prediction counts, suggesting the model performs well on these activities.

Lower Accuracy Classes: The lowest number of correct predictions is for class 3, with 370 correct predictions. This might indicate that the model struggles to distinguish this activity from others.

Misclassification is shown by non-diagonal cells. For example, 75 instances of class 4 were misclassified as class 5. This suggests a need for further model tuning or feature engineering to reduce these errors

**Evaluation for RNN Model**

A screenshot of a computer

Description automatically generated

The RNN model achieved an accuracy of 94.47%, indicating that it correctly classified 94.47% of the instances in the test set. This is an excellent performance metric for a multi-class classification task.

**High Accuracy:** The model shows high accuracy across most classes, with particularly high performance in classifying class 6 and class 5, where it achieves perfect or near-perfect classification.

Misclassifications can be observed, such as class 2 being sometimes confused with class 1 and class 3. These errors could be due to similarities in the features of these classes, which might require further feature engineering or data augmentation to improve differentiation.

**Comparison**

**Accuracy Plot (LSTM Vs RNN)**

A screenshot of a computer screen

Description automatically generated

LSTM'S Training Accuracy: Reached 0.95 within the first few epochs and stabilized.

Validation Accuracy: Quickly rose to around 0.85, fluctuating slightly afterward.

similarly,

RNN’s Training Accuracy: Also reached 0.95 within the first few epochs and stabilized.

Validation Accuracy: Similar to the LSTM, quickly rose to around 0.85 and showed some fluctuations.

Both models show high accuracy and low loss metrics, indicating strong performance in classifying human activities.

Overfitting: Both models exhibit signs of overfitting, with training accuracies higher than validation accuracies.

**Training and Validation Loss Plot (LSTM Vs RNN)**

A screenshot of a computer screen

Description automatically generated

The blue line represents the training loss.

The orange line represents the validation loss.

for rnn ploit The orange line represents the training loss.

The red line represents the validation loss.

Both losses decrease over time, indicating that the LSTM model is learning and improving its performance on both the training and validation sets.

Both LSTM and RNN models show similar trends in loss reduction, but the LSTM model have a slight edge in stability and performance due to its ability to handle longer dependencies in sequential data.

Detecting Abnormal activity

A screenshot of a computer program

Description automatically generated

The model is highly confident (100%).

Real Time Implementation

**VII. Developing SOS Alert Application**

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated  
Fig: Interface of Application

Upon detecting abnormal activity, the pops up. The person can click and send an SOS alert Email if needed to the preferred person.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**VIII. Conclusion**

* LSTM Performance: The LSTM model demonstrated superior performance in terms of accuracy and robustness. Both LSTM and RNN models show similar trends in loss reduction, but the LSTM model have a slight edge in stability and performance due to its ability to handle longer dependencies in sequential data.
* The LSTM model is often preferred for tasks requiring the modeling of long-term dependencies, such as time-series prediction or sequence classification, due to its gated architecture that mitigates the vanishing gradient problem common in standard RNNs.
* RNN Performance: The RNN model achieved reasonable accuracy but struggled with certain classes due to its limited capability to retain long-term information.
* Class-wise Performance: Some activity classes, such as "Walking Downstairs," showed higher misclassification rates, possibly due to overlapping feature characteristics.
* SOS Alert System: The SOS alert system was successfully implemented. It used model predictions to detect abnormal activities and send an email notification to emergency contacts when necessary.
* Application Features: The developed application demonstrated user-friendliness and functionality, offering real-time monitoring of activities, abnormal behavior detection, and automated alerting. This makes it a potentially valuable tool in healthcare, eldercare, and personal safety scenarios.

**IX. Key Takeaways**

* The LSTM model demonstrated superior effectiveness over the RNN model for time-series data.
* Standardization and one-hot encoding were critical in achieving high accuracy.
* Exploratory Data Analysis (EDA) revealed significant patterns and relationships within the dataset.
* The SOS alert system illustrated the real-world applicability of the project.

**X. Future Directions**

Exploring advanced models such as Bidirectional LSTMs, Transformers, or hybrid architectures to improve accuracy.

Real-time Deployment: Extending the system to process real-time data streams from wearable devices.

Mobile Integration: Developing a mobile application for wider accessibility and real-world usage.

Class Imbalance Handling: Implementing techniques like oversampling, undersampling, or class weighting to improve classification performance for minority classes.

**References**

1. Anguita, D., Ghio, A., Oneto, L., Parra, X., & Reyes-Ortiz, J. L. (2013). A Public Domain Dataset for Human Activity Recognition Using Smartphones. In Proceedings of the 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2013).  
   URL: [UCI HAR Dataset](https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones)
2. Zeng, M., Nguyen, L. T., Yu, B., Mengshoel, O. J., Zhu, J., Wu, P., & Zhang, J. (2014). Convolutional Neural Networks for Human Activity Recognition using Mobile Sensors. *Proceedings of the 6th International Conference on Mobile Computing, Applications and Services (MobiCASE)*.  
   URL: [CNN for HAR](https://link.springer.com/article/10.1007/s00521-019-04149-1)
3. Khan, A., Nicholson, J., & Mellor, R. (2020). LSTM Networks for Human Activity Recognition Using Accelerometer and Gyroscope. *Journal of Machine Learning Research, 21(56), 1-24*.  
   URL: LSTM for HAR
4. Anguita, D., Ghio, A., Oneto, L., Parra, X., & Reyes-Ortiz, J. L. (2013). A Public Domain Dataset for Human Activity Recognition Using Smartphones. 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning.
5. Graves, A., Mohamed, A., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).
6. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.
7. Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2), 157-166.
8. Ordóñez, F. J., & Roggen, D. (2016). Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. Sensors, 16(1), 115.
9. Liu, J., Gao, Q., & Wu, Y. (2018). Human Activity Recognition Using Smartphones Based on LSTM Networks. IEEE Access, 6, 76896-76906.
10. Yao, S., Zhao, Y., Zhang, A., & Abdelzaher, T. (2017). DeepSense: A unified deep learning framework for time-series mobile sensing data processing. Proceedings of the 26th International Conference on World Wide Web.
11. Ravi, D., Wong, C., Lo, B., & Yang, G. Z. (2016). Deep learning for human activity recognition: A resource-efficient implementation on low-power devices. 2016 IEEE 13th International Conference on Wearable and Implantable Body Sensor Networks (BSN).
12. Hammerla, N. Y., Halloran, S., & Plotz, T. (2016). Deep, convolutional, and recurrent models for human activity recognition using wearables. Proceedings of the 25th ACM International Conference on Information and Knowledge Management.
13. Zeng, M., et al. (2014). Convolutional neural networks for human activity recognition using mobile sensors. 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services.
14. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems.
15. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
16. Karpathy, A., Johnson, J., & Fei-Fei, L. (2015). Visualizing and understanding recurrent networks. arXiv preprint arXiv:1506.02078.
17. Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
18. Srivastava, N., et al. (2014). Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1), 1929-1958.
19. Xie, Z., Zhang, Y., Zhang, Y., & Wang, Y. (2019). Human activity recognition using a hybrid of handcrafted features and deep features. IEEE Access, 7, 29924-29938.
20. Tufekci, Z., & Tufekci, D. (2018). Sensor-based human activity recognition using LSTM and convolutional neural networks. International Journal of Innovative Computing, Information and Control, 14(4), 1401-1413.
21. Yin, J., et al. (2015). A deep learning approach for human activity recognition based on single accelerometer. 2015 IEEE International Conference on Machine Learning and Applications (ICMLA).
22. Sánchez-Miralles, Á., et al. (2020). A review of human activity recognition for the Internet of Things: Toward smart environments. Sensors, 20(16), 4565.
23. Reiss, A., & Stricker, D. (2012). Introducing a new benchmarked dataset for activity monitoring. Proceedings of the 16th International Symposium on Wearable Computers.