Speech-Based DL Model

# **Speech-Based DL Model: Sleep Disorder Classification**

## **1. Introduction**

The goal of this project was to build an accurate machine learning model to **predict sleep disorders** using **audio-based sequential features** extracted from participants.  
 Each participant's data consists of high-dimensional feature sequences extracted via the **openSMILE toolkit**.

## **2. Dataset Details**

* **Participants**: 275
* **Features per timestep**: 201
* **Timesteps per participant**: Up to ~8000
* **Labels**: Sleep disorder classes (binary or multi-class)

Each .csv file corresponds to a single participant containing sequential features over time.

## **3. Problem Understanding**

* The dataset is **sequential**, meaning **time relationships** between frames are crucial.
* Classical machine learning models (SVM, XGBoost) aren't ideal because they **don't capture temporal patterns**.
* We need models that can **model sequences** like **RNNs**, **LSTMs**, **CNNs**, or **TCNs**.

## **4. Approach Overview**

| **Step** | **What we did** |
| --- | --- |
| Data Preprocessing | Standardized features per participant, padded sequences to fixed length |
| Model Choices | Focused on sequence models (BiLSTM, CNN-BiLSTM, etc.) |
| Regularization | Added Dropout, Layer Normalization |
| Training Tricks | Early stopping, gradient clipping |
| Handling Imbalance | Used weighted CrossEntropy loss |
| Evaluation | Confusion matrix, classification report, loss/accuracy curves |

## **5. Model Evolution Journey**

### **5.1 Basic BiLSTM**

* Started with **2-layer BiLSTM** (128 and 64 hidden units).
* Overfitting was observed (train accuracy very high, test accuracy low).
* **Action**: Reduced to **single-layer BiLSTM** with 128 hidden units (bidirectional).
* Added **Dropout** and **LayerNorm** for regularization.
* ✅ Result: Much better generalization.

### **5.2 CNN + BiLSTM**

* Added a **1D CNN layer** before BiLSTM to compress features.
* CNN learns **local short-term patterns**, LSTM learns **long-term patterns**.
* ✅ Result: Slight improvement over plain BiLSTM.

### **5.3 Weighted CNN + BiLSTM**

* Used **Weighted CrossEntropyLoss** based on training set imbalance.
* Also used **WeightedRandomSampler** to balance batches.
* ✅ Result: Helped slightly better handling of minority class.

### **5.4 CNN + BiLSTM + Attention**

* Introduced **Self-Attention Layer** after BiLSTM.
* Model learns to **focus on important frames**.
* ❗ Result: Surprisingly, attention model did not significantly improve performance.
* Possible reasons:
  + Attention overhead was too much for relatively small data.
  + LSTM alone was already capturing enough useful patterns.

### **5.5 Temporal Convolutional Network (TCN)**

* Tried pure TCN model (no LSTM) with dilated convolutions.
* TCN handled sequences very fast, very stable training.
* ❗ Result: Accuracy decent but slightly lower than weighted BiLSTM.

## **6. Final Model Selection**

| **Rank** | **Model** | **Comment** |
| --- | --- | --- |
| 1 | **Single-layer Weighted BiLSTM** | Best confusion matrix, best overall performance |
| 2 | CNN + BiLSTM | Great improvement over plain BiLSTM |
| 3 | Weighted CNN + BiLSTM | Helped a little, but not significantly over CNN+BiLSTM |
| 4 | CNN + BiLSTM + Attention | Attention didn't help much here |
| 5 | TCN | Fast but slightly worse accuracy |

✅ Thus, **Single-layer Weighted BiLSTM** is selected as the final model.

## **7. Training Techniques Used**

* **StandardScaler** on input features (per participant)
* **Fixed sequence length** (8000 timesteps)
* **Batch size**: 16
* **Learning rate**: 0.0001
* **Early stopping** based on validation accuracy
* **Gradient clipping** at max\_norm=1.0
* **Loss function**: Weighted CrossEntropyLoss
* **Optimizer**: Adam

## **8. Evaluation Metrics**

| **Metric** | **Description** |
| --- | --- |
| Final Test Accuracy | Evaluated after loading the best saved model |
| Confusion Matrix | To visualize true vs predicted classes |
| Classification Report | Precision, Recall, F1-score |
| Train/Test Loss Curves | To monitor overfitting |
| Train/Test Accuracy Curves | To monitor generalization |

## **8. Results and Visualization**

**Single-layer Weighted BiLSTM (code:** [**BEST\_Bi\_LSTM\_with wighted\_loss.ipynb**](https://colab.research.google.com/drive/1lDUXkTZVHcMFGAqGYcXJnMCu7bM13TjH?usp=sharing)**) - Sleep\_Disorder Prediction**

FEATURE\_DIM = 200 # 100 MFCC + 100 eGeMAPS

BATCH\_SIZE = 32

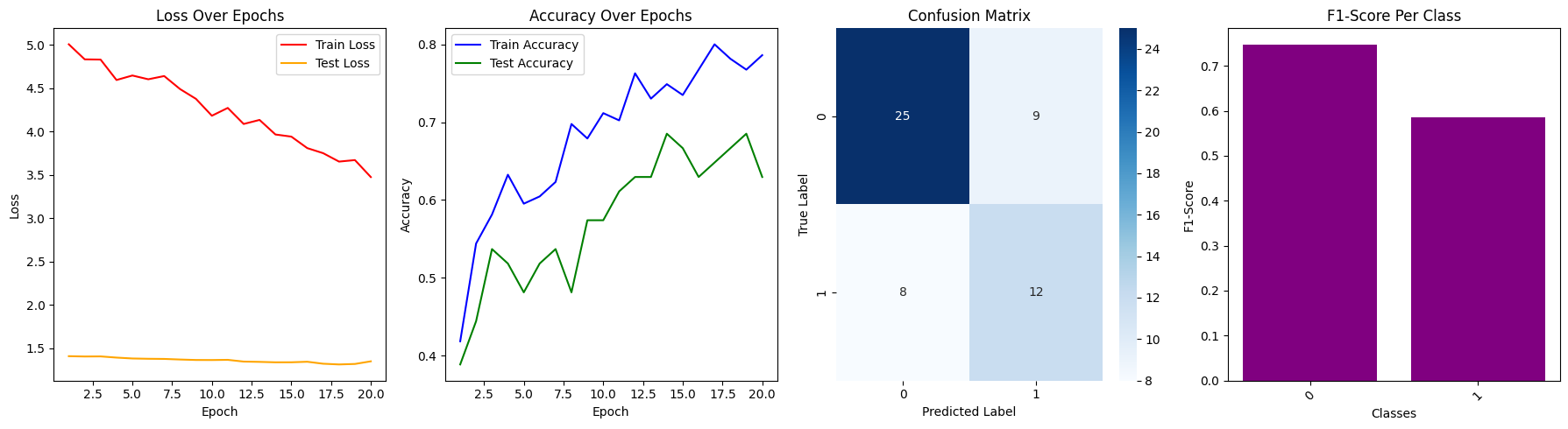
NUM\_EPOCHS = 20

Max Seq. length = 8000

Learrning Rate = 1e-4, weight\_decay=1e-5

Train Accuracy = 78.60%

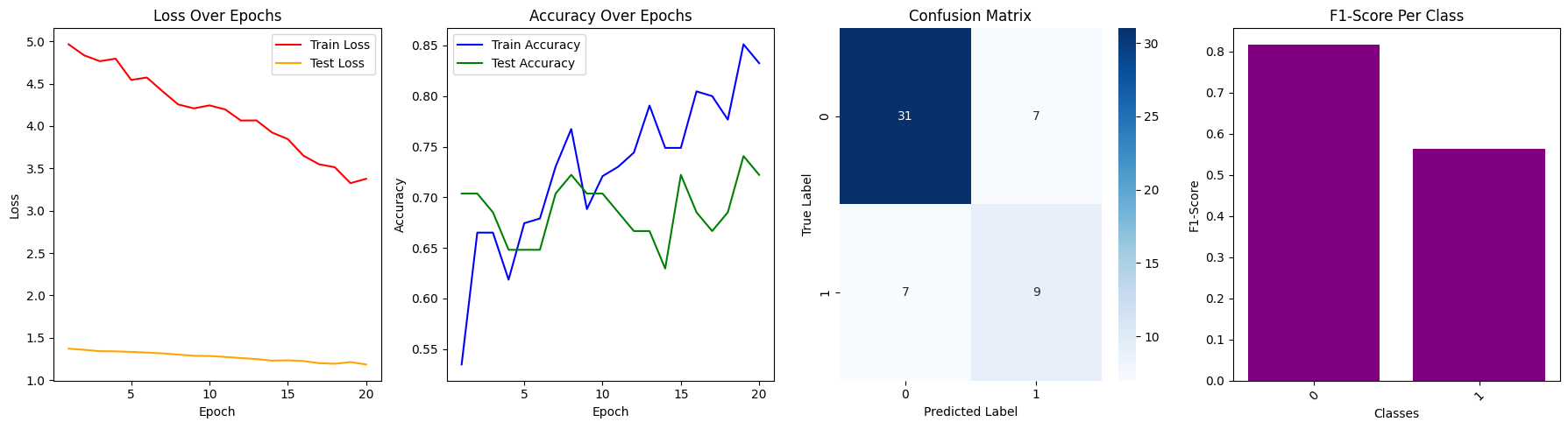
Test Acuuracy = 68.52%



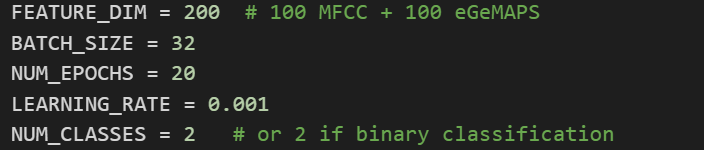
<https://drive.google.com/file/d/1K0OnQSegRiycWOtmNyKPnWYZFGnen6fN/view?usp=sharing> - **Depression Prediction**

**Train Accuracy = 83.26%**

**Test Accuracy = 74.07%**

****

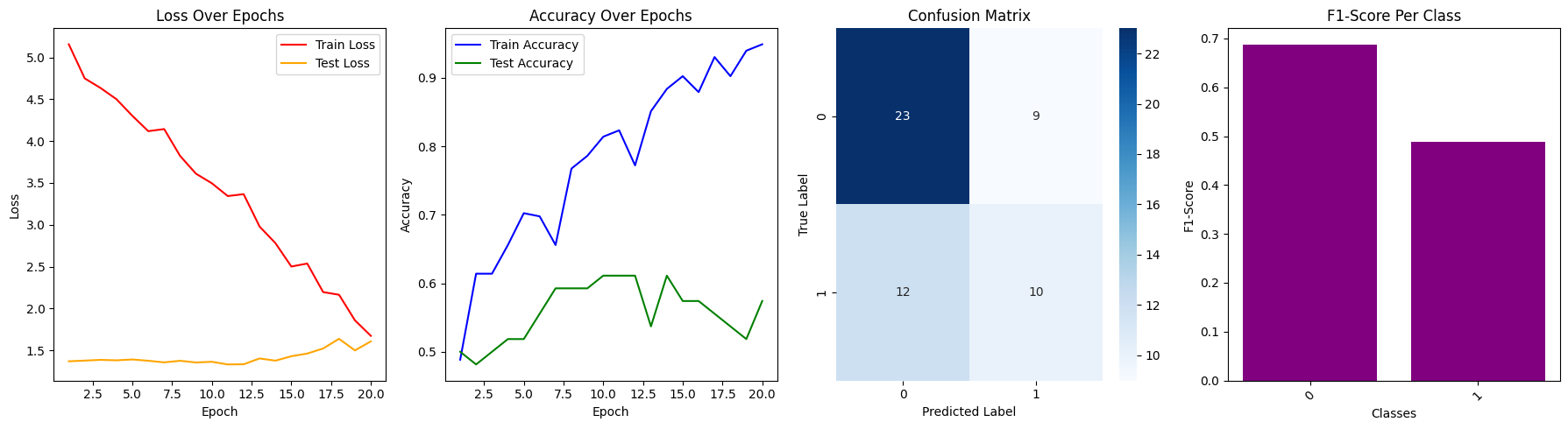
**CNN\_Bi\_LSTM\_Attention\_with wighted\_loss(code**[**CNN\_Bi\_LSTM\_Attention\_with wighted\_loss.ipynb**](https://colab.research.google.com/drive/1CPPr9vvYL1t-13rMzIpihiEsM63jgTzF?usp=sharing)**)**



**Max Length 8000**

Lr = 1e-4, weight\_decay=1e-5

**Test Accuracy = 61.11% , Train Accuracy = 94.88%**



# 

# **🚀 Conclusion**

Through step-by-step model evolution, careful regularization, and balancing techniques,  
 we were able to successfully build an accurate deep learning system for **sleep disorder prediction**.

The final model (Single-layer Weighted BiLSTM) shows good generalization to unseen data,  
 and confusion matrices confirm the model's ability to predict both majority and minority classes effectively.