Comparative Analysis of Models for 3D Human Motion Classification

Evaluating DSVM, LSTM, and STGCN on Motion Capture Data

Jaroslav Kvasnička
Institute of Pervasive Computing
Johannes Kepler University
Linz, Upper Austria, Austria
k12140156@students.jku.at

ABSTRACT

This study compares three machine learning models: Dual Support Vector Machine (DSVM), Long Short-Term Memory (LSTM), and Spatial Temporal Graph Convolutional Network (ST-GCN) for 3D human activity recognition using skeletal motion data. The dataset includes time series of 3D joint coordinates for various activities. A K-Means clustering step was used to extract representative labels for classification.

Results show that DSVM achieved high accuracy with minimal training time and a very small model size, outperforming expectations. LSTM also delivered strong accuracy but required significantly more time and memory. ST-GCN, despite its complexity, underperformed, consistently predicting only one class and resulting in low overall accuracy. These findings suggest ST-GCN may be better suited to continuous video input rather than raw 3D skeleton data.

The confusion matrices further demonstrate the reliable classification by DSVM and LSTM across all classes, while ST-GCN failed to distinguish between them. Additionally, the choice and preprocessing of data—particularly the use of K-Means—proved crucial to overall model success.

The study highlights that model selection should be guided by both data structure and application context, with DSVM offering a strong baseline for simple and efficient human activity recognition.

KEYWORDS

Human Activity Recognition, Support Vector Machine (SVM), LSTM, Spatial Temporal Graph Convolutional Networks (ST-GCN), K-Means Clustering, Activity Classification, Model Comparison, Performance Evaluation, Feature Engineering

1. Introduction

Human Activity Recognition (HAR) plays an important role in industrial and occupational safety by enabling continuous monitoring of worker posture and movement. Accurate identification of postures yields valuable insights into biomechanical load, ergonomic risk, and adherence to safety protocols.

With recent advances in marker-based motion capture and machine learning, automated HAR systems now offer high-resolution, data-driven analysis of human motion. In this study, we employ an OptiTrack setup to record three-dimensional joint trajectories during repeated trials of target postures, like squatting, sitting, and standing. The resulting spatial–temporal data serve as input to machine learning classifiers.

To evaluate the performance of different modeling paradigms on this task, we conduct a comparative analysis of three machine learning approaches: Dual Support Vector Machines (DSVM), Long Short-Term Memory networks (LSTM), and Spatio-Temporal Graph Convolutional Networks (ST-GCN). These models are chosen for their dis-

tinct capabilities in handling structured, temporal, and graph-based representations of human motion. Specifically, DSVM serves as a baseline for traditional classification; LSTM captures sequential dependencies in joint movements; and ST-GCN leverages the skeletal graph structure to learn spatial-temporal features jointly.

Each model was trained and evaluated on the same OptiTrack-derived dataset, with a focus on classification accuracy, training time, and class-specific performance for the three postures. The findings aim to inform model selection for real-world HAR applications in industrial contexts, where both accuracy and efficiency are critical.

We hypothesize that all three models will achieve comparable accuracy in classifying these postures (stand, sit, squat). However, we expect that LSTM and ST-GCN models will require significantly longer training times due to their increased architectural complexity, in contrast to the lighter-weight DSVM.

The paper is organized as follows: Section 2 reviews existing work on HAR and machine learning models for motion data; Section 3 briefly describes the data collection process and preprocessing pipeline; Section 4 describes the architecture and implementation of the evaluated models; Results are presented in Section 5; and Section 6 concludes with a discussion and potential directions for future research.

2. Related Work

2.1. Support Vector Machines (SVM)

Are supervised learning models are used for binary classification to calculate the optimal hyperplane that separates two classes in the feature space. In human activity recognition, each action (for example, walking, sitting, or waving) is translated into a set of numerical measurements. The SVM finds the line or surface in this high-dimensional feature space that maximizes the margin between activities, helping to distinguish even subtle differences. [8]

2.2. Long Short-Term Memory (LSTM)

Networks are a class of recurrent neural networks (RNNs) that excel at modeling temporal dependencies in sequential data. For HAR, LSTMs are well-suited to learning motion patterns from time-series data, such as joint trajectories. Unlike traditional RNNs, LSTMs address the vanishing gradient problem, allowing them to retain long-term dependencies critical for understanding the dynamics of complex activities. [6]

2.3. Convolutional Neural Networks (CNNs)

Have also been applied to HAR, especially when the input data can be represented in a grid-like structure, such as images or spatiotemporal maps of joint movements. CNNs are particularly strong at capturing local spatial patterns, but they do not inherently model temporal dependencies or the topological relationships between joints in the human body. [12]

2.4. Spatio-Temporal Graph Convolutional Networks (ST-GCNs)

Unlike CNNs, which operate on fixed grid structures like images, ST-GCNs are designed to handle graph-structured data such as human skeletons. In ST-GCNs, joints are represented as nodes and bones as edges, allowing the model to capture both spatial relationships and temporal dynamics in motion sequences. This makes them particularly effective for tasks like human activity recognition, where movement patterns and joint connectivity matter. Compared to CNNs, ST-GCNs offer more flexibility in modeling the structure of the human body, but they are also more computationally demanding and require careful graph design. [10] [12]

3. Dataset

The dataset used in this study was collected using an OptiTrack motion capture system, which recorded full-body 3D skeletal data during a variety of human activities. A total of 12 recordings were made, primarily focused on squatting, sitting, and several squat variations, including pistol squats and squats with arm movements around the head and torso.

To prepare the data for machine learning, a custom preprocessing pipeline was developed to clean and restructure the raw motion capture output, as illustrated in Figure 3. Unnecessary rows and columns were removed, and non-essential body points such as arms, shoulders, and hands were excluded, since the focus of classification was limited to leg and chest-based movements. The coordinate system was re-centered by defining the chest as the new origin.

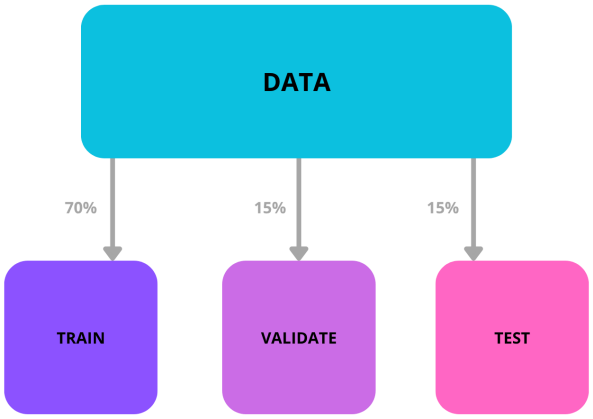

Figure 1: Distribution of the dataset [1]

The data was then normalized to ensure comparability across sessions and subjects. K-Means clustering was applied to assign preliminary labels, which were manually refined to produce final annotations. The resulting dataset was split into training (70%), validation (15%), and testing (15%) subsets. Summary visualizations of the dataset distribution and structure are provided in the Appendix. [4]

$$\text{Tensor}[N=1, C=3, T=5, V=4] =$$

$$\left[\begin{array}{c} \text{Channel X} \\ \begin{bmatrix} 0.0 & 1.0 & 2.0 & 1.0 \\ 0.1 & 1.1 & 2.1 & 1.1 \\ 0.2 & 1.2 & 2.2 & 1.2 \\ 0.3 & 1.3 & 2.3 & 1.3 \\ 0.4 & 1.4 & 2.4 & 1.4 \end{bmatrix}, \quad \text{Channel Y} \\ \begin{bmatrix} 0.0 & 2.0 & 0.0 & -1.0 \\ 0.2 & 2.1 & 0.1 & -1.0 \\ 0.4 & 2.2 & 0.2 & -0.9 \\ 0.6 & 2.3 & 0.3 & -0.8 \\ 0.8 & 2.4 & 0.4 & -0.7 \end{bmatrix}, \quad \text{Channel Z} \\ \begin{bmatrix} 0.0 & 1.0 & 1.0 & 0.0 \\ 0.0 & 1.1 & 1.0 & 0.0 \\ 0.0 & 1.1 & 1.0 & 0.0 \\ 0.0 & 1.2 & 1.0 & 0.0 \\ 0.0 & 1.2 & 1.0 & 0.0 \end{bmatrix} \end{array}\right]$$

Figure 2: Example of Reshaped Input Data

4. Methodology

We implemented and trained three distinct architectures: a Dual Support Vector Machine (DSVM), a Long Short-Term Memory (LSTM) network, and a Spatio-Temporal Graph Convolutional Network (ST-GCN).

4.1. DSVM Architecture

The DSVM model operates by decomposing the multi-class classification task into multiple binary SVMs. Given k clusters identified by KMeans, we construct $\frac{k(k-1)}{2}$ binary SVM classifiers, each trained on a unique pair of clusters. During inference, all classifiers make binary predictions, and the class receiving the highest number of votes is selected as the final prediction. Each individual SVM is trained using a standard fit procedure on its respective two-class subset. [8]

4.2. LSTM Architecture

The LSTM model was implemented using PyTorch and trained for 50 epochs. The architecture consists of two LSTM layers followed by fully connected layers and a softmax classifier. The first LSTM layer is bidirectional, allowing it to capture dependencies in both forward and backward temporal directions. The second LSTM layer refines the temporal encoding and passes the output to a series of dense layers. The model outputs a probability distribution across the cluster labels identified during preprocessing. [6]

4.3. ST-GCN Architecture

The ST-GCN model was adapted from the architecture proposed in Yan et al. (2018), originally designed for skeleton-based action recognition. The model takes a tensor representing spatial joints over time and applies a series of spatio-temporal graph convolutional layers. Each layer includes a graph convolution followed by a temporal convolution (TCN) and optional residual connections. The model incorporates edge importance weighting and dropout regularization. Final predictions are made via global pooling and a linear classifier. The complete architecture includes 9 graph-temporal convolutional layers with increasing feature dimensions and downsampling. The architecture was originally proposed to work on 2D image data. We had to adjust our data input to a different shape as shown in this Figure 2 to fit our architecture. [12]

5. Results

Table 1 summarizes the performance and resource metrics of the three evaluated models: Distributed SVM (DSVM), LSTM, and ST-GCN.

Metric Tests	DSVM	LSTM	ST-GCN
Training Time	1.05 s	535.66 s	1177.92 s
Accuracy: Squat	0.9883	0.9796	0.6035
Accuracy: Pistol Squat	0.9506	0.9886	0.5323
Accuracy: Sit	0.9567	0.9808	0.2356
Model Size	192 kB	1424 kB	10140 kB
Personal Evaluation	9.5/10	8.5/10	4.0/10

Table 1: Comparison of model performance, training time, and size

The DSVM model achieved the fastest training time (approximately 1 second) and demonstrated strong accuracy across all test sets, particularly in squatting activities. The LSTM model required substantially longer training time but achieved slightly better accuracy on pistol squats and sitting activities compared to DSVM. In contrast, the ST-GCN model had the longest training time and the largest model size, yet its accuracy lagged behind both DSVM and LSTM on all test tasks.

Our personal evaluation rates DSVM as the most favorable model due to its balance of high accuracy, efficient training time, and compact size, whereas the ST-GCN, despite its theoretical advantages, performed poorly in this context and was less practical due to its resource demands.

The confusion matrices for the three models (Figures 4, 5, and 6) illustrate their classification performance in more detail. The DSVM confusion matrix shows a strong diagonal representation, indicating that the model correctly predicted most of the samples across the different classes, confirming its high accuracy. Similarly, the LSTM confusion matrix demonstrates excellent classification performance with minimal misclassifications, further supporting its effectiveness.

In contrast, the STGCN confusion matrix reveals a highly imbalanced prediction pattern where the model predominantly predicted a single class for most samples. This behavior explains the substantially lower accuracy scores observed for STGCN, as the model failed to properly distinguish between different classes.

6. Conclusion

Overall, the experimental results largely confirmed our initial hypotheses. As expected, the ST-GCN model required a significantly longer training time due to its complexity. However, one surprising outcome was the performance of the DSVM model, which not only trained extremely quickly but also achieved remarkably high accuracy across all tested activities. This highlights the potential of combining traditional machine learning with well-structured feature engineering, especially when computational efficiency is critical.

Contrary to expectations, the ST-GCN model did not perform well. Instead of correctly classifying multiple classes, it predominantly predicted a single label, leading to low classification performance. While this result did not support our hypothesis, it does not necessarily indicate a flaw in the architecture itself. ST-GCNs are primarily designed for spatiotemporal graph data, and they may be better suited for tasks like action recognition in videos rather than sparse 3D joint-based data used in this study.

As for the discussion, a key consideration is how these models scale with an increasing number of activity classes. While the DSVM showed excellent accuracy with the current class set, its structure would require training numerous SVMs as the number of classes increases, which may become impractical. In such scenarios, sequence-based models like LSTM might offer better scalability. On the other hand, ST-GCN may be more effective for high-dimensional visual inputs (e.g., RGB video) where spatial relationships are more complex and informative.

Lastly, an essential component influencing the models' performance was the K-Means algorithm used in the preprocessing phase. Since K-Means was responsible for clustering and labeling motion segments, the quality of these clusters significantly impacted the models' prediction accuracy. More insights on this dependency can be found in the accompanying code in the file `code/notes.txt`. [7]

References

[1] URL: `http://www.canva.com/`.

[2] URL: `https://www.shutterstock.com/search/annotation-icon`.

[3] *Flowchart Maker & Online Diagram Software*. URL: http://www.draw.io/.

[4] Salvatore Gaglio, Giuseppe Lo Re, and Marco Morana. "Human Activity Recognition Process Using 3-D Posture Data". In: *IEEE Transactions on Human-Machine Systems* 45.5 (2015), pp. 586–597. DOI: 10.1109/THMS.2014.2377111.

[5] *Gumball — Rhino 3-D modeling*. URL: https://docs.mcneel.com/rhino/8mac/help/en-us/commands/gumball.htm.

[6] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9 (Nov. 1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

[7] Jaroslav Kvasnička. *Code Repository*. URL: https://github.com/xkvasn14/Seminar-Practical-Work-AI.

[8] Alessandro Manzi, Paolo Dario, and Filippo Cavallo. "A Human Activity Recognition System Based on Dynamic Clustering of Skeleton Data". In: *Sensors* 17.5 (2017). ISSN: 1424-8220. DOI: 10.3390/s17051100. URL: https://www.mdpi.com/1424-8220/17/5/1100.

[9] Alex O'Neill. *Implementing the K-means algorithm in Python*. Sept. 2022. URL: https://www.linkedin.com/pulse/implementing-k-means-algorithm-python-alex-o-neill/.

[10] Yi-Fan Song et al. "Stronger, Faster and More Explainable: A Graph Convolutional Baseline for Skeleton-based Action Recognition". In: *Proceedings of the 28th ACM International Conference on Multimedia*. MM '20. Seattle, WA, USA: Association for Computing Machinery, 2020, pp. 1625–1633. ISBN: 9781450379885. DOI: 10.1145/3394171.3413802. URL: https://doi.org/10.1145/3394171.3413802.

[11] Estuary Twitter. *Data Normalization explained: Types, examples, & methods*. Jan. 2025. URL: https://estuary.dev/blog/data-normalization/.

[12] Sijie Yan, Yuanjun Xiong, and Dahua Lin. "Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1 (Apr. 2018). DOI: 10.1609/aaai.v32i1.12328. URL: https://ojs.aaai.org/index.php/AAAI/article/view/12328.

A. Appendix

A.1. Pictures in Figures References

- Preprocessing Pipeline - Gumball picture [5]
- Preprocessing Pipeline - Normalization picture [11]
- Preprocessing Pipeline - Kmeans [9]
- Preprocessing Pipeline - Annotation [2]
- Preprocessing Pipeline - Data Parsing [1]
- Preprocessing Pipeline - Removing points [3]

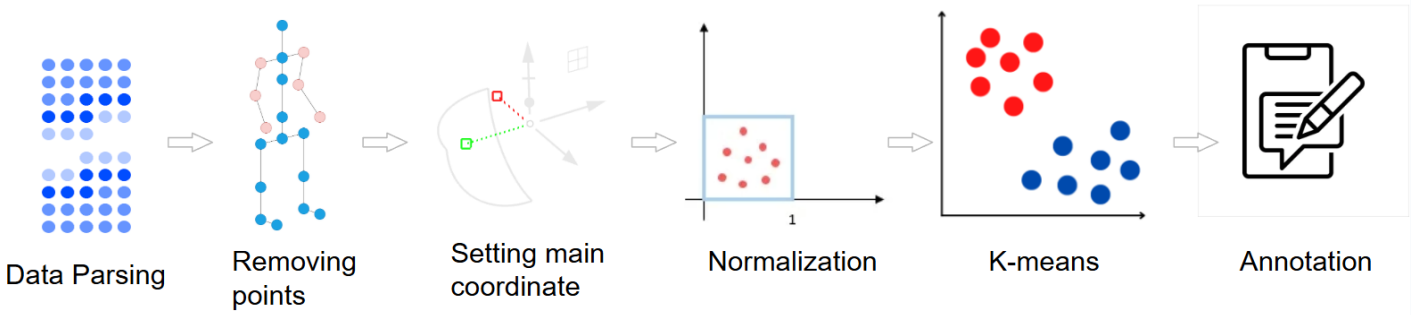

Figure 3: Preprocessing Pipeline [4]

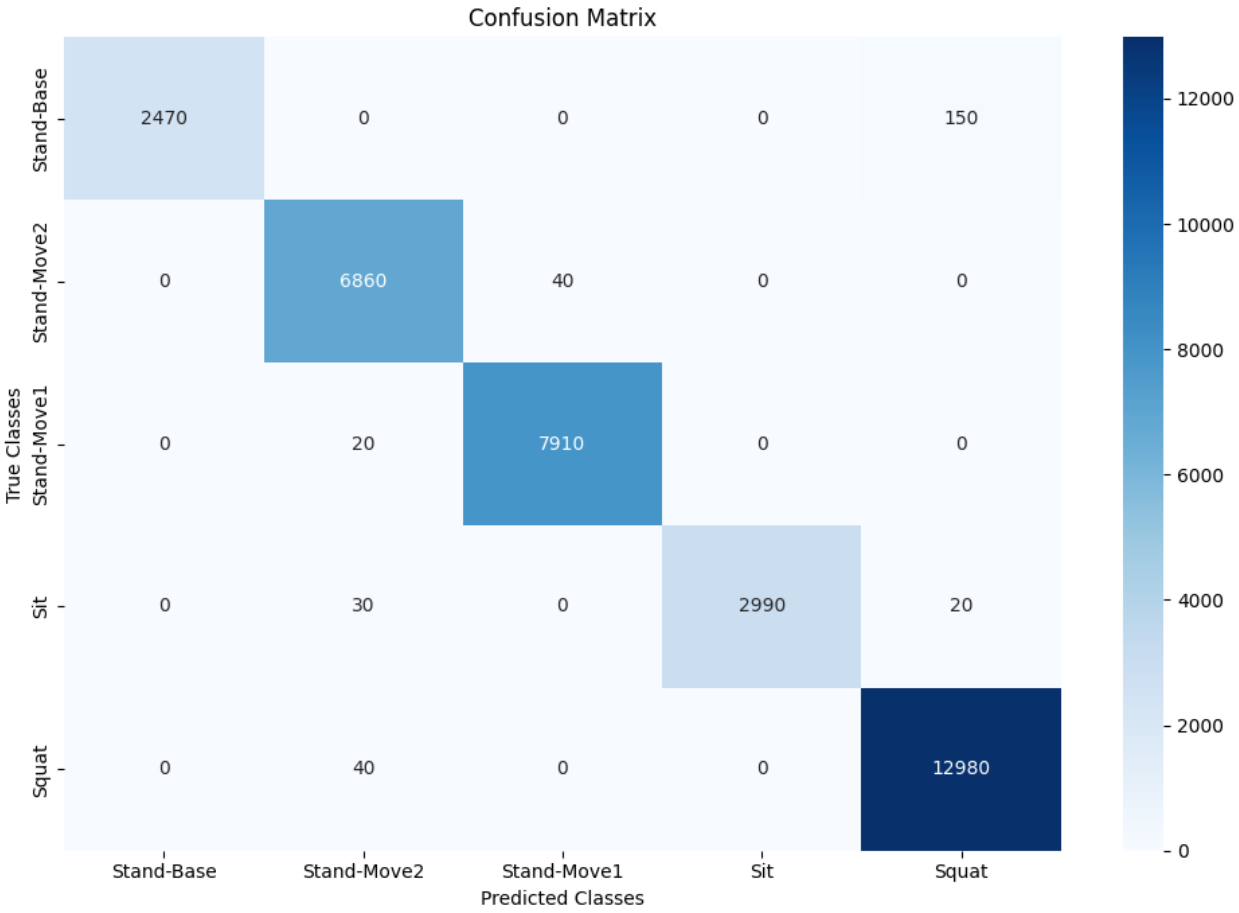

Figure 4: Confusion Matrix DSVM

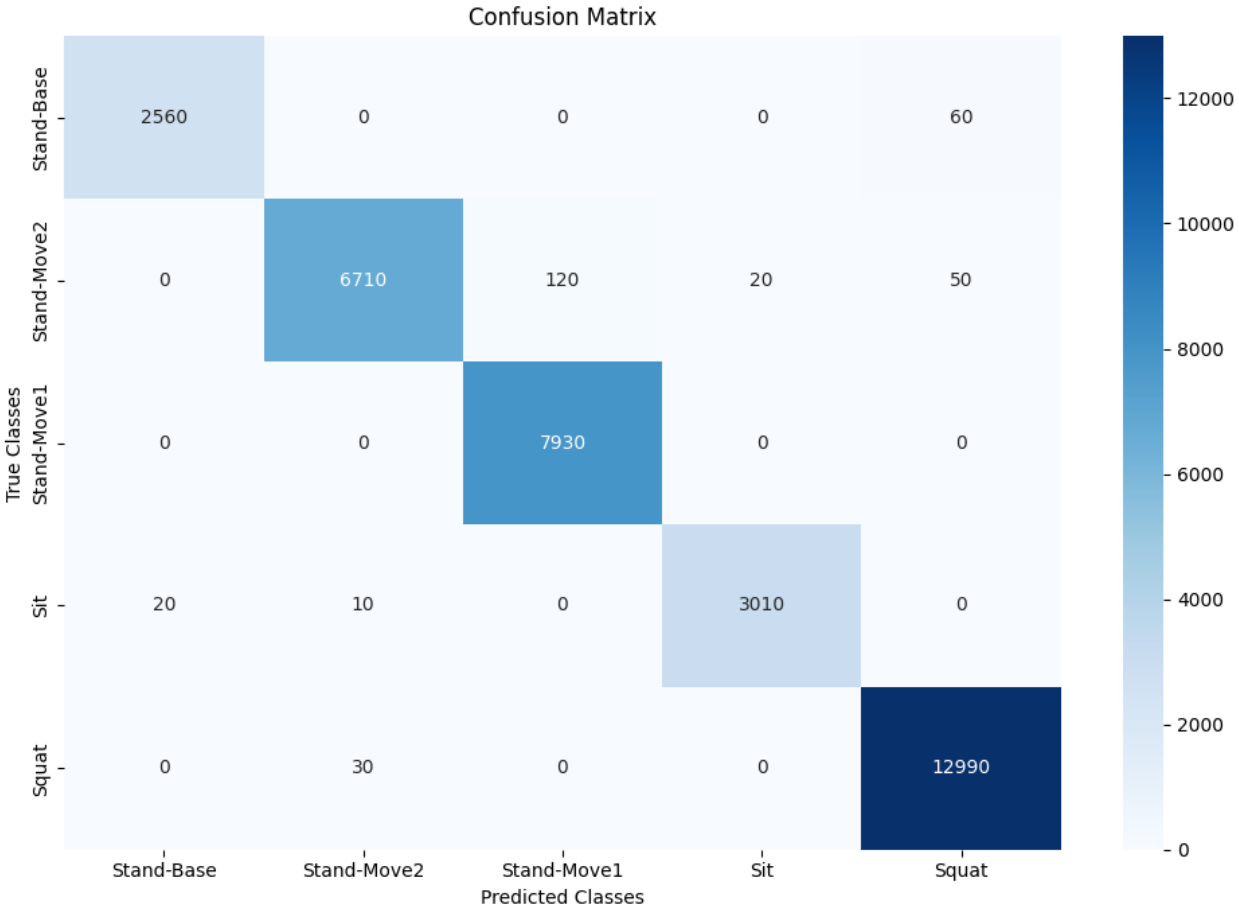

Figure 5: Confusion Matrix LSTM

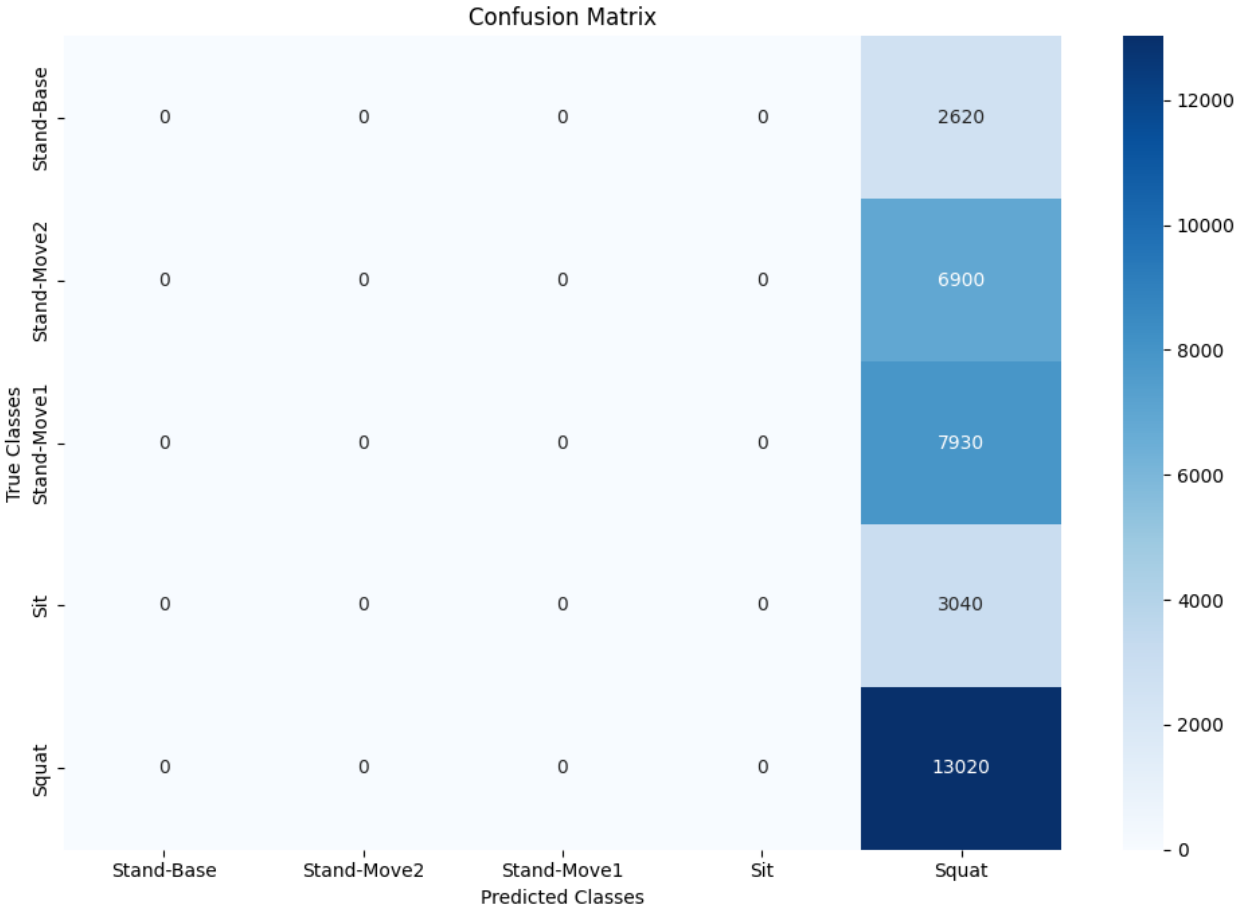

Figure 6: Confusion Matrix STGCN

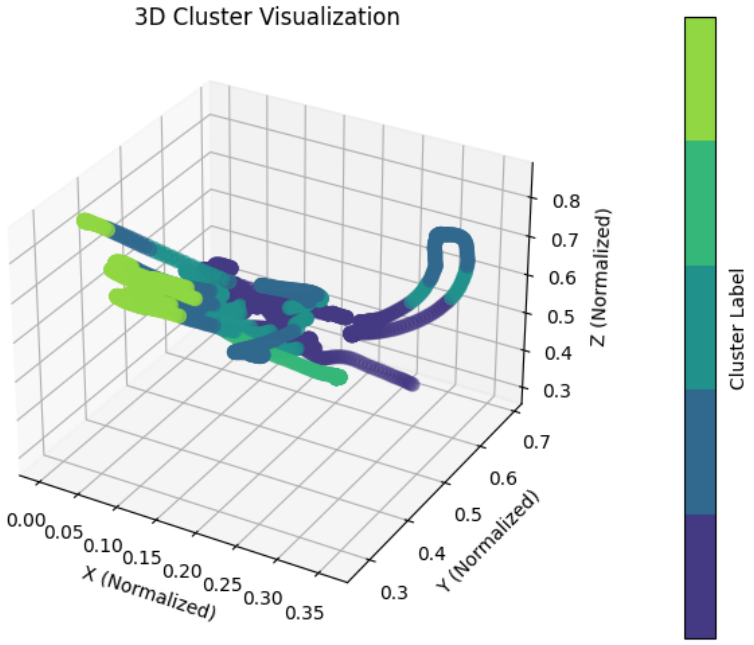

Figure 7: Dataset

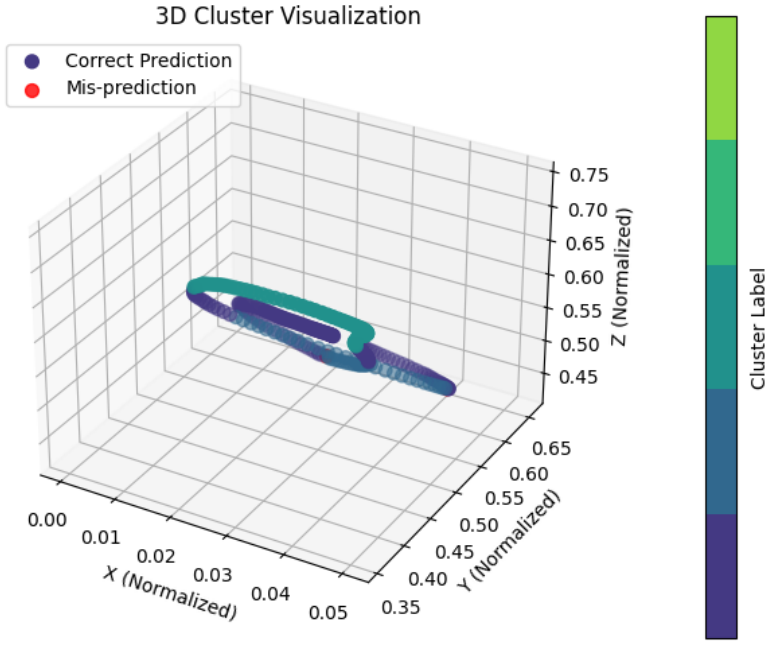

Figure 8: Squat - DSVM

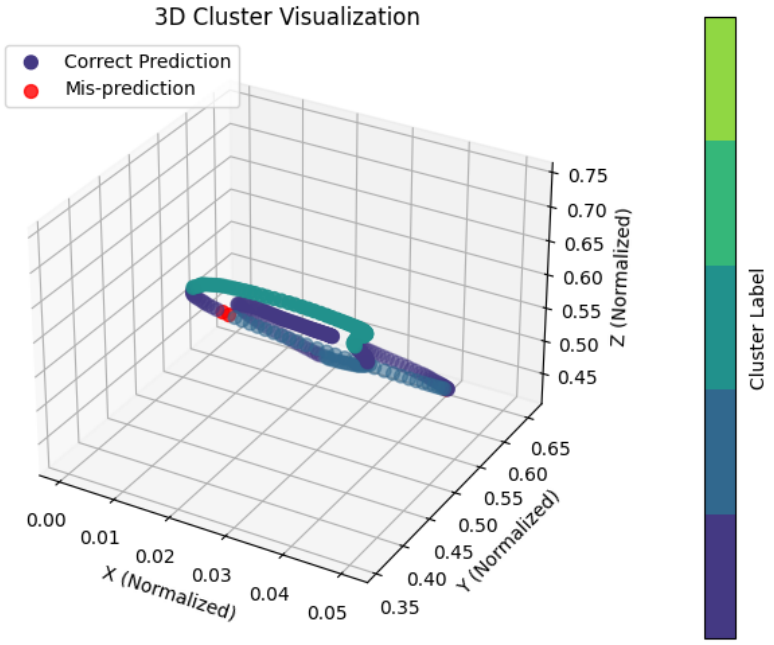

Figure 9: Squat - LSTM

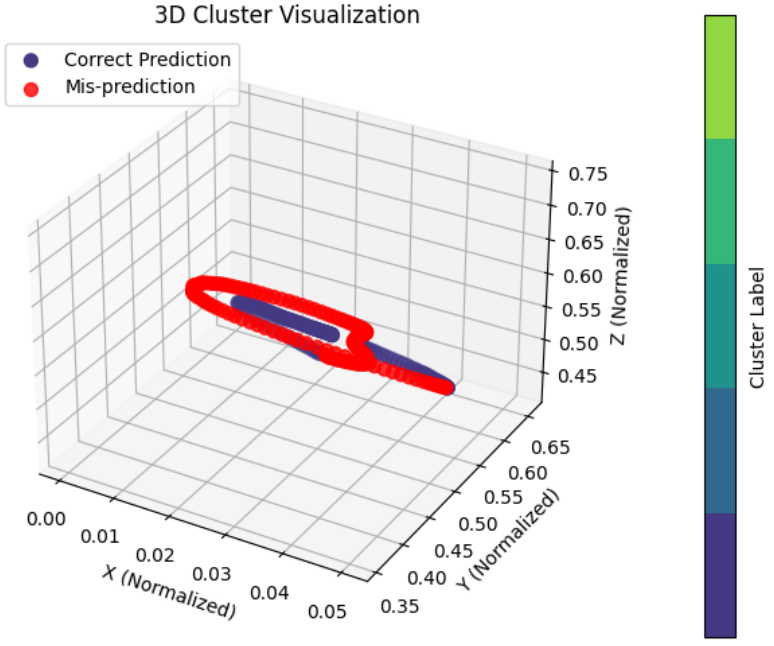

Figure 10: Squat - STGCN

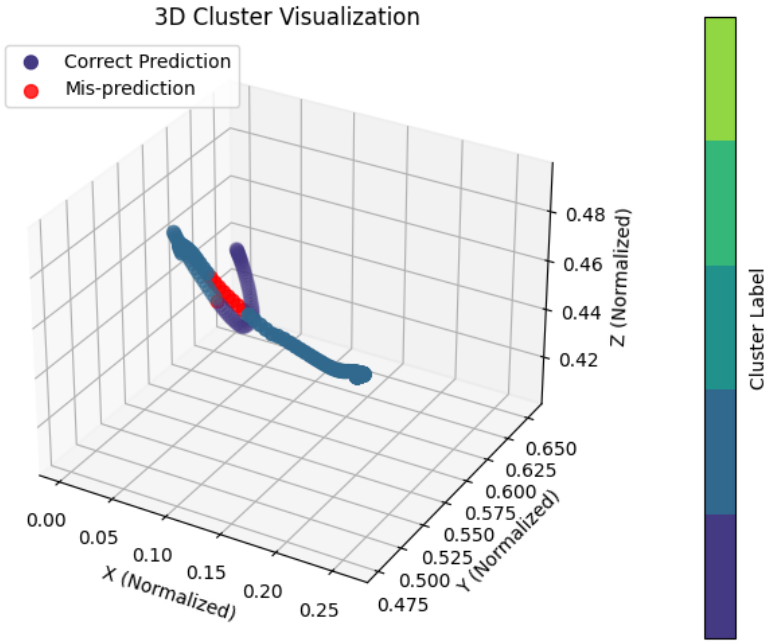

Figure 11: Sit - DSVM

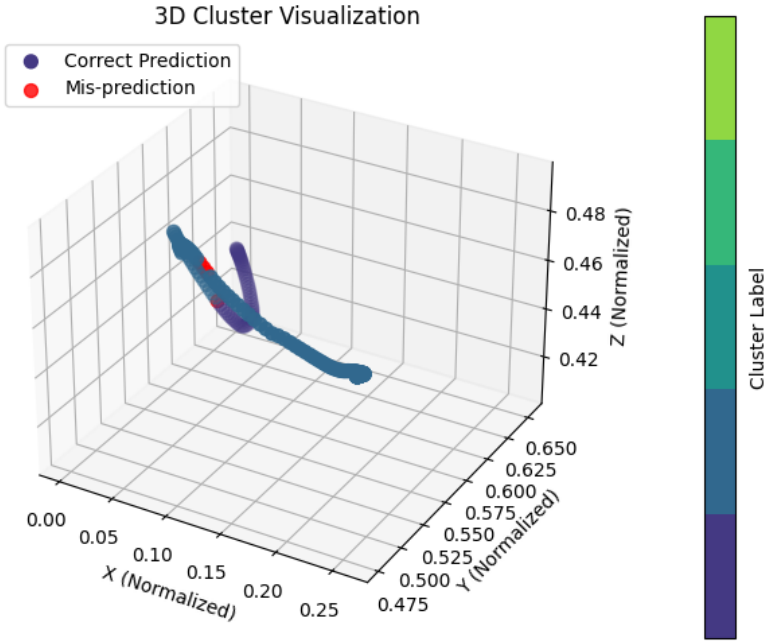

Figure 12: Sit - LSTM

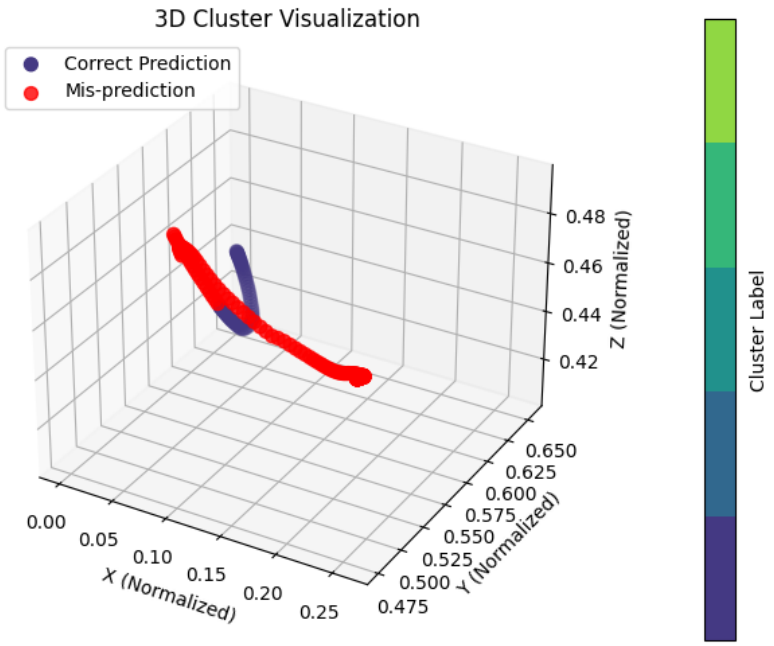

Figure 13: Sit - STGCN