Multivariate Time Series Classification for Stress Detection: A case study in near real-time stress detection

Harikrishnan Changarnkothapeecherikkal
Data Science Graduate Student
Michigan Technological University
Houghton, Michigan
hchangar@mtu.edu

Ankit Chhillar
Health Informatics Graduate Student
Michigan Technological University
Houghton, Michigan
achhilla@mtu.edu

Manpreet Singh Sandhu
Data Science Graduate Student
Michigan Technological University
Houghton, Michigan
msandhu@mtu.edu

***Abstract*—Stress is a commonly observed problem within the healthcare industry. Identifying stress levels in real time can help prevent negative outcomes like burnout among nurses. Researchers are exploring the potential of using machine learning for stress detection among nurses in hospitals. They used physiological data like heart rate and respiration to predict stress levels. However, finding this sort of solution is a challenging and complicated issue. This is a big data problem, thus sophisticated approaches are needed to make sure the data is correctly gathered, transported, and analyzed. Multiple sensors must provide data for stress detection, and the data must be acquired in a brief period of time. Some of the learning models that are used to forecast stress need a lot of computation. In this project, we tried evaluating multiple machine learning and deep learning models for predicting stress. We also tried to identify suitable preprocessing strategies based on models. The best model trained was able to identify stress using Heart rate and respiratory rate with an accuracy of 59 percent and an F1 score of 0.57. Despite the unexceptional findings, we are certain that some of the processes and models will be useful if we get access to more sensor data, such as the EDA and skin temperature sensor, which are essential for stress prediction.**

***Keywords*— deep learning, time series classification, convolutional neural network, LSTM, transformer, machine learning**

I. Introduction

In today's world, people are advancing in terms of technology but with the advancement of technology, the stress-related issues are increasing day by day. Stress is a commonly observed problem within the healthcare industry, irrespective of clinicians, physicians, nurses, pharmacists, or any other healthcare professional everyone is exposed to stress. Particularly the nurses are exposed to more stress due to their overtime or changing shifts thus experiencing high levels of stress in their daily work. Identifying this stress level and addressing the changes in real-time will help in preventing negative outcomes like those of a decrease in quality care, decreasing job satisfaction among nurses, and burnout because of irregular working hours. Traditionally stress is detected among nurses through self-report, observation, etc. have certain limitations and are not much reliable and accurate stress level representations. The Multivariate time series classification for stress detection through real-time stress detection among nurses is a good opportunity and also provides a promising approach by analyzing the multiple physiological data of nurses.

The use of various physiological data like heart rate, the respiration rate of nurses, and skin conductance have the capability to provide more accurate and real-time stress assessment reports in comparison to the traditional stress calculation methods. The project focuses more on the nurses working on healthcare premises. In this project, we have taken the stress dataset of nurses from Dryad which is available as open source.

This project aims to explore the potential of using machine learning algorithms and big data tools for stress detection among nurses in hospitals. The case study approach takes a detailed observation and provides conclusions based on Multivariate Time series and classifying the stress levels in nurses in a healthcare setting. It has potential implications for applying it further within the complete healthcare industry. Using this case study approach, we are trying to implement and find the feasibility and effectiveness of using the approach of real-time stress detection. This will also result in potential implications of stress management, improving healthcare outcomes, and reducing burnout among healthcare professionals.

II. Related Work

III. Dataset Description

The dataset includes physiological data gathered from 35 students and staff members of the University of Galway in Ireland and the University Hospital Galway in Ireland. [1] 35 healthy individuals who completed three stressful activities provided the stress-predict dataset, which included a BVP signal, inter-beat intervals, heart rate, respiration rate, and accelerometer data. These stressful activities include an interview, a Stroop color-word test, and a period of hyperventilation with a baseline/normal time. [2]The respiratory rate was determined using a PPG-based respiratory rate estimation method while the blood volume pulse (BVP), inter-beat-intervals, and heart rate were constantly monitored using Empatica watches.

The full dataset, however, wasn't available. The original dataset's authors released a cleaned version of the data for public access, which may be used for descriptive, statistical, and classification analysis [3]. The produced dataset differs from all other publicly available datasets in that it contains details on the estimated RR for each participant. The dataset includes data such as the timestamp, participant identifier, heart rate, respiratory rate, and a label indicating stress or non-stress.

IV. Experimental Setup

A. Train and Validation Split

The gold standard is often seen as a three-fold split, separating the dataset into Training, Validation, and Testing datasets. With 70 percent of the data going into the train split and 15 percent going into the validation split and test split, we divided the data into three groups. 75811 records in the Train subset and 16246 records in the test and validation splits may be found in the final subsets. To ensure consistent class distribution, the split was stratified depending on the status variable and carried out at random. There are uneven distributions of classes.

B. Model Performance Evaluation

The generalizability of the model was assessed using the test dataset, which also provided a precise assessment of its performance on unseen data. After using the training dataset, the model was improved depending on the performance of the validation data. The receiver operating characteristic (ROC) curve was visually assessed when we compared models based on Accuracy, Precision, Recall, and F1 score. Since the F1 score includes a model's accuracy and recall ratings, it was used to determine which model was the best.

V. Methods

The project's objective is to predict stress using classification models based on heart rate and breathing rate using machine learning and deep learning. Stress cannot be predicated on sensor data in a single instant, but is often detected over a period of time. The heart rate and respiratory rate of a participant at rest and during stress are shown in Fig. 1. The red line denotes stress, whereas the green line represents the baseline. The graphic makes it obvious that the pattern changes when the subject is under stress. Given the sensor data at a certain time and a few lag factors, the models should be able to recognize this pattern and identify stress.

We tested a variety of deep learning architectures that could automatically extract features from time-series sequence data without the need for feature engineering. Models for machine learning were also tested. Tabular classifiers, however, could not be effective in classifying time series. We utilized simple statistics, such as min, max, standard deviation, etc. with 60 lag features to describe the time series as feature vectors for machine learning models.

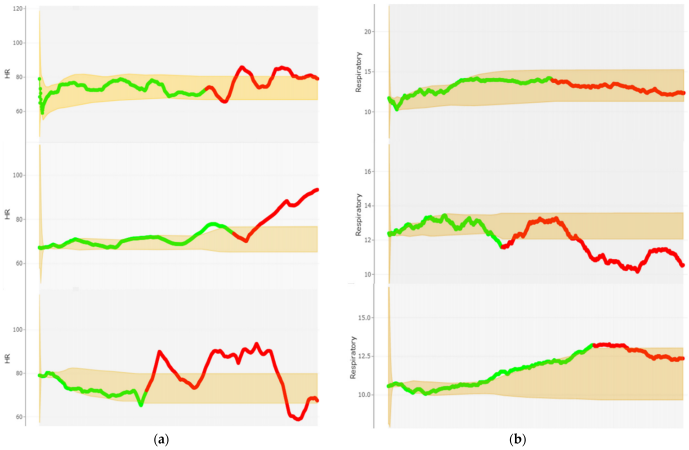

Fig. 1. (a) Heart rate reading: baseline (green) vs. stress (red) task (b) Respiratory rate: During each baseline (green) vs. stress (red) task. Source: [2]

A. Preprocessing Procedure

The pre-processing procedure consists of two stages. The signal pre-processing phase comes first. The extraction of signals from the non-signal segment is a part of this stage. [2] This stage also includes noise removal. Advanced signal processing methods like peak augmentation may be necessary for certain of sensor signals. For sensors like the photoplethysmogram, additional information like the Signal Quality Index could be obtained. These signal processing and enhancement methods demand extensive subject-matter knowledge. These steps were omitted because they were outside the purview of this project.

Additional signal analysis procedures include detecting signal peaks, calculating RR intervals, correcting errors in discovered peaks, and estimating respiratory rates. The dataset we utilized already includes some of these analyses and estimates of respiratory rates.

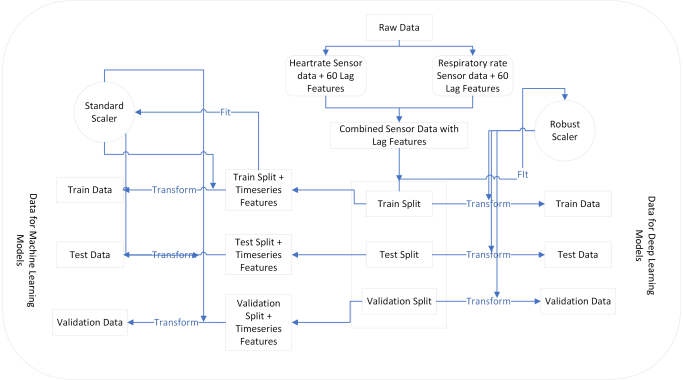

Fig. 2. Pre-processing Flowchart

A flowchart of the data preparation steps used for this project is shown in Figure 2. While deep learning models automatically extracted features from time series sequences, the machine learning models we employed did not. To preserve

consistency, we initially used the TsExtract Library to extract lag characteristics. For each heartbeat and respiratory rate data, 60 lag characteristics were retrieved using a window size of 60. The final dataset had 120 features. The generated dataset was divided into test, train, and validate subsets.

Additional feature engineering was required for machine learning models. Using 60 lag features and basic statistics like min, max, standard deviation, etc., your time series will be represented as feature vectors for machine learning models. The data were scaled using a standard scaler(sklearn). The mean and standard deviation were calculated by fitting the scaler to the Train data, and the resulting values were then applied to the test and validation datasets. Deep learning models, on the other hand, might take features out of a time series sequence. When scaling features for data to be utilized in the deep learning model, a robust scaler(sklearn) was employed. For usage in multidimensional time series classification, the lag characteristics are also organized as sequences and reshaped as multidimensional data.

B. Deep Learning Models

We looked at a number of Multivariate time series models instead of our intended aim of testing with multimodal Deep learning models due to the dataset's limited accessibility. Conventional methods often use dynamic temporal warping (DTW) or shapelet transformation to extract discriminative features from the original time series, and as a result, an off-the-shelf classifier is used. The accuracy performance of these approaches is constrained by the separation of the feature extraction and classification stages. Additionally, the majority of the methods in use today do not consider the fact that time series typically contain characteristics at many time scales. [4]

However, Deep Learning Models are capable of extracting features and automatically producing useful time series representations. Heart rate and respiratory rate were utilized to do multivariate time series classification using a Convolutional Neural Network, 3 Variations of LSTM (Long Short-Term Memory Networks), and Transformer models to predict stress. Each of these architectures provides a unique type of representation and is beneficial for various types of data.

1) Convolutional Neural Network:

The standard choice for image classification and various other computer vision applications is convolutional neural networks. Despite the convolution neural network's starting point in images, research demonstrates that it is as effective with other sequential data types, such as natural language processing. A Multi-Scale Convolutional Neural Network for Time Series Classification developed by Cui, Z., Chen, et al. in 2016 beat several other models for 44 UCR time-series datasets.

The convolution neural network that we used has seven 1D convolution layers connected in sequence, a BatchNormalization layer, and a LeakyRelu nonlinearity layer after each convolution. We are able to employ a greater learning rate because of batch normalization, which improves the way input flows across intermediate layers of the neural network. In our scenario, LeakyRelu's ability to allow a tiny gradient while the unit is not in use helped to somewhat enhance performance. The classification head was a fully connected layer, and a dropout layer was employed after the pooling layer to avoid overfitting. We employed dropout after pooling since our model was severely overfitted, even though dropout is often not required when we have batch normalization.

For our model, we employed the Adam Optimizer [5]. By storing both the individual learning rate of RMSProp and the weighted average of momentum, Adam employs both Momentum and Root Mean Squared Propagation heuristics. Adam is quicker than Stochastic Gradient Descent and converges sooner.

2) Long Short-Term Memory Networks: A deep recurrent neural network architecture called long short-term memory (LSTM) is used to classify time-series data. Recurrent neural network types that can learn from and remember over long input data sequences include LSTM network models. LSTM models have been shown to be successful in classifying time series, particularly for the reliable classification of physiological data. Time-frequency time-space LSTM, developed by T. D. Pham in 2021 for the accurate categorization of physiological data [6], has proven successful in classification tasks using physiological signals.

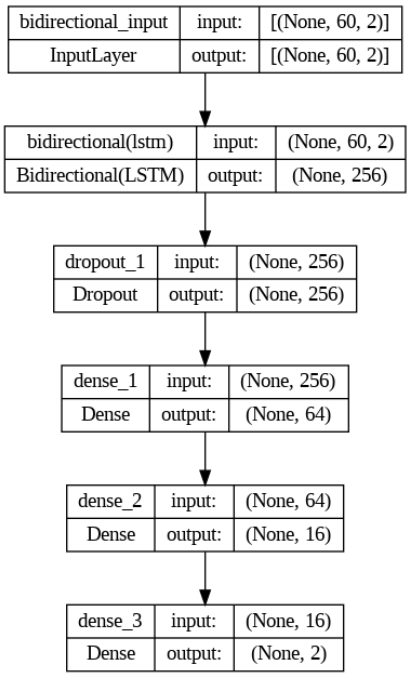

Fig. 3. LSTM Model Architecture

We experimented with three distinct LSTM-based models. In the first model, the classification head was made up of three densely connected layers and just one bidirectional LSTM layer. Bidirectional LSTM may use information from both sides and, unlike conventional LSTM, accepts input that flows both ways. 128 units make up the LSTM layer. The dimension of the hidden state is in fact expressed by the number of units.

The dropout layer was utilized to suppress overfitting since the model was becoming overfitted on the training dataset. Adam served as the optimizer. The model architecture for model is in Figure 3.

The second LSTM model was similar to the first model, but we used two Bidirectional LSTM layers instead of one. The dropout layer was used after each LSTM layer to avoid overfitting. In this model, we have four densely connected layers for classification compared to the 3 layers used in the earlier model. Adam was used as an optimizer for this model as well. The third LSTM model is the same as the second model except for the loss function. The second model uses a sigmoid loss function. But for the 3rd LSTM model, we tried using focal loss instead of the sigmoid loss function.

The first mention of focal loss was made in the retinanet paper [7]. When you have very unbalanced classes, the focal loss is very helpful for classification. It gives hard instances more weight than properly labeled ones. In comparison to the loss value associated with a correctly categorized example, the loss value for a sample that the classifier misclassified is substantially larger. A type of loss function called focal loss helps the model's ability to discriminate. In this case, the majority class is not Stress category and the data is heavily imbalanced.

3) Transformer: Another well-liked Deep Learning model that might be applied to applications involving sequence categorization is Transformer. When processing big data sequences and natural language, this attention-based strategy has been shown to be successful. Because it is not recurrent, it may employ earlier time step characteristics without losing information, making it superior to RNNs. With a little change to the design, transformers could also employ time series data that were sequential for classification jobs when utilizing a classification head. [8]

Instead of using natural language, we employed the time series-based Transformer architecture from "Attention Is All You Need." [8] To utilize as input for the classification, a transformer encoder was employed, and the representations from the transformer were concatenated using a pooling layer, and a multi-layer perceptron was used as the classification head. Dropout layers were used to reduce overfitting.

C. Training and optimization strategies for Deep Learning Models

Given that the class distribution was severely unbalanced, we employed the class weight parameter for all deep learning models. The fraction of the class distribution that the class weights used to be was inverse. Due to the ease of overfitting the majority class, this aids in reducing the weight of its losses. One of the most useful techniques to prevent overfitting on the majority class for highly unbalanced datasets is what is commonly referred to as cost-sensitive learning.

For all of our deep learning models, we employed the learning rate decay strategy, which is a method for training modern neural networks. The network is first trained with a high learning rate, which is gradually reduced or decayed until local minima are attained. Empirically, it is seen to support both generalization and optimization. Additionally, we employed early termination if the validation loss did not improve after a particular number of consecutive epochs. The cap for all epochs was set at 100.

D. Machine Learning Models

Since the Deep Learning models did not perform as expected, we also attempted employing a few Machine Learning methods. Most machine learning models need extra stages, unlike deep learning models. Applying a typical learning algorithm directly and treating each time point as a new feature is a popular but problematic approach to time series categorization. In this method, the algorithm disregards the data's temporal order-related information. The forecasts would remain the same even if the feature order was changed. Giving the model more features might be a solution. Sliding-window representations may be used to convert time series datasets into supervised learning. But typically, feature engineering calls for some amount of subject-matter knowledge. For this project's scope, we used Basic statistics including Min, Max, and Standard Deviation, etc. with 60 lag features to represent your time series as feature vectors.

We used the supervised learning representation as input to the RandomForest classifier and Adaptive Boosting Classifier.

VI. Results

Figure 5 shows a comparison of the model performances based on accuracy. The results show that most of the models are overfitted and not generalizing. Most of the models have a high bias and high variance. The best-performing model AdaBoost has an Accuracy of 59 percent and an F1 score of 0.57.

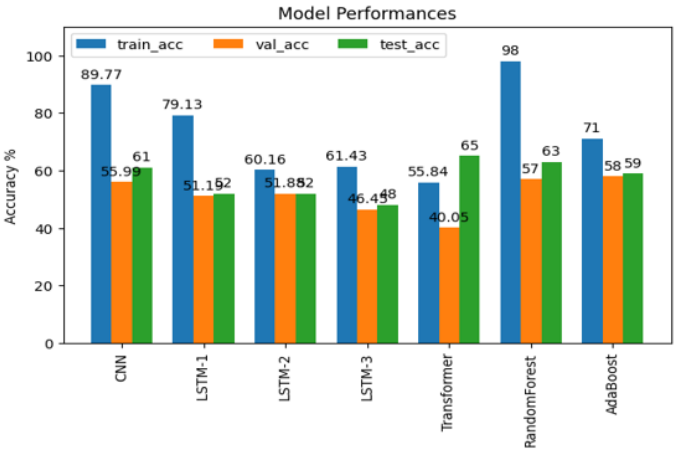

Fig. 4. Model Performance Caparison

We also examined the feature importance of the AdaBoost model, using the mean decrease in impurity(MDI) measure, and the statistical features are seen to be more important for the prediction than the lag features.

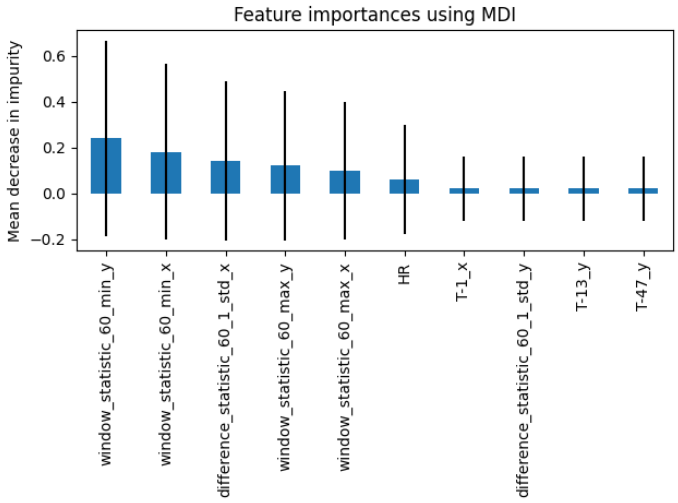

Fig. 5. Feature Importance - AdaBoost model

Unlike other models, Transformer Model was under-fitted and the performance of the model could potentially improve if the model is trained for more epochs.

The training history plots were also evaluated for the deep learning models which revealed some information that will be helpful in tuning the model.

VII. Conclusion

In this project, we tried evaluating multiple machine learning and deep learning models for predicting stress. Most models did not generalize very well and did not perform as expected. Iqbal T et al. [1] utilized the same dataset and predicted stress with an accuracy of roughly 94 percent; however, they did it using information from skin-based temperature sensors, electrodermal activity sensors, and average heart rate data that was retrieved using photoplethysmography. Since the heart rate and breathing rate we use to predict stress might not be adequate, further sensor data could be required. Some of the signal pre-processing phases may have been skipped because they weren't necessary for this project, which could have affected the models' performance.

In addition to the difficulties mentioned above, near-real-time automatic stress detection is a significant issue in the real world. By alerting people to situations connected to stress, an organization's workplace environment or a person's physical and mental health may be improved. The growing need for wearable technology makes this type of solution conceivable today. However, finding this sort of solution is a challenging and complicated issue. This is a big data problem, thus sophisticated approaches are needed to make sure the data is correctly gathered, transported, and analyzed. Multiple sensors must provide data for stress detection, and the data must be acquired in a brief period of time. Some of the learning models that are used to forecast stress need a lot of computation. Future efforts will go in the direction of combining all the procedures to produce a dependable and effective solution.

References

[1] T. Iqbal, A. J. Simpkin, D. Roshan, N. Glynn, J. Killilea, J. Walsh, G. Molloy, S. Ganly, H. Ryman, E. Coen, A. Elahi, W. Wijns, and A. Shahzad, "Stress monitoring using wearable sensors: A pilot study and stress-predict dataset," *Sensors*, vol. 22, no. 21, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/21/8135

[2] T. Iqbal, A. Elahi, W. Wijns, B. Amin, and A. Shahzad, "Improved stress classification using automatic feature selection from heart rate and respiratory rate time signals," *Applied Sciences*, vol. 13, no. 5, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/5/2950

[3] Italha-D, "Italha-d/stress-predict-dataset: This dataset is associated with "stress monitoring using wearable sensors: A pilot study and stress-predict dataset" paper." [Online]. Available: https://github.com/italha-d/Stress-Predict-Dataset

[4] Z. Cui, W. Chen, and Y. Chen, "Multi-Scale Convolutional Neural Networks for Time Series Classification," *arXiv e-prints*, p. arXiv:1603.06995, Mar. 2016.

[5] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[6] T. D. Pham, "Time–frequency time–space lstm for robust classification of physiological signals," *Scientific Reports*, vol. 11, no. 1, p. 6936, Mar 2021. [Online]. Available: https://doi.org/10.1038/s41598-021-86432-7

[7] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," *arXiv e-prints*, p. arXiv:1708.02002, Aug. 2017.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," *arXiv e-prints*, p. arXiv:1706.03762, Jun. 2017.

Appendix

A. Contribution of Each Member

1) Harikrishnan Changarnkothapeecherikkal:

- Report: Abstract, Data set Description, Experimental Setup, Methods, Results
- Model Building: Model selection, Deep Learning Models Training and Evaluation, Preprocessing, Data Preparation, Feature Engineering
- Presentation

2) Ankit Chhillar:

- Report: Introduction, Conclusion
- Model Building: Model selection, Machine Learning Training
- Documentation

3) Manpreet Singh Sandhu:

- Report: Related Work
- Model Building: Model selection, Machine Learning Training
- Presentation