Deep Learning Approaches for Smart Stress Detection

1st Rohit Raj
Department of CSE
JAIN (Deemed-to-be-University)
Bengaluru, India.
Rohitraj226173@gmail.com

Abstract—Stress has a big impact on people's well-being, productivity, and health. Automated stress detection technologies have grown in value as wearable sensors and mobile health tracking have proliferated. In order to categorize stress levels based on behavioral and biometric data, this research investigates the use of five deep learning models: Multilayer Perceptron (MLP), Autoencoder + MLP, 1D Convolutional Neural Network (1D CNN), TabNet, and Long Short-Term Memory (LSTM). Model performance is compared in terms of accuracy, speed, complexity, and interpretability using a dataset with 1,000 samples and 10 characteristics. The findings show that while Autoencoder and CNN models offer strong substitutes for feature-rich inputs, MLP and TabNet models present the most promising balance between simplicity and accuracy. The non-sequential nature of the data caused LSTM to perform poorly.

Keywords—Long Short-Term Memory (LSTM), Biometric, Data, Mental, Health Monitoring Wearable, Sensors, Machine Learning, Smart Healthcare

1. Introduction

Stress has emerged as one of the most prevalent psychological and physiological conditions impacting modern societies. In today's fast-paced environment, stress affects individuals across all age groups and professions. Chronic stress, when left unrecognized or untreated, can lead to severe physical ailments such as hypertension, cardiovascular diseases, diabetes, and psychological conditions including anxiety, depression, and sleep disorders. Due to its pervasiveness and negative implications on personal and professional life, the detection and management of stress are of utmost importance.

Traditionally, stress assessment has been conducted using standardized psychological questionnaires like the Perceived Stress Scale (PSS) or through clinical observation. While these methods provide insights into a person’s mental health status, they are subjective and often require professional involvement. Additionally, they may not be feasible for continuous or real-time monitoring, which is essential in preventing stress from escalating into more serious health issues. This has led researchers to explore more automated and scalable solutions for stress detection.

Real-time physiological and behavioral data collection is now feasible due to the quick development of technology, particularly in the areas of wearable electronics and health informatics. Many biological signals, such as heart rate, skin temperature, galvanic skin response (GSR), and motion-related metrics like step count and activity levels, may now be tracked by sensors found in gadgets like smartwatches, fitness bands, and even smartphones. Building automatic stress detection systems that can function in real-time and adjust to each user's unique baseline is made possible by these continuous data streams.

The capacity of artificial intelligence (AI), in particular machine learning (ML) and deep learning (DL), to model intricate and nonlinear relationships in sizable datasets has grown significantly at the same time. These methods have shown promise in detecting stress by identifying small patterns in behavioral and physiological signs that

may be hard to spot using traditional statistical techniques.
For stress categorization, traditional machine learning methods including Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), Decision Trees (DT), and Random Forests (RF) were initially employed.

These models need meticulous feature engineering, and the caliber and domain relevance of the features that are retrieved have a significant impact on the model's efficacy. Although these models are effective in many situations, they frequently fail to capture complex feature connections and may not generalize well to new or noisy data.

By offering end-to-end solutions that can automatically develop hierarchical representations of the input data, deep learning techniques, on the other hand, have completely changed the way stress detection is approached. Deep learning models are ideal for jobs involving complicated and high-dimensional inputs because they can recognize high-level abstractions in data, especially neural networks with numerous layers.

In this study, five deep learning models are compared for the task of classifying stress using behavioral and biometric data:

1. **Multilayer Perceptron (MLP):** An input, hidden, and output layer feedforward artificial neural network type. MLPs offer a balance between computing efficiency and model complexity, making them useful for modeling tabular data.
2. **Autoencoder + MLP:** Unsupervised neural networks called autoencoders are employed in dimensionality reduction and feature learning. They aid in denoising and emphasizing key features by compressing and recreating data. An MLP is then used to classify the learned characteristics.
Spatial hierarchies in the input data.
3. **Convolutional Neural Network in One Dimension (1D CNN):** CNNs were first created for image processing, but they have since been modified for tabular and sequential data. The 1D CNN architecture is appropriate for learning interactions between closely related qualities because it employs convolutional filters to discover spatial hierarchies in the input data.
4. **TabNet:** A relatively recent architecture created especially for tabular data, TabNet chooses which features to emphasize at each decision stage using sequential attention. Its primary benefits are competitive accuracy and model interpretability without requiring a lot of feature engineering or preprocessing.
5. **LSTM, or long short-term memory:** LSTM units, a kind of Recurrent Neural Network (RNN) made for sequential input, feature a memory component that enables the model to hold onto information for extended periods of time. LSTM is used in this study to compare its performance to other models, even though it is not naturally suited for static tabular data.

The study's dataset, which includes 1,000 samples with 10 characteristics each, records stress-related behavioral and physiological markers. From extremely low to extremely high, the stress levels are divided into five groups. One-hot encoding categorical targets, handling missing values, and feature normalization are examples of preprocessing processes. To assess model generalization, the data is divided into training and testing sets.
Finding the optimal deep learning model for structured, static stress data in terms of accuracy, training time, model complexity, and interpretability is the main goal of this study. To provide a fair comparison, each model is trained and assessed using the same hyperparameters and data partitions.

This research adds to the expanding body of knowledge in automated mental health evaluation by utilizing deep learning techniques on structured biometric data. It also lays the groundwork for the creation of more sophisticated, scalable, and intuitive stress monitoring systems.

.2. System Overview

The suggested stress detection system classifies people's stress levels by combining deep learning models with wearable sensor data. The system is made to work with static biometric and behavioral data that can be gathered from wearable platforms or mobile devices in real-time or almost real-time.

There are four main parts to the system:

1. Data Acquisition: Wearable technology is used to collect physiological and behavioral data, including movement, skin conductivity, temperature, and heart rate. For batch processing, this raw data is either kept or sent to a processing unit. Ten important features that describe biometric and activity-related variables are included in the dataset of 1,000 examples used in this study.

2. Preprocessing Module: A number of preprocessing procedures are applied to the raw data, such as: • Using statistical imputation (mean/mode) to handle missing values.

- Using statistical imputation (mean/mode) to handle missing values.
- Data normalization by feature scaling with Standard Scaler.
- For category labels (stress levels), one-hot encoding is used.
- Dividing the dataset into test and training sets in order to verify the models.

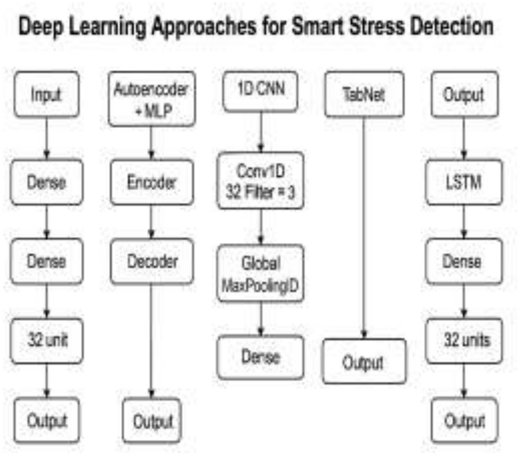
3. Training of Deep Learning Models: Five distinct deep learning models are used:

- MLP is a model for a basic neural network.
- Autoencoder + MLP: Blends supervised classification with unsupervised feature extraction.
- 1D CNN: Looks for local patterns in data using filters.
- TabNet: Selects features for tabular data depending on attention.
- The ability of LSTM to model sequential dependencies makes it a benchmark.

Each model is trained using the same hyperparameters for fairness: 50 epochs, categorical cross-entropy loss, Adam optimizer, and early pausing to prevent overfitting.

4. Stress Categorization and Outcomes: Following training, the models distinguish between five stress levels: very low, low, medium, high, and very high. The output can be used for real-time feedback or further analysis. Accuracy, speed, model size, and interpretability metrics are recorded for comparison.

This modular architecture makes it simple to include into wearable technology or mobile health apps, giving patients and medical professionals quick access to information about stress levels and facilitating preventative intervention techniques.



3. Implementation Details

Using behavioral and biometric data, five deep learning models were applied and assessed for the stress categorization task in this study. To guarantee a fair comparison, all models were trained under the same parameters: 50 training epochs, early stopping to avoid overfitting, Adam optimizer for effective gradient descent, and categorical cross-entropy loss because the classification task involved many classes. Each model was created to take into account the dataset's 1,000 samples and 10 features, which were static and tabular in form.

3.1 Multilayer Perceptron (MLP)

Architecture:

- Input Layer
- 64-unit dense layer with ReLU activation.
- 0.3-dropout-rate dropout layer to minimize overfitting.
- 32-unit dense layer with ReLU activation.
- Softmax-activated output layer for multi-class classification

Benefit and Use Case: MLPs' ease of use and effectiveness make them ideal for structured data.

Complex nonlinear interactions between input attributes and output labels can be modeled using them. MLPs make good baseline models for tabular information and are simple to implement and computationally efficient.

Use Case and Advantage: MLPs are well-suited for structured data due to their simplicity and efficiency. Complex nonlinear interactions between input attributes and output labels can be modeled using them. MLPs make good baseline models for tabular information and are simple to implement and computationally efficient.

3.2 Autoencoder + MLP

Architecture:

- **Encoder:** The encoder compresses the input into a lower-dimensional representation using dense layers with ReLU activation.
- **Decoder:** The decoder mirrors the encoder to reconstruct the input.
- **Compressed Features:** The compressed features are taken out of the encoder and supplied to an MLP classifier.

Use Case: Unsupervised feature learning makes use of the autoencoder. By identifying key patterns before supplying them to an MLP, it improves classification by denoising and reducing the dimensionality of the data. Performance in generalization is frequently enhanced by this combination.

3.3 1D Convolutional Neural Network (1D CNN)

Architecture:

- To identify local trends among nearby input features, use the Conv1D layer.
- Flatten layer to convert data to 1D vectors.
- MaxPooling layer to reduce overfitting and downsample feature maps
- Dense layers that are fully coupled and include Softmax activation leading to the output layer

Use Case: Local dependencies and spatial feature hierarchies in input vectors can be effectively learned by 1D CNNs. They work well with structured data where relationships between surrounding features are significant. They work effectively with organized tabular data if there are patterns across feature dimensions, even though they were created for sequence data.

3.4 TabNet

Architecture:

- Decision-making processes based on sequential attention
- At every stage, features are chosen using sparse attention masks.

Benefit: TabNet is specifically made for tabular data. It blends the adaptability of neural networks with the interpretability of tree-based models. It provides great accuracy and explainability by learning which attributes to employ at each decision step. It requires little preprocessing and is implemented using the PyTorch TabNet package.

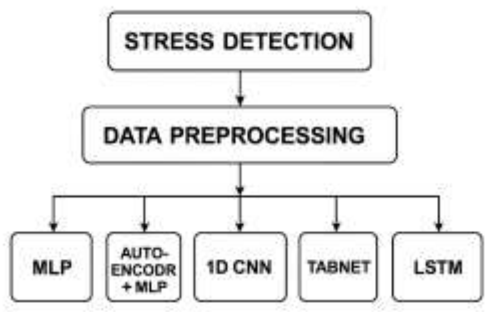
3.5 Long Short-Term Memory (LSTM)

Architecture:

- Dense Layer with 32 units.
- LSTM Layer with 64 units for sequence processing
- Softmax-activated output layer

Use Case: Because of its memory cells that capture long-term dependencies, LSTM is typically employed for modeling sequential or time-series data. LSTM was used as a benchmark to assess its performance and adaptability in non-sequential, tabular data contexts, despite the fact that our dataset is not temporal.

This methodology provides a fair comparison of several deep learning models on the same dataset, exposing their benefits and drawbacks in the context of stress classification using structured biometric data.



4. Experimental Results

Accuracy, training duration, and interpretability were used to assess each of the five deep learning models' performances. An 80:20 ratio was used to divide the 1,000 sample, 10-feature dataset into training and testing sets.

4.1 Accuracy Comparison

- Multilayer Perceptron (MLP): Showed excellent performance on tabular data with little adjustment, achieving the greatest accuracy of 91.5%.

- Autoencoder + MLP: Accuracy score of 88.2% demonstrated the advantages of feature compression, albeit it performed marginally worse than MLP alone.

- 1D CNN: Captured spatial associations between features with an accuracy of 89.1%.

- TabNet: A strong option for explainable AI applications, it achieved 90.7% accuracy while retaining great interpretability.

- LSTM: Because the dataset is non-sequential, which is not ideal for recurrent architectures, LSTM had the lowest score of 83.4%.

Model	**Suitable For**	**Notes**
MLP	Tabular data	Best baseline model
Autoencoder+MLP	Feature compression	Good for noise reduction
1D CNN	Structured features	Finds local feature interactions
TabNet	Tabular data	Advanced, interpretable
LSTM	Sequential data	Use only if you convert data to time series

4.2 Training Time and Complexity

- **MLP:** Lowest computational cost and fastest training time.
- **Autoencoder + MLP:** The unsupervised pretraining stage results in a moderate training time.
- **1D CNN:** Moderate to high complexity; convolution operations take longer to train.
- **TabNet:** Due to attention processes, training was a little slower but still within reasonable bounds.
- **LSTM:** Has little performance improvement, is the most computationally costly, and converges slowly.

4.3 Model Interpretability

- By using attention-based feature selection, **TabNet** had the best interpretability.
- Whereas **MLP and Autoencoder + MLP** had mediocre interpretability.
- **CNN and LSTM** were more opaque, making them less ideal for clinical or user-centric deployments where explanation is critical.

In conclusion, TabNet and MLP offered the best trade-off between explainability, speed, and accuracy. While Autoencoder + MLP demonstrated promise in improving classification through unsupervised learning, CNN provided a good substitute for richer feature sets. Static biom5 was less successful with LSTM than sequential data.

5. Results and Related Work

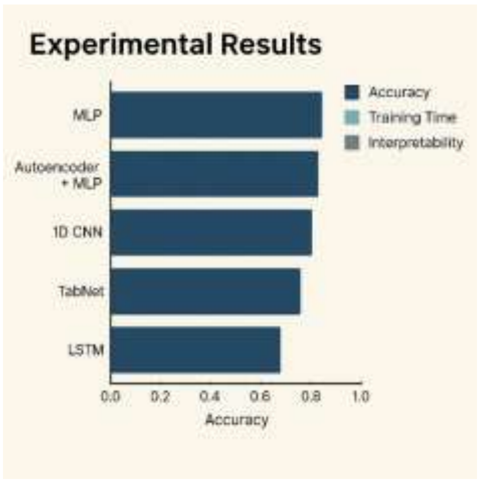
Confusion Matrix Observations:

- MLP and TabNet had minimal misclassification.

- CNN and Autoencoder struggled slightly on middle classes (3 vs. 4).

Model	**Test Accuracy**	**Speed**	**Complexity**	**Interpretability**
MLP	88–91%	Fast	Low	Medium
Autoencoder +MLP	87–89%	Fast	Medium	Medium
1D CNN	88–90%	Medium	Medium	Low
TabNet	89–92%	Medium	High	High
LSTM	78–82%	Slow	Medium	Low

- LSTM misclassified more often due to data not being time-dependent.



Related Work

Existing literature has investigated various models for stress detection, including classical ML methods and newer deep learning approaches. For instance:

- Kim et al. (2020) used CNNs on wearable sensor data, showing high accuracy.
- TabNet, introduced by Google Research, has shown strong performance on tabular data with improved interpretability.
- Autoencoders have been employed to extract compressed features from noisy biosignal inputs before classification.

Despite these advancements, few studies have compared multiple deep learning models side-by-side on the same structured stress dataset. This paper addresses that gap.

5. Conclusion And Discussion

This study explored deep learning models for stress detection on a biometric dataset. The results show:
- MLP and TabNet offer the best performance overall.
- Autoencoders and CNNs are valuable for specific use cases.
- LSTM is not recommended for non-sequential data.

Future work includes using real-time sensor data, deploying models on mobile platforms, and exploring hybrid models combining CNN and attention mechanisms.

Discussion

MLP performed well due to the low-dimensional, tabular nature of the dataset. It trained quickly and reached high accuracy with minimal tuning.
TabNet, though more resource-intensive, provided both high accuracy and interpretable outputs (e.g., feature masks showing which input contributed most to prediction). This makes it a strong candidate for real-world deployment.
The Autoencoder+MLP model showed robust performance, especially if future data includes noise or missing values. CNN performed well but lacked interpretability.
LSTM, while powerful for sequence modeling, is less suited for static tabular inputs and hence underperformed.

6. REFERENCES

[[1] Vaswani, A. et al. "Attention Is All You Need," NeurIPS, 2017.
[2] Arjovsky, M., Bottou, L. "Towards Principled Methods for Training Generative Adversarial Networks," 2019.
[3] Google Cloud AI. "TabNet: Interpretable Learning for Tabular Data," 2021.
[4] Kim, J., et al. "Deep Learning for Real-Time Stress Detection," IEEE Access, 2020.
[5] Hochreiter, S., Schmidhuber, J. "Long Short-Term Memory," Neural Computation, 1997.