Enhancing Time-series Classification Through Self-Supervised Learning

A simplified approach to understanding complex data

Submitted By

Pragya Gupta

Background

Problem Statement:

- Traditional supervised learning models require large amounts of labeled data. However, labeling is often expensive and time-consuming.

Challenges:

- Scarcity of labeled data for gesture recognition.
- High intra-class variability and inter-class similarity.
- Can we learn meaningful patterns without relying on labeled data?

Why Self-Supervised Learning?

- Self-supervised approaches like autoencoders can leverage unlabeled data to learn meaningful representations.

Objective

Develop a robust gesture recognition pipeline by leveraging self-supervised learning(minimum labeled data) for feature extraction.

Specific Objectives

Train an autoencoder to learn latent representations.

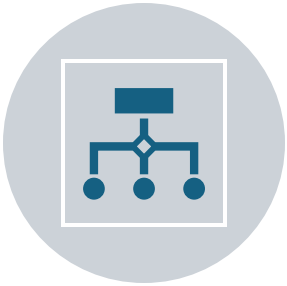
Evaluate the effectiveness of these features using various classifiers (Logistic Regression, SVM, Random Forest).

Validate performance on a gesture test set and assess transfer learning capabilities.

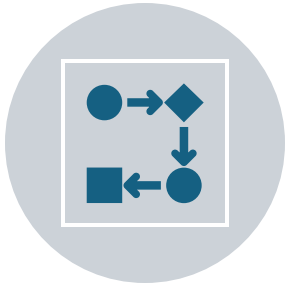
Methodology

Step 1: Preprocess time-series gesture and HAR data.

Step 2: Train an **Autoencoder** to extract key patterns using HAR data.



Step 3: Use these patterns (latent features) for classification.



Step 4: Validate on gesture data and refine the model.

Technical Overview: Decoding the Autoencoder

Brief Overview

- **Autoencoder Architecture**: Encoder learns to compress data into meaningful patterns (latent space). Decoder reconstructs the original data from latent features.
- **Classifier Integration**: Latent features are used as input for traditional classifiers like SVM and Random Forest.

Why Autoencoders?

- They provide a compressed representation of high-dimensional data.
- Enable downstream tasks like classification and anomaly detection.
- Helps discover hidden patterns without labeled data.

Data Insights: Knowing Data is Important!!

HAR Dataset: Used for self-supervised pre-training.

Gesture Dataset: Used for classification.

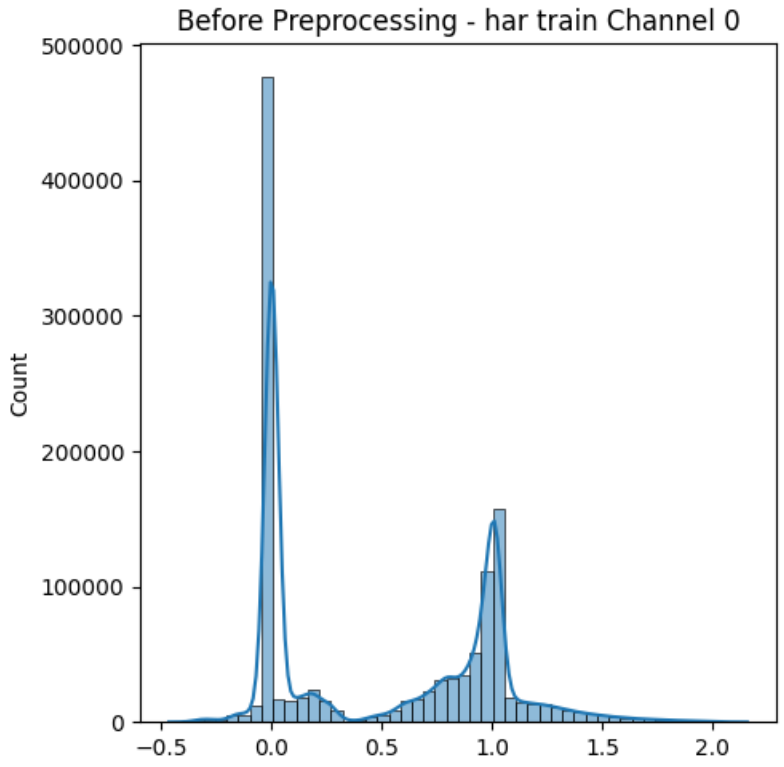
Key Insights

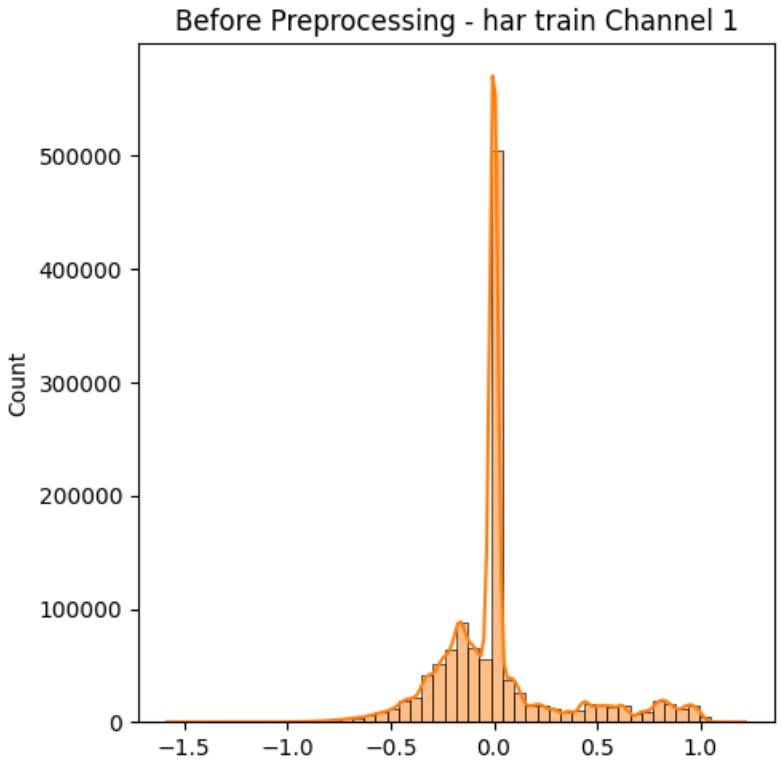
- HAR's Channel 1 has a mean close to 0.5, while Gesture is centered near 0.
- Gesture has a wider range compared to HAR.

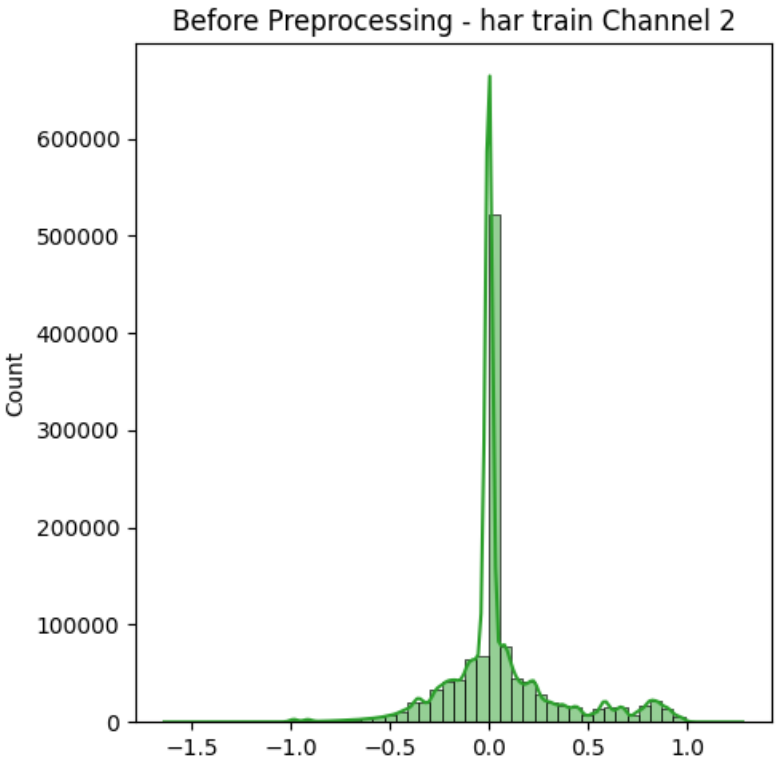
Preprocessing Need:

- Normalization/Standardization: Necessary to bring both datasets to a similar scale, especially since gesture data spans a larger range.
- Consistent scaling across datasets will help the model generalize better.

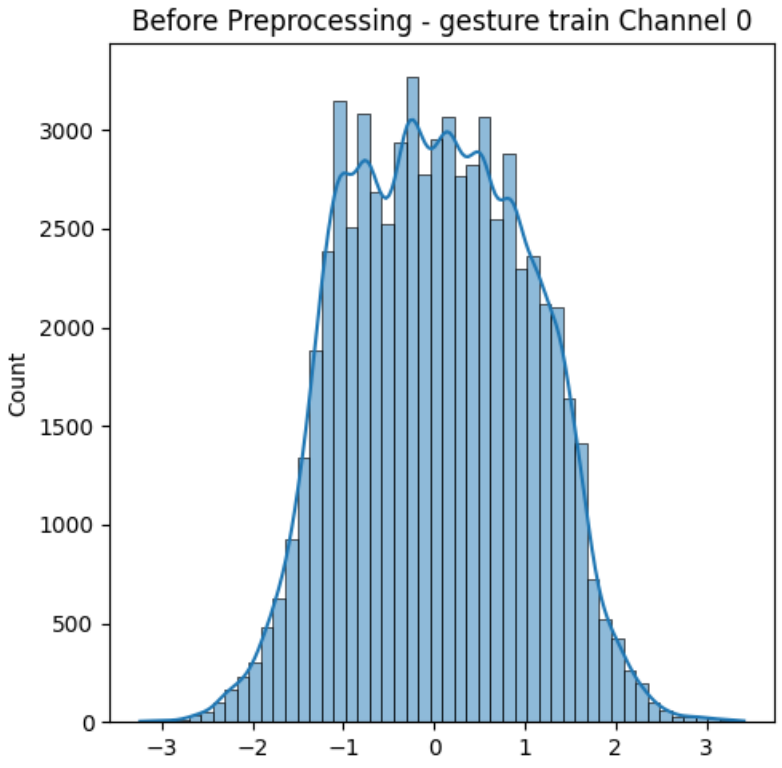
Exploratory Data Analysis (EDA)

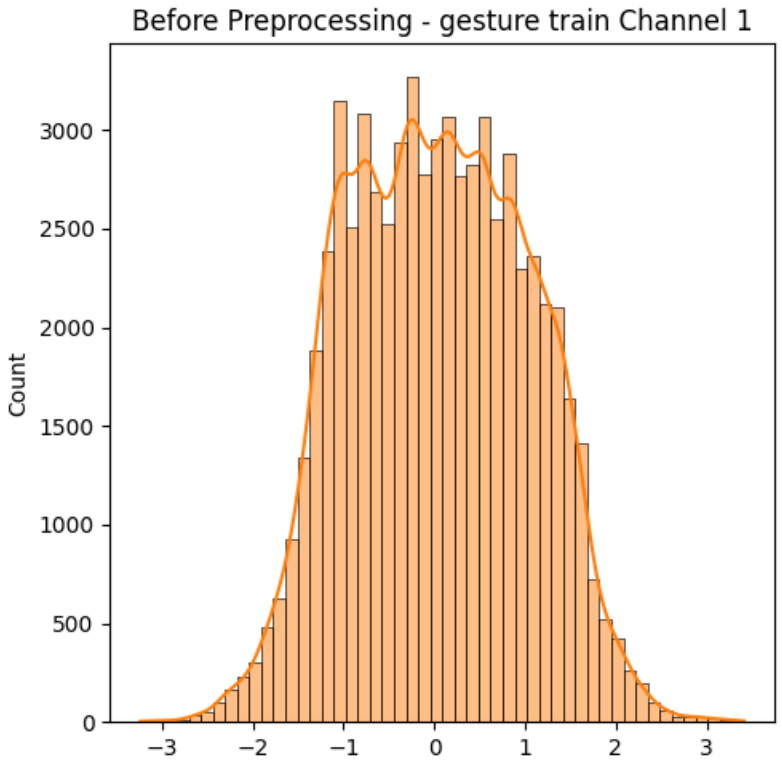


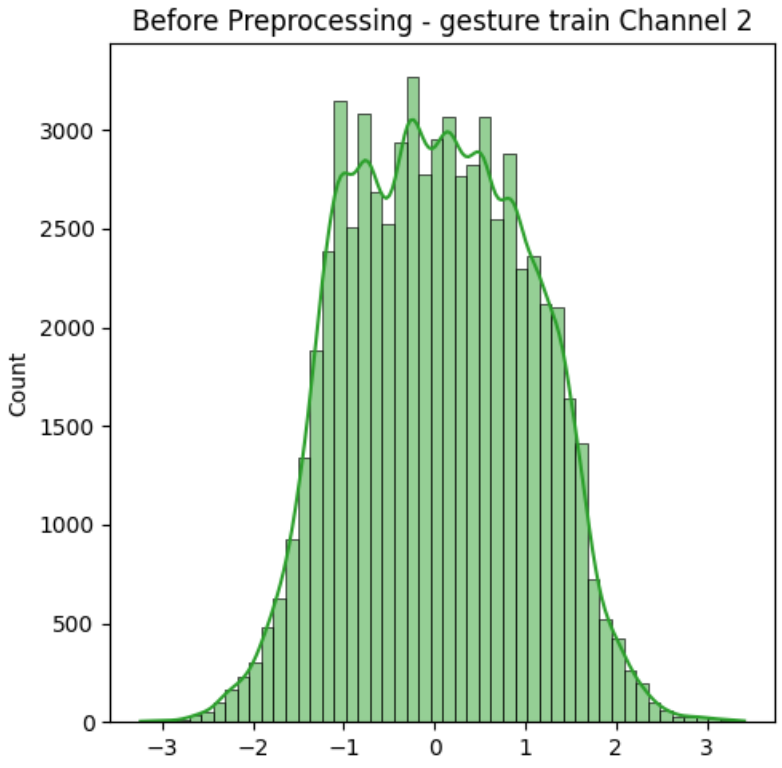




EDA


Before Preprocessing - gesture train Channel 0
Count
0
500
1000
1500
2000
2500
3000
−3
−2
−1
0
1
2
3


Before Preprocessing - gesture train Channel 1
Count
0
500
1000
1500
2000
2500
3000
−3
−2
−1
0
1
2
3


Before Preprocessing - gesture train Channel 2
Count
0
500
1000
1500
2000
2500
3000
−3
−2
−1
0
1
2
3

Digging deeper into the Data

HAR Dataset:

Channel 0:

- Multimodal distribution with three distinct peaks, suggesting multiple underlying patterns or activities. Right-skewed, indicating a longer tail towards higher values. Potential outliers at the extreme ends.

Channel 1:

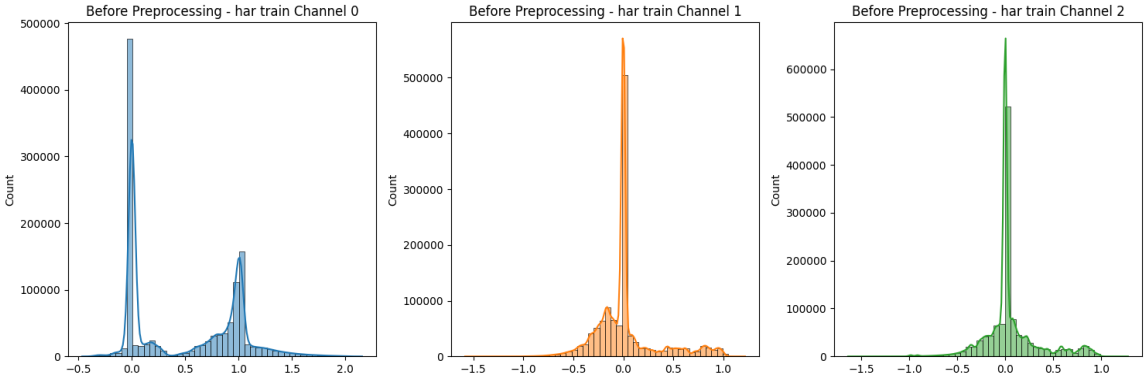
- Multimodal distribution with three distinct peaks, suggesting multiple underlying patterns or activities. Right-skewed, indicating a longer tail towards higher values. Potential outliers at the extreme ends.

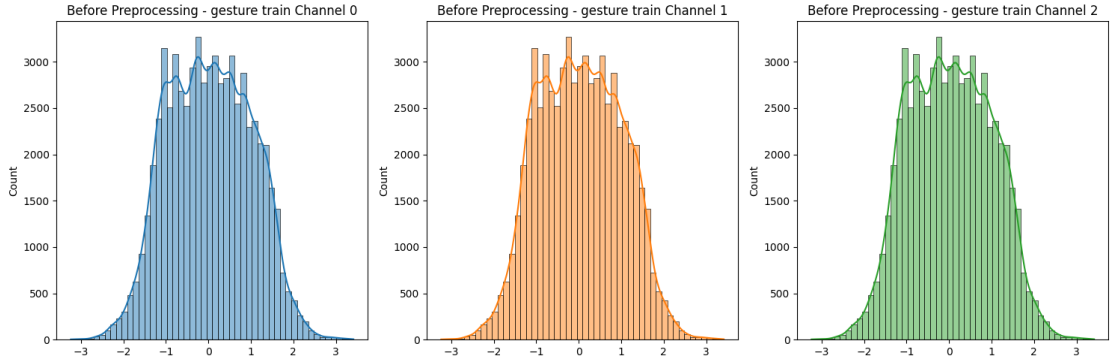
Channel 2:

- Unimodal distribution with a single peak, indicating a relatively homogeneous distribution. Right-skewed, indicating a longer tail towards higher values. Potential outliers at the extreme ends.

Gesture Dataset:

Channel 0, 1, and 2: Approximately normally distributed, indicating a bell-shaped curve with data centered around a specific value. Slight right-skewness, indicating a slightly longer tail towards higher values. Potential outliers at the extreme ends.

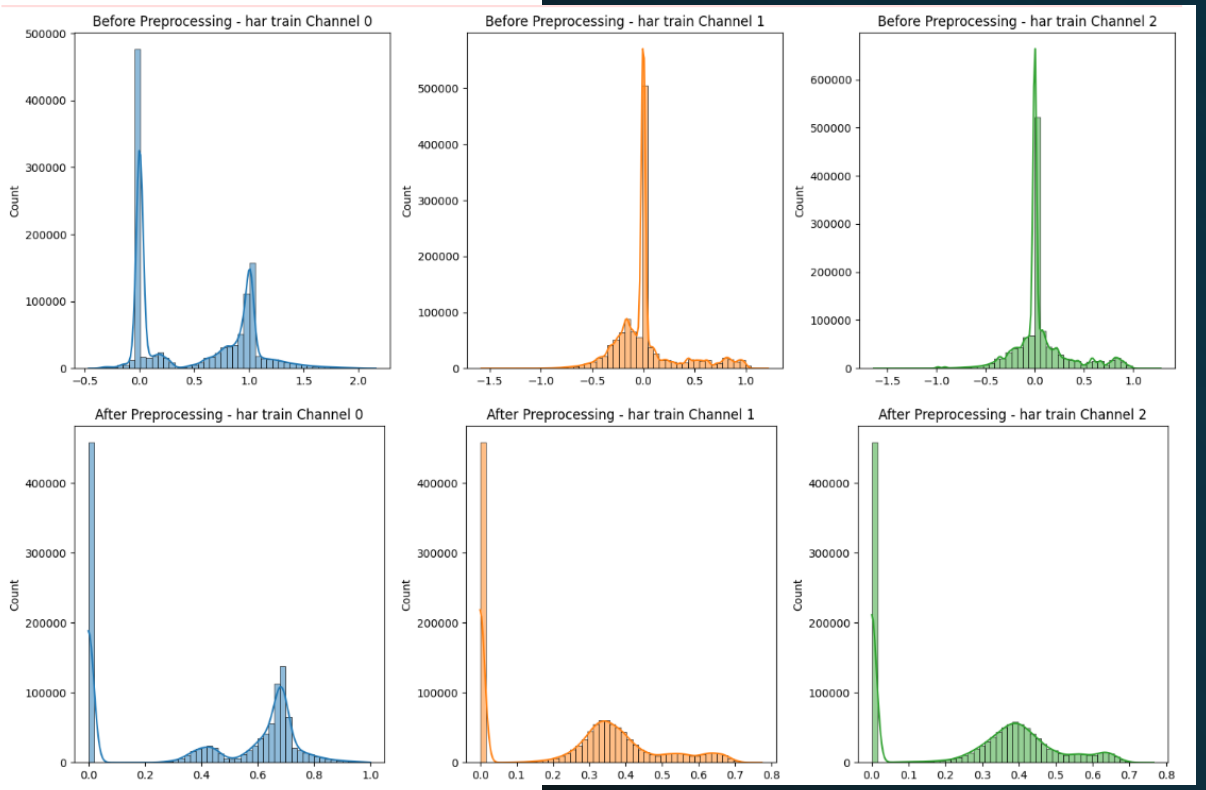




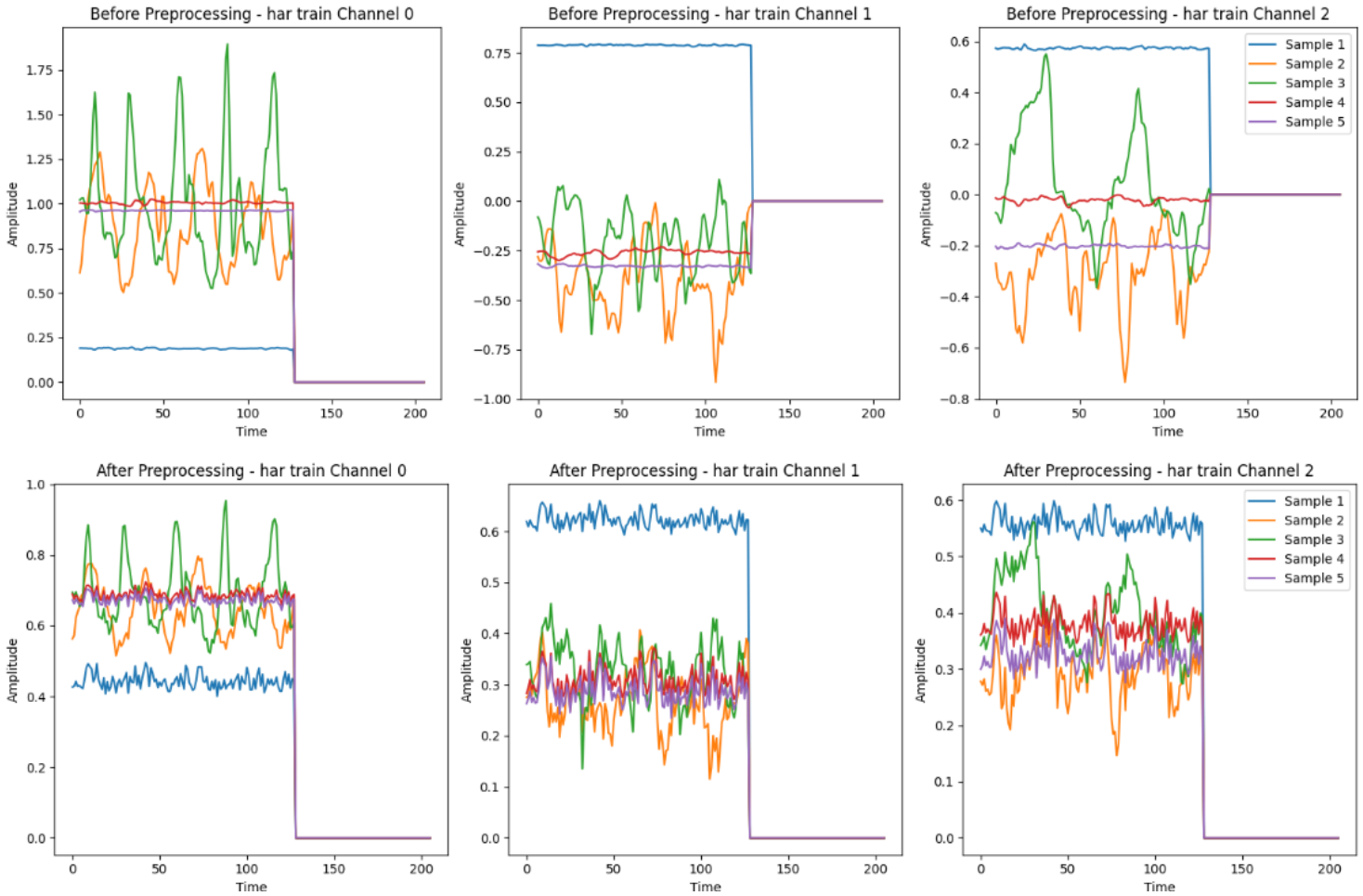
Preprocessing Steps

- Peak-Based Segmentation (HAR only).
- Min-Max Scaling (HAR and Gesture).
- IQR-Based Outlier Handling.

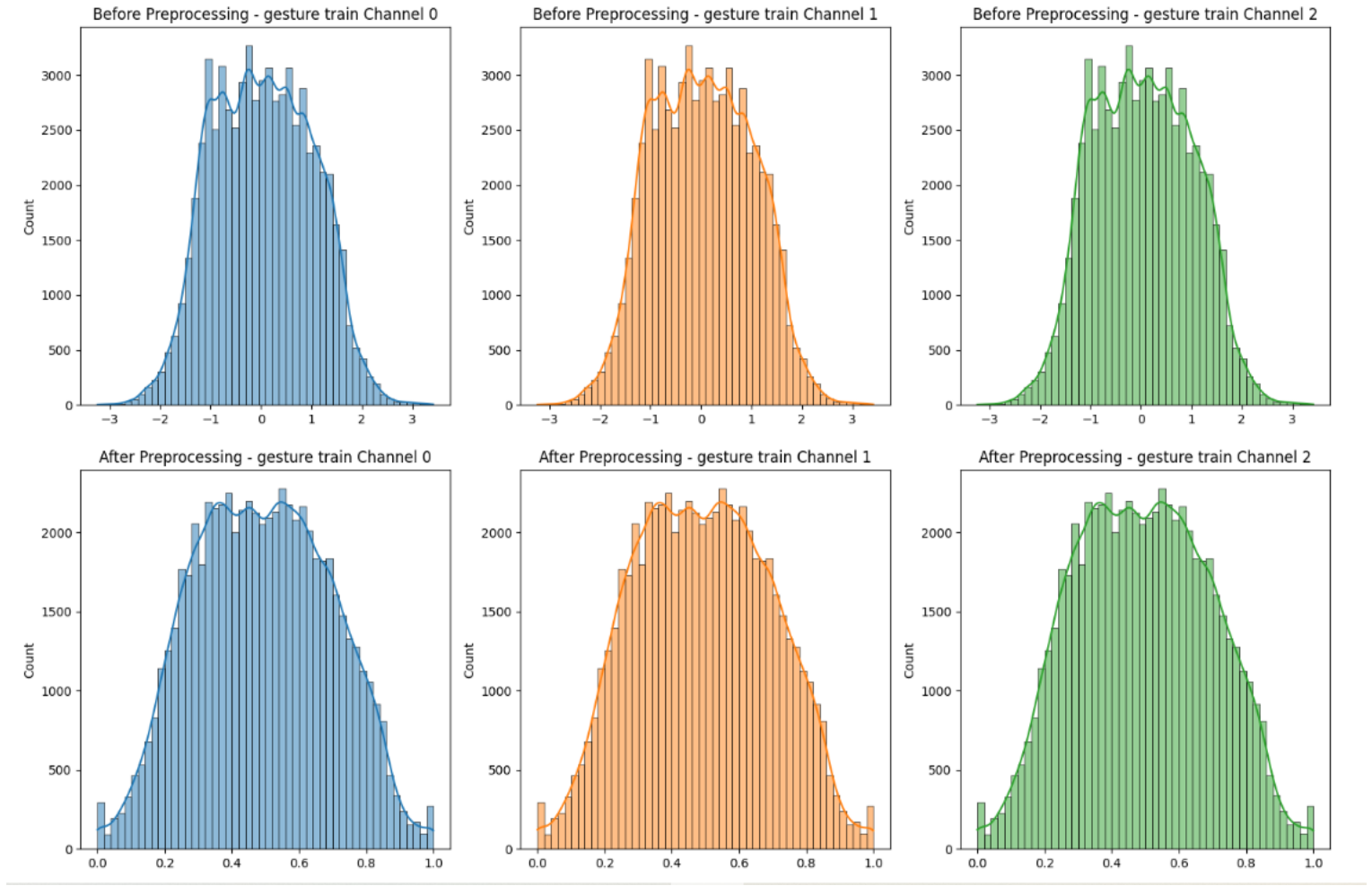
Applied preprocessing to both training and validation data.



Preprocessing Results



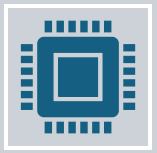
Preprocessing Result



Model Selection

We aim to utilize Self-Supervised Learning (SSL) to improve the performance of classification models on time series data.

Contrastive Learning is a powerful self-supervised technique that learns useful feature representations without needing labeled data. The method works by bringing similar data points closer together in feature space and pushing dissimilar points apart. This is particularly beneficial for time series data, as it can capture subtle patterns, leading to robust features for downstream tasks like classification.



But it was computationally intensive. So we switched to Autoencoders and decided to come back to contrastive learning later.

Model Architecture

- Input → Dense Layers (Encoder) → Latent Space → Dense Layers (Decoder) → Output.
- **Latent Dimension:** 64 features capturing gesture patterns.

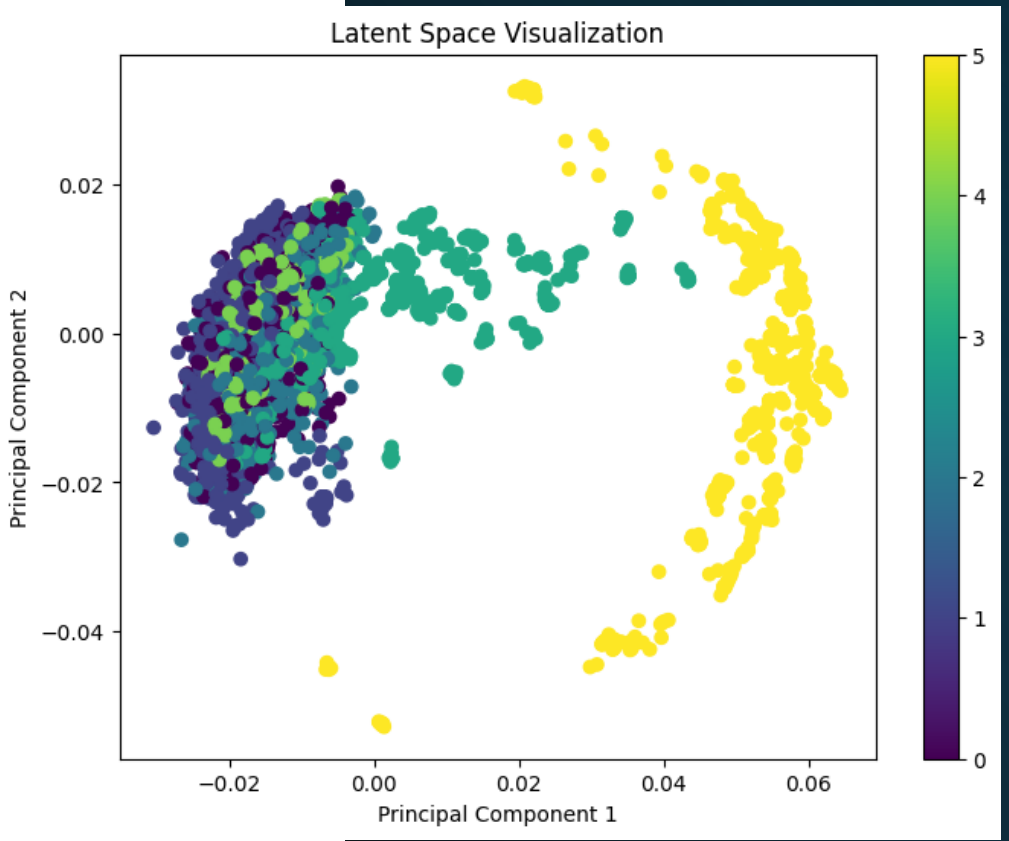
Training Results

```
Training Autoencoder...
Epoch 1/10, Loss: 0.0070, Reconstruction Accuracy: 0.4308
Epoch 2/10, Loss: 0.0014, Reconstruction Accuracy: 0.5946
Epoch 3/10, Loss: 0.0012, Reconstruction Accuracy: 0.6349
Epoch 4/10, Loss: 0.0012, Reconstruction Accuracy: 0.6524
Epoch 5/10, Loss: 0.0009, Reconstruction Accuracy: 0.6773
Epoch 6/10, Loss: 0.0009, Reconstruction Accuracy: 0.6886
Epoch 7/10, Loss: 0.0009, Reconstruction Accuracy: 0.6881
Epoch 8/10, Loss: 0.0008, Reconstruction Accuracy: 0.7008
Epoch 9/10, Loss: 0.0007, Reconstruction Accuracy: 0.7055
Epoch 10/10, Loss: 0.0007, Reconstruction Accuracy: 0.7104
Autoencoder training completed in 27.21 seconds.
```

Training Results and Latent features Visualization

Summary of our Observations:

- **Distinct Clusters:** This is a good indication that the latent space is capturing meaningful patterns.
- **Cluster Separation:** Indicates that the model is able to map similar data points close to each other in the latent space, which is an important property for tasks like classification.
- **Color Coding:** Similar colored points cluster together, it suggests that the autoencoder is preserving class information within the latent space. This could make it easier for downstream models to classify the data based on these representations.



Evaluation metrics

- **Accuracy**: Overall correctness of predictions.
- **Precision**: Measure of relevant instances retrieved.
- **Recall**: Measure of true positive rate.
- **F1-Score**: Harmonic mean of precision and recall.
- **Confusion Matrix**: Provides a detailed view of classification performance per class.

Rationale: These metrics ensure a holistic evaluation, particularly in imbalanced datasets.

Validation and Results

- Overall accuracy: **58%**.
- Classes **1, 2, 3, and 7** had good performance (f1-scores > 0.60).
- Significant misclassifications in **Class 4 and 5**, suggesting a need for better feature separation.

Classification Report on Gesture Test Set:

	precision	recall	f1-score	support
0.0	0.55	0.70	0.62	40
1.0	0.62	0.85	0.72	40
2.0	0.61	0.90	0.73	40
3.0	0.48	0.85	0.61	40
4.0	0.00	0.00	0.00	40
5.0	0.50	0.05	0.09	40
6.0	0.70	0.57	0.63	40
7.0	0.66	0.72	0.69	40
accuracy			0.58	320
macro avg	0.51	0.58	0.51	320
weighted avg	0.51	0.58	0.51	320

Confusion Matrix on Gesture Test Set:

```
[[28  6  0  1  1  1  2  1]
 [ 1 34  1  0  0  1  2  1]
 [ 1  0 36  0  0  0  3  0]
 [ 1  2  0 34  0  0  0  3]
 [ 2  5  7 17  0  0  3  6]
 [ 6  5 11 12  0  2  0  4]
 [ 7  3  4  1  2  0 23  0]
 [ 5  0  0  6  0  0  0 29]]
```

Insights from Classification

- **Overall Accuracy:** 58% on the gesture test set. While decent, there is room for improvement in handling specific classes.
- **Strong Class Performance:** Class 0, 1, 2, 3, and 7 , (f1-scores ranging from 0.61 to 0.73) i.e. the model captures these patterns effectively.
- **Weak Class Performance:** Class 4 and 5, (f1-scores near 0) . This indicates that the model struggles to distinguish these classes, likely due to insufficient latent space representation or overlap with other classes.
- **Confusion Observations:** Class 4 was frequently misclassified as Class 3 or 7, showing a lack of clear boundary in the latent space. Class 5 showed significant misclassification across multiple classes, which highlights instability in recognizing this class.
- **Cluster Interpretation:** The confusion matrix suggests that some classes (e.g., 4, 5) may need further refinement in the latent space to better separate their patterns.

Conclusion

- **Summary**:
- Self-supervised autoencoder successfully learned latent representations.
- Moderate performance on test sets highlights strengths and weaknesses.
- Transfer learning to gesture data showed potential but requires further tuning.
- **Proposed Future Directions**:
- Experiment with **contrastive learning** to improve latent space separation.
- Fine-tune hyperparameters of the autoencoder.
- Investigate ensemble methods for improving classification accuracy.

Technical Learnings

AUTOENCODERS CAN EFFECTIVELY LEARN REPRESENTATIONS WITH MINIMAL LABELS.

CLASSIFIERS' PERFORMANCE IS HEAVILY DEPENDENT ON THE QUALITY OF LEARNED FEATURES.

SVM AND RANDOM FOREST SHOWED PROMISE BUT REQUIRE OPTIMIZED HYPERPARAMETERS.

Thank you