

# AI2100:DEEP LEARNING PROJECT PRESENTATION

## Team Members

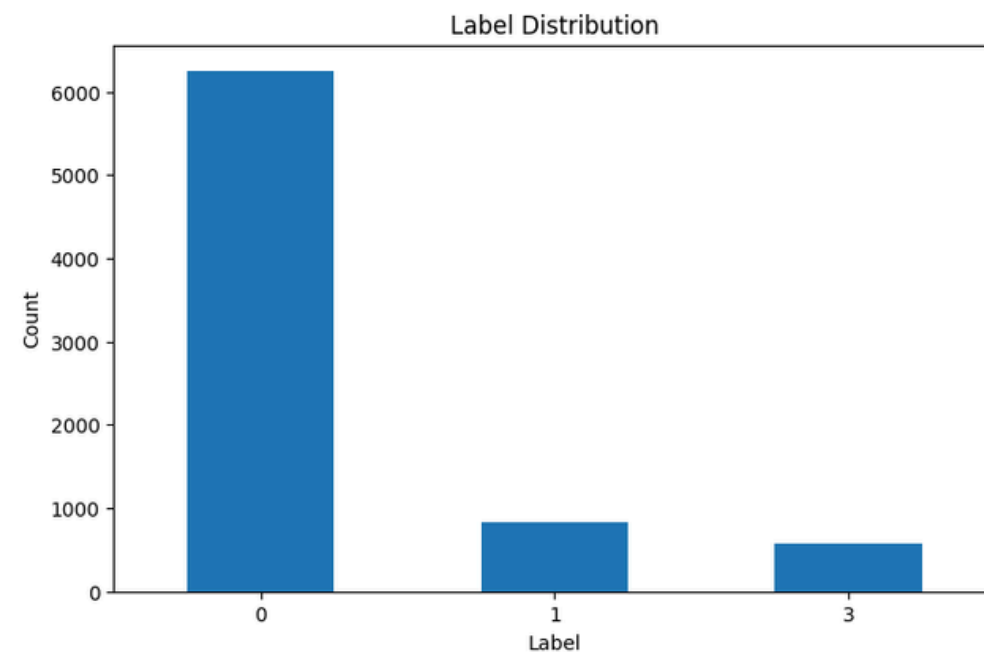
Vattivella Deexitha Reddy	- ES22BTECH11036
Nalavolu Chetana	- CS22BTECH11042
Rishitha Surineni	- CS22BTECH11050

# Problem Understanding

- Goal is to predict human activity labels using data collected from a PIR sensing node
- The dataset is collected from office and residential environments
- Time-Series Features:
  - PIR Values: 55 analog sensor readings captured over 4 seconds.
  - These values reflect how much IR radiation the sensor detects at that moment.
- Features:
  - Date: The date of the observation.
  - Time: The time of the observation.
  - Label: The target variable for classification.
  - Temperature: Ambient temperature in °F, providing contextual environmental information.
- 3 Class Labels :
  - 0: Vacancy
  - 1: Stationary human presence
  - 3: Other activity/motion

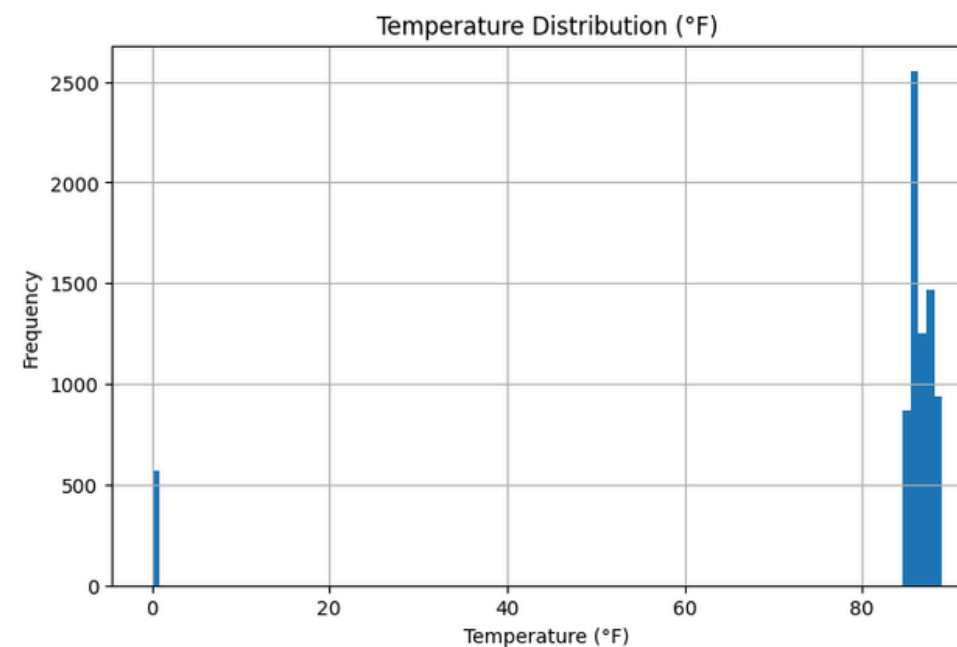
# Exploratory Data Analysis

- **Distribution of Class Labels**



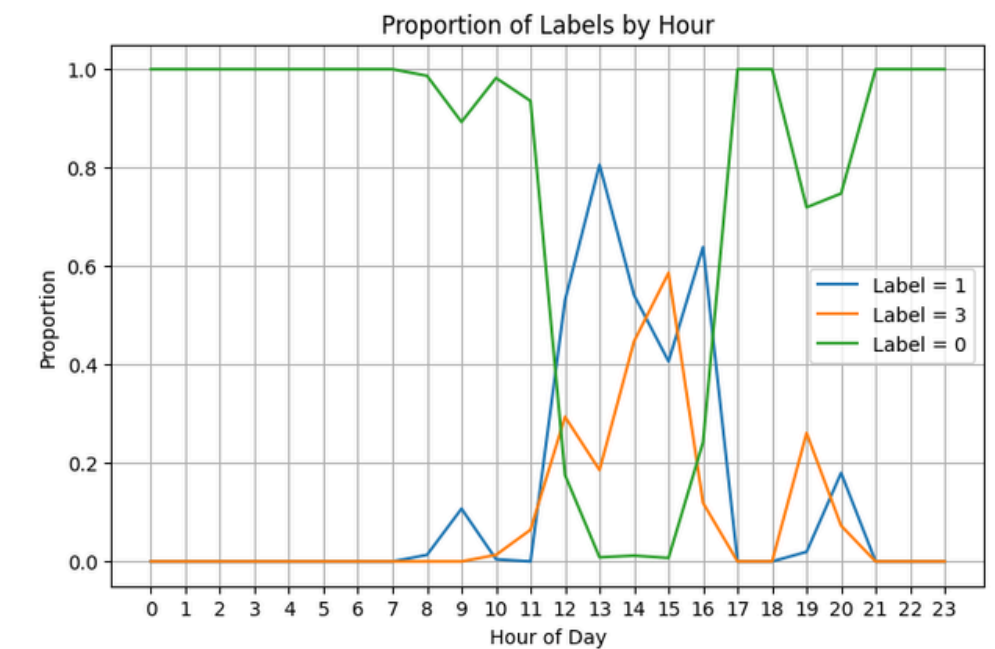
The dataset has class imbalance with majority class being Class 0

- **Distribution of Temperature**



Majority of temperature readings are clustered around room temperature

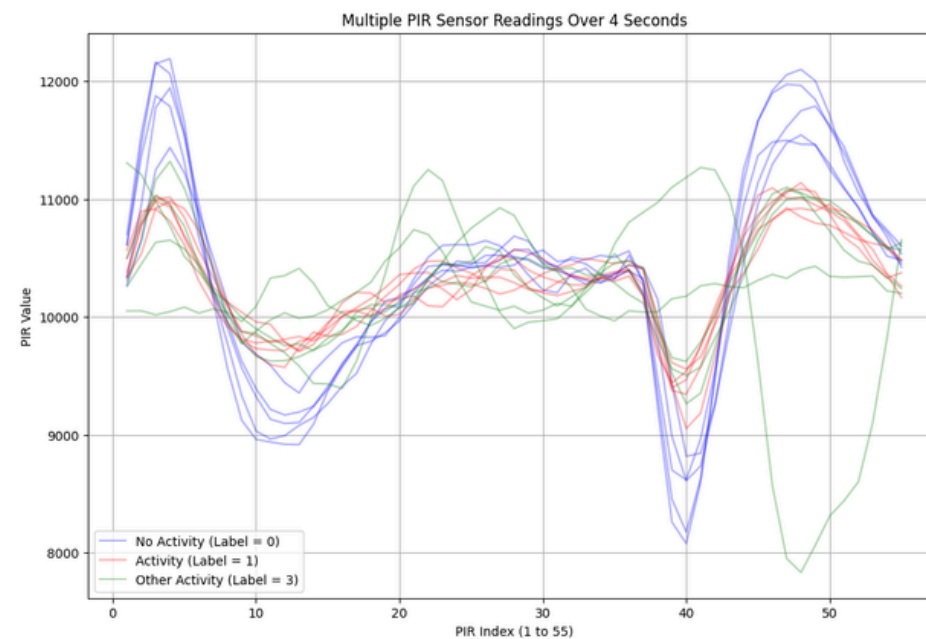
- **Proportion of Labels by Hour**



Proportion of Class 0 is highest in early morning and late evening. Class 1 and 3 indicate more activity during daytime.

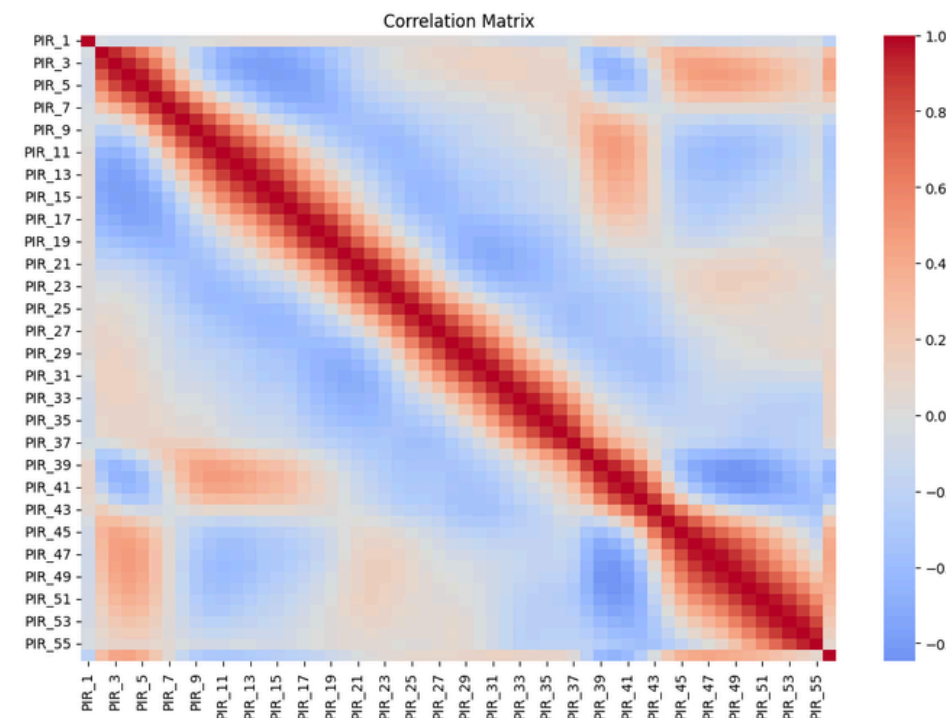
# Exploratory Data Analysis

- PIR Sensor Readings



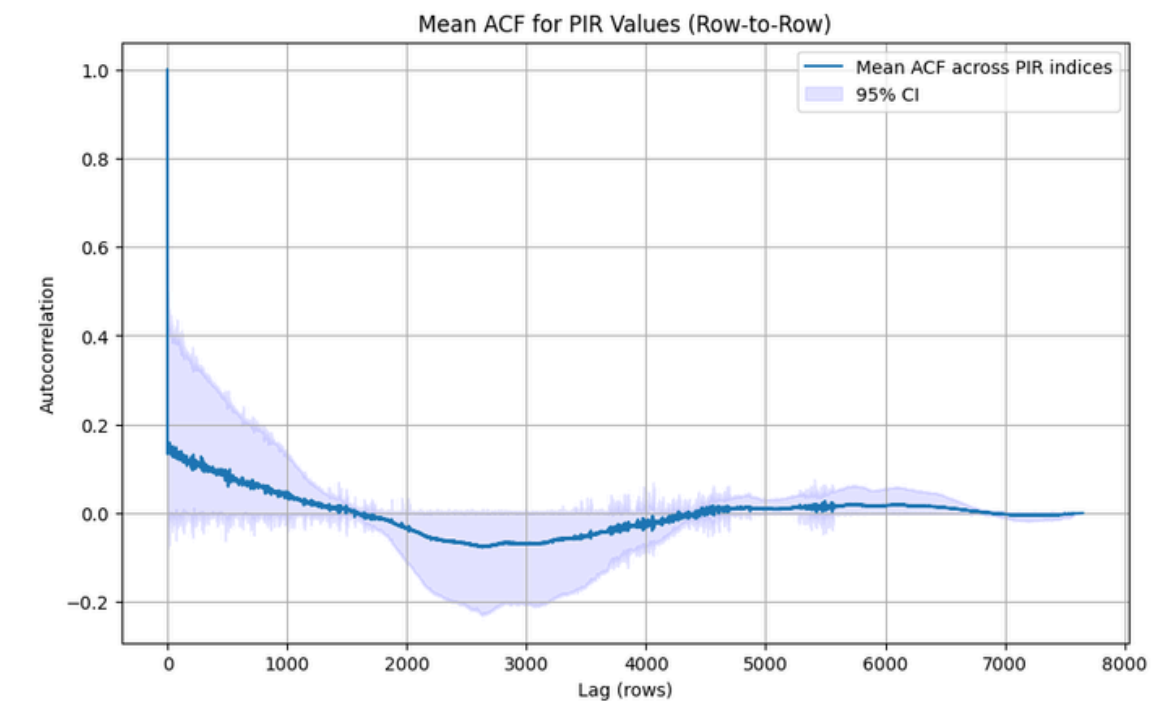
Each curve shows fluctuations over the 4-second period, with peaks and troughs at different PIR indices. Peaks are at similar indices for Class 0,1 but are shifted for Class 3

- Correlation Matrix



Consecutive sensor measurements over the 4-second window are highly dependent, reinforcing the need for sequential modeling to capture these patterns

- AutoCorrelation between rows



The Mean ACF for PIR values across rows shows a rapid decay in autocorrelation after the initial lags indicating there isn't much correlation between the rows.

# Feature Engineering and Selection

- **Time Feature Conversion:** The Time feature, originally in a string format (HH:MM:SS), was converted to seconds. This transformation brings time into a numerical scale, enabling the model to capture patterns that might correlate with activity labels.
- **Date Feature Conversion:** The Date feature was converted to the day of the week. This step aimed to incorporate weekly patterns that might influence occupancy detection.
- **SMOTE Experiment:** SMOTE was attempted to address imbalances in the dataset. However, it did not improve performance and, in some cases, degraded validation accuracy. This was likely because the PIR time-series data is structured and sequential, and SMOTE's synthetic samples introduced noise that disrupted temporal patterns.
- **StandardScaler:** All numerical features were standardized. This preprocessing step was crucial because deep learning models, particularly those with gradient-based optimization, perform better when features are on a similar scale. Standardization prevented features with larger ranges from dominating the learning process and ensured stable convergence during training.



# Model Selection & Validation



# FCNN

- **Rationale:**

- The FCNN was chosen as a simple model to give a basic understanding of the dataset's patterns. It helps set a starting point to compare with more advanced models.

- **Results:**

- mean accuracy of  **$0.9865 \pm 0.0058$**  across folds,
- Macro F1-score of **0.9777**.
- There was some misclassifications issue between classes 0 and 1, with 33 false positives for class 0 predicted as class 1.

- **Analysis:**

- As FCNNs focus more on treating data as independent features which might not be much crucial for temporal patterns. The FCNN struggled to capture the temporal dependencies in the 55-step PIR data due to its lack of sequential processing, which can lead to higher misclassification rates



# LSTM

- **Rationale:**

- The dataset contains timeseries data and the sequence length is not huge as well (just 55 timesteps). So LSTM seemed like a good place to start.

- **Results:**

- mean accuracy of  **$0.9795 \pm 0.0042$**  across folds,
- Macro F1-score of **0.9660**.
- There was some misclassification issue between classes 0 and 1, with 23 false positives for class 0 predicted as class 1.

- **Analysis:**

- The performance of LSTM was abit lower than some of the other models we tried. The reason for that could be LSTM focuses more on long-term memory, which is less critical for sequence where local temporal patterns dominate, as shown by the correlation matrix. Its sequential processing nature may struggle to efficiently capture nearby relations. Additionally, the number of samples might be insufficient to properly train the LSTM's numerous parameters.



# CNN

- **Rationale:**

- The CNN was chosen because the correlation matrix revealed strong short-term patterns in the PIR data. CNNs effectively capture these local relationships for activity classification.

- **Results:**

- Mean accuracy of  **$0.9902 \pm 0.0043$**  across folds,
- Macro F1-score of **0.9836**.
- There was some misclassification issue between classes 0 and 1, with 5 false positives for class 0 predicted as class 1.

- **Analysis:**

- CNNs effectively capture local temporal patterns, which are crucial for the PIRvision dataset, as shown by strong short-term correlations in the correlation matrix. This led to minimal misclassifications.

# TCN

- **Rationale:**

- On the success of CNNs in capturing short-term patterns in the PIR data, the TCN was tested as another variation using dilated convolutions.

- **Results:**

- mean accuracy of  **$0.9838 \pm 0.0044$**  across folds,
- Macro F1-score of **0.9734**.
- There was some misclassifications issue between classes 0 and 1, with 20 false positives for class 0 predicted as class 1.

- **Analysis:**

- The performance of TCN was abit lower than some of the other models we tried. The reason for that could be its dilated convolutions, as expanding the receptive field, may not optimally align with the short-term dependencies highlighted by the correlation matrix.

# Transformer

- **Rationale:** The Transformer model was explored as an alternative sequential modeling approach. Motivated by its proven effectiveness and widespread adoption this felt like a logical next step in our experimentation.
- **Implementation:**
  - **Hyperparameter tuning:** Optuna was used to optimize hyperparameters. The best configuration was:
    - d\_model: 128
    - nhead: 16
    - num\_layers: 4
    - dim\_feedforward: 97
    - dropout: 0.143
    - lr: 0.000114
- **Results:**
  - Validation accuracy of **99.70% ± 0.17%** and a Macro F1-score of **0.9971**.
  - Test set accuracy was **99.02%**, with a Macro F1-score of **0.9825**.
- **Analysis:**
  - Its self-attention mechanism, with quadratic complexity ( $O(n^2)$ ) relative to sequence length, may overcomplicate the modeling of a moderate 55-timestep sequence. And the number of samples might be insufficient to fully optimize the Transformer's complex parameters.



# Results and Insights

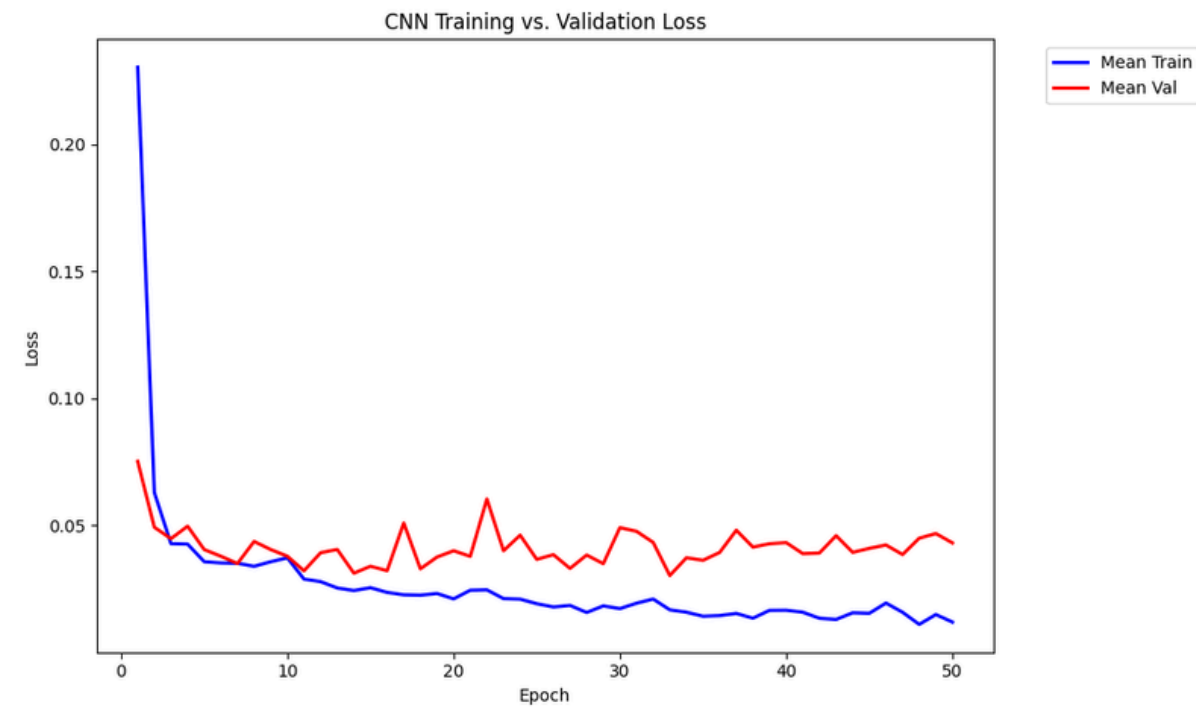


# Results

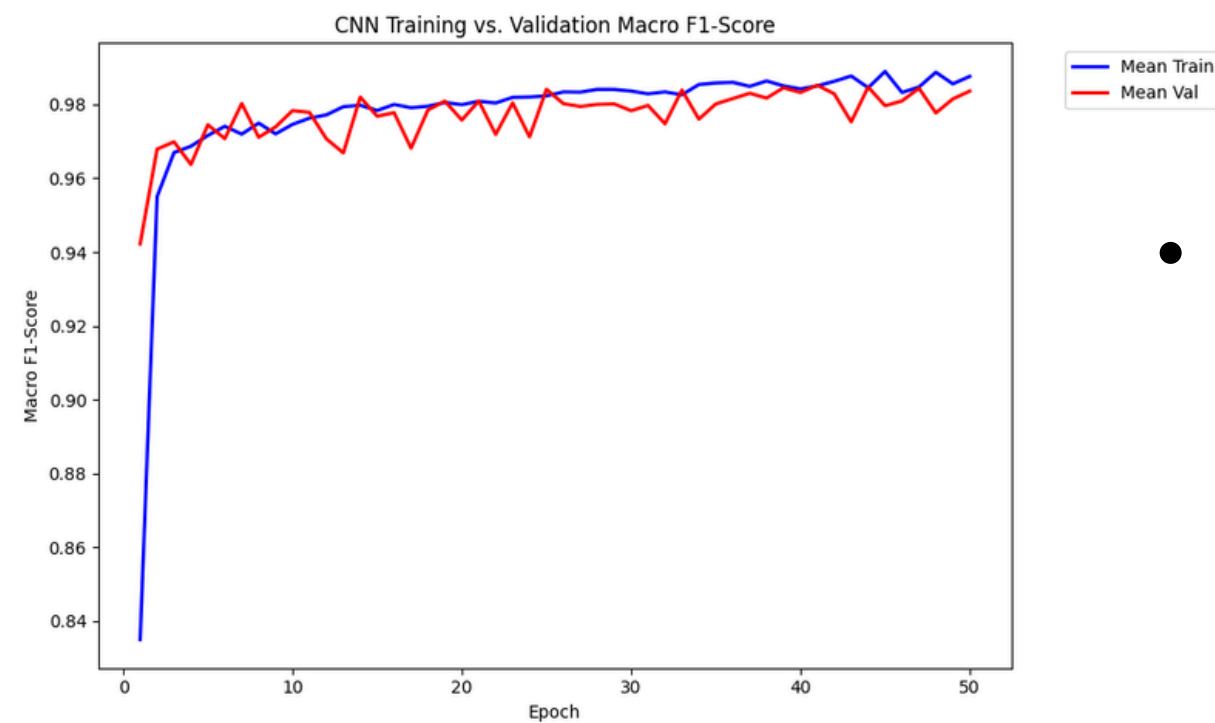
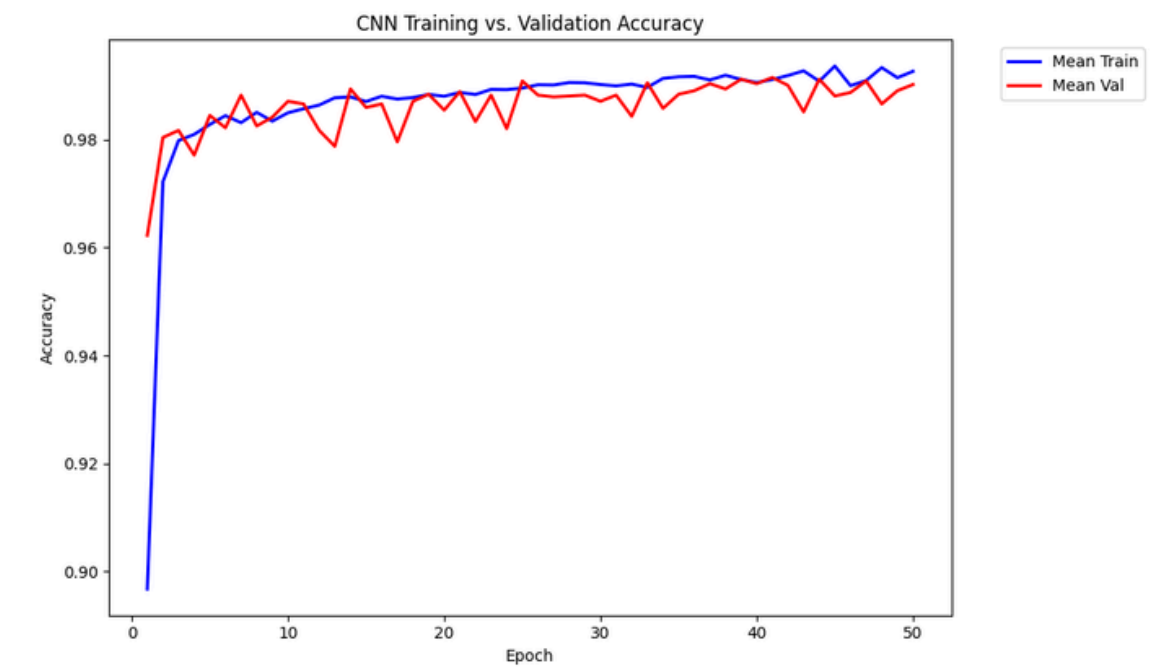
- We chose CNN as our final model.
- Validation Results :
  - Mean accuracy of  $0.9902 \pm 0.0043$  across folds,
  - Macro F1-score of 0.9836.
  - Class 0: Precision=0.9982, Recall=0.9898, F1=0.9940
  - Class 1: Precision=0.9303, Recall=0.9865, F1=0.9569
  - Class 2: Precision=1.0000, Recall=1.0000, F1=1.0000
- Test Results :
  - Mean Accuracy of  $0.9928 \pm 0.0039$
  - Mean Macro F1-Score: 0.9879
  - Class 0: Precision=0.9992, Recall=0.9920, F1=0.9956
  - Class 1: Precision=0.9440, Recall=0.9940, F1=0.9682
  - Class 2: Precision=1.0000, Recall=1.0000, F1=1.0000

# Plots

- Training and Validation Loss over epochs



- Training and Validation Accuracy over epochs



- Training and Validation Macro F1-Score over epochs

# Why CNN?

- **Superior Local Feature Extraction** : The PIR signals over 4 s are highly correlated in nearby sensor indices (according to correlation matrix). 1D convolutions with small kernels naturally exploit those local dependencies via weight-sharing and locality.
- **Less Parameters** : CNN has far fewer parameters than a Transformer or even a fully-connected net of comparable depth. This strong architectural bias against overfitting is especially valuable on a modest-sized dataset.
- **Computational Efficiency**: The CNN's processing of convolutions makes it more computationally efficient than sequential models like LSTM or complex models like the Transformer.
- **Better Generalisation** : Its simpler architecture, combined with dropout, ensures better generalization, even with a potentially limited number of samples.




# Challenges Faced

- **Overfitting:** Models, such as the LSTM and transformers, achieved very high training accuracy but showed lower validation accuracy, suggesting they were overfitting to the training data.
- **Hyperparameter Tuning:** Models had multiple hyperparameters and finding the best settings was not easy.
- **Interpretability:** Deep models were hard to interpret, making it challenging to understand why certain predictions were made or why errors occurred, especially for misclassifications between classes 0 and 1.
- **Skewed Data:** A baseline model that always predicted class 0, without any learning, achieved 81% accuracy, indicating the dataset was highly skewed toward one class. Attempts to generate additional data to balance the dataset did not give significant improvement.

# Future Improvements

- **Increase Dataset Size and Diversity:** The current dataset has less samples. Collecting more samples and including more diverse ones would improve the generalization of the model.
- **Time Series Augmentation:** Data augmentation techniques specific to time-series data, such as adding controlled noise or jitter to the 55-timestep PIR sequences can be used, to simulate real-world variations This could enhance the model's resilience to noise and improve its ability to handle unseen patterns.
- Explore Hybrid Models



**THANK YOU**