**Gesture-Based Communication**

**Between Workers and Robots**

Prepared for

Dr. Mohammadreza Ebrahimi

University of South Florida, Muma College of Business

By

Bhargav Rishi Medisetti

Chi Phuong Diep

Devansh Pavdighada

May 8, 2025

Table of Contents

[Introduction 3](#_Toc197613378)

[Background and Motivation 3](#_Toc197613379)

[Research Question and Approaches 3](#_Toc197613380)

[Dataset 4](#_Toc197613381)

[Dataset Overview 4](#_Toc197613382)

[Preprocessing and Feature Engineering 4](#_Toc197613383)

[Model Justifications 5](#_Toc197613384)

[Experimental Results 6](#_Toc197613385)

[Accuracy and Confusion Matrix 6](#_Toc197613386)

[Top Model Performance 9](#_Toc197613387)

[Model Performance Comparison 9](#_Toc197613388)

[Business Implications & Recommendations 10](#_Toc197613389)

[Business Values 10](#_Toc197613390)

[Recommendations 10](#_Toc197613391)

[References 10](#_Toc197613392)

# Introduction

## Background and Motivation

The integration of robots into workspaces is becoming increasingly common across a wide range of sectors, including healthcare, retail, manufacturing, and hospitality. According to a case study we found, Amazon alone has deployed over 750,000 mobile robots and tens of thousands of robotic arms in its fulfillment centers. This implementation has contributed to a 25 percent reduction in order fulfillment costs and is projected to save the company $10 billion annually by 2030 (Williams, 2024). While this shift reflects promising advancements in automation, many robots still rely on voice commands or touchscreen interfaces for communication. These methods of interaction are not always ideal or effective, especially in fast-paced or noisy environments where workers may not be able to speak clearly or be close enough to physically interact with a device.

Our project was inspired by Kroger’s shelf-scanning robot known as Barney which assists inventory management. Although effective in its current role, Barney depends primarily on voice recognition and touchscreen inputs. This left us wondering whether robots could understand human gestures instead. In busy environments, gestures are often quicker and more intuitive which makes them a convincing alternative for human robot communication.

Beyond improving interaction, gesture recognition also offers cost-saving benefits for companies. Research has shown that integrating gesture-based control can reduce cycle times by up to 20 percent which leads to more efficient workflows and lower operational delays (Feldhorst, 2021). Additionally, this enhancement is more cost effective to implement than traditional interfaces because it relies on computer vision rather than hardware or complex tactile systems (Kokic, 2023). This can reduce both initial investments and ongoing maintenance costs. These systems may also help lower human error, enhance workplace safety, and support automation in environments where other interaction methods may fall short. With the global gesture recognition market projected to grow from 31.8 billion dollars in 2025 to over 161 billion dollars by 2032, the increasing adoption of this technology reflects both its practical value and strong return on investment (Fortune Business Insights, 2024).

Given these technical advantages and the clear business relevance, we believe our project is strongly motivated. Our team wanted to take this a step further by exploring whether deep learning models could be used to recognize and interpret human gestures from skeleton-based motion data.

## Research Question and Approaches

We wondered what it would take for a robot to recognize simple but meaningful actions such as a wave to get attention, a nod to confirm or a hand signal to pause. These gestures can be easily interpreted by humans, but teaching a machine to understand them is far from simple. To achieve that, we wanted to explore these research questions:

* Can we teach robots to understand human gestures using deep learning techniques?
* Which gestures are most accurately recognized through this approach?

To answer this question, we drew out the objectives for our project. First, we analyzed NTU RGB+D Action Recognition Dataset and narrowed our focus on movements that could realistically be used in workplace settings. Second, we collaborated as a team to design and train three deep learning models to classify these gestures and compare their performance across different architectures. We kept the limitations in the actual world in mind and since we used Google Colab as our development environment, we had to carefully manage memory and compute usage. Finally, we evaluated how well these models understood human intent through metrics like classification accuracy and confusion matrices which gave us insight into both the strengths and limitations of each approach.

# Dataset

## Dataset Overview

As we searched for a dataset that was not only extensive but also structurally rich, we came across the NTU RGB+D dataset (<https://rose1.ntu.edu.sg/dataset/actionRecognition/>) which captured the complexity of real human movements. This dataset is ideal for our project as we try to develop a gesture recognition model where robots can recognize human actions. The NTU RGB+D dataset contains 56,880 video samples covering 60 different action classes performed by 40 subjects across 80 camera viewpoints. We decided to focus on the skeleton modality which captures the 3D (x,y,z) coordinates of 25 human joints per frame. We were especially interested in using skeleton-based data rather than relying on raw video or audio. This method not only preserves user privacy but also makes the system more powerful in busy or noisy environments.

The 60 action classes were designed to reflect human behaviors in collaborative or assistive environments. They are divided into three major groups:

* 40 daily activities such as drinking, eating, reading, brushing teeth, clapping
* 9 health-related actions such as sneezing, staggering, and falling
* 11 interactive actions such as handshaking, patting on the back, hugging, and giving objects

## Preprocessing and Feature Engineering

To prepare the data for modeling, we implemented a custom preprocessing function that reads raw .skeleton files and extracts joint positions for each person in every frame. The function processes each file line by line, extracting the total number of frames and bodies, followed by the number of joints per frame. It then stores the 3D joint positions for each person in a structured dictionary format using keys such as skel\_body0, skel\_body1, etc. Each entry is a NumPy array of shape (frames, joints, 3), representing how a person’s joints move over time. To handle known issues within the dataset, we incorporated a predefined list of missing or corrupted samples, skipping these files during conversion. These processed dictionaries, along with the action label derived from the filename (e.g., A001 corresponds to label 0), were saved as individual .npy files for efficient loading during model training.

Following the initial conversion, distinct preprocessing pipelines are implemented within the respective PyTorch Dataset classes for each model. Feature engineering distinguishes the models: the GCN-based architectures (2S-STGCN and 2S-TCN) compute an additional "bone stream" feature, derived from the vector differences between anatomically connected joints, to represent relative motion. The Transformer model, however, exclusively utilizes the joint coordinate information. Finally, the data is formatted for model consumption; the GCN models transpose the joint and bone streams into a (Channel, Time, Vertex) tensor structure, whereas the Transformer dataset standardizes sequences to 50 frames and reshapes the joint data into a (Time, Features) sequence format compatible with its encoder input requirements. To evaluate performance fairly, we split the dataset into 80% training and 20% validation sets using a random split. This allowed us to train our model while holding out a portion of the data to monitor generalization. We used 45,537 samples for training and 11,343 for validation, ensuring the model could be tested on unseen data to measure its ability to recognize gestures it had not encountered during training.

To support more complex training workloads, we also invested in Colab Pro, which gave us access to extended GPU sessions and helped reduce training time.

# Model Justifications

Based on our objective of classifying human gestures, we selected models that can capture how joints move over time and how they relate to one another in space. To evaluate different modeling strategies, we chose three models with a specific goal in mind: 2S-TCN for strong temporal modeling, Transformer for global sequence attention, and 2S-STGCN for full spatiotemporal understanding.

1. *Two-stream Temporal Convolutional Network (2S-TCN)*

We started with 2S-TCN as our baseline model because it is lightweight, easy to train, and performs well on sequential tasks which makes it ideal for initial experimentation. It uses temporal convolution layers to learn how joints move over time, which made it a natural starting point for gesture recognition. We implemented a two-stream version that includes both joint positions and bone vectors, allowing the model to learn both absolute motion and relative movement between joints. While it does not capture the full complexity of spatial relationships in the skeleton, it provided valuable insights into the importance of motion patterns and demonstrated that basic gesture recognition is still achievable.

1. *Transformer Encoder*

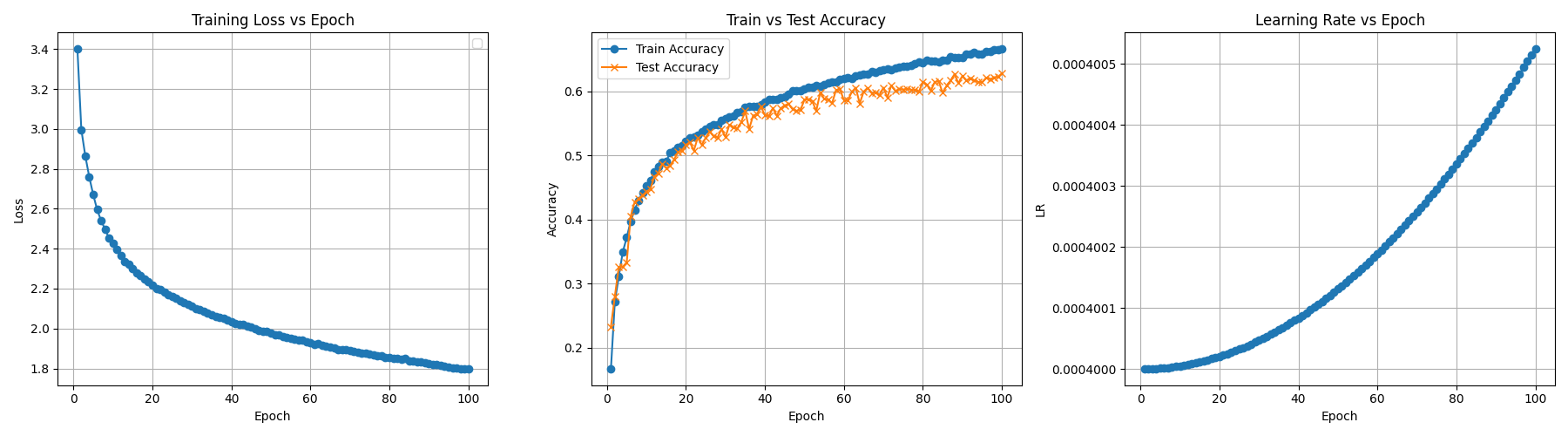
We selected the Transformer encoder model because of its strong reputation in natural language processing and time series analysis. Transformers use attention mechanisms to model long-term dependencies in sequences, which we believe could help detect subtle gestures that unfold slowly or across distant frames. Before building, we expected the Transformer to offer flexibility and high learning capacity. However, we also knew it did not include spatial structure, which we suspected might limit its ability to fully understand body movement.

1. *Two-stream Spatial-Temporal Graph Convolutional Network (2S-STGCN)*

We chose 2S-STGCN because it directly aligns with the structure of our skeleton data. It uses spatial graph convolution to model joint relationships based on a predefined skeleton structure, and temporal convolution to track how these relationships evolve over time. We also included a second stream for bones to provide additional motion cues. Before building, we believed this model would be the most powerful because it was specifically designed for human action recognition using skeleton graphs. It captures both spatial and temporal patterns which we thought were essential for accurate gesture classification.

# Experimental Results

## Accuracy and Confusion Matrix



*Figure 1. Accuracy Results of 2S-TCN*

Explanation of Figure 1: Accuracy Results of 2S-TCN

After 100 epochs, the 2STCN model reached the final test accuracy of approximately 63% which makes it a strong baseline model for comparison. On the left-hand side, the graph illustrates a steady decrease in training loss meaning the model is learning well. The middle graph shows how closely the accuracy curves between train and test aligned which suggests minimal overfitting and good generalization to unseen data. On the right-hand side, the OneCycle learning rate we used for this model gradually increased to promote better convergence during training. There are some misclassifications between similar gestures such as hugging a person or handshaking which is likely due to overlapping joint movements and the model’s limited ability to detect spatial relationships between joints.

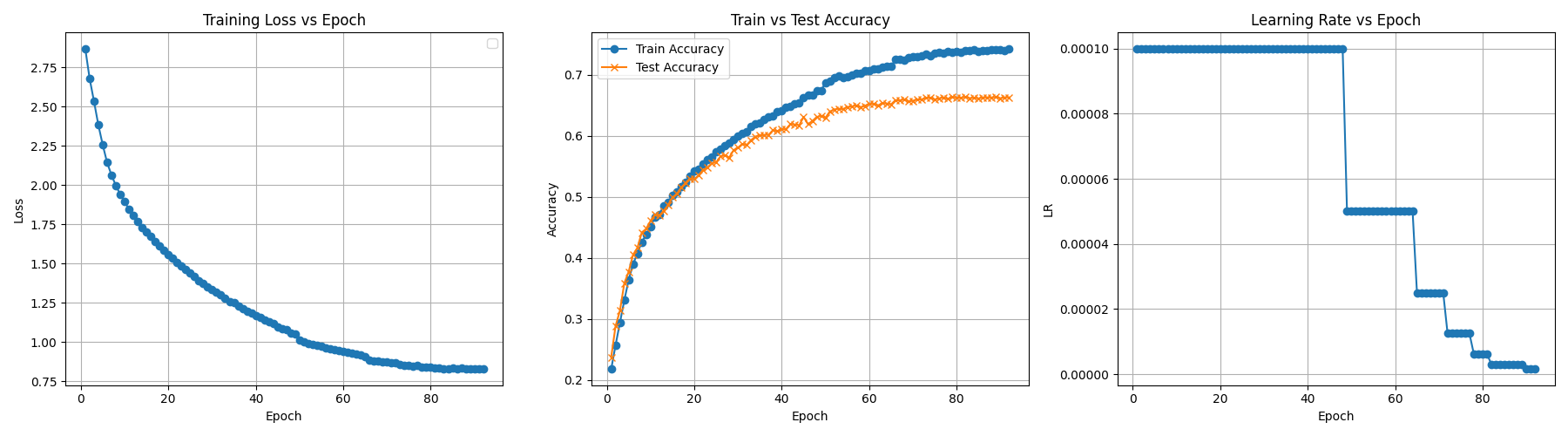
*A graph with a line in the middle

AI-generated content may be incorrect.*

*Figure 2. Confusion Matrix of 2S-TCN*

Explanation of Figure 2: Confusion Matrix of 2S-TCN

The confusion matrix for the 2STCN model shows strong overall performance. However, the model struggles with gestures that involve similar motion patterns such as hugging person vs handshake and put on hat vs take off hat. For classes which have accuracy < 50%, they have been highlighted with a red border in the matrix. These misclassifications suggest that while 2STCN effectively captures temporal dynamics, it lacks the spatial relationships needed to distinguish between actions with overlapping joint movements. This highlights a key limitation of this model’s architecture which leads us to explore other models.

**

*Figure 3. Accuracy Results of Transformer Encoder*

Explanation of Figure 3: Accuracy Results of Transformer Encoder

After 90 epochs, the model achieved a final test accuracy of approximately 65% which shows a solid learning and improved generalization compared to the baseline. On the left, the training loss consistently decreases from around 2.75 to just under 0.9 which indicates that the model was learning effectively throughout training. The middle graph shows a strong alignment between the training and test accuracy curves which suggests that overfitting was minimal. Although accuracy plateaued in the later epochs, the performance remained stable, and the gap between train and test curves stayed relatively small. On the right, the learning rate plot reveals a step decay schedule where the learning rate is reduced in stages. This approach helped the model fine tune its performance as training progressed.

*A graph with a line in the middle

AI-generated content may be incorrect.*

*Figure 4. Confusion Matrix of Transformer Encoder*

Explanation of Figure 4: Confusion Matrix of Transformer Encoder

The confusion matrix for the Transformer encoder shows that the model performed fairly well overall. However, there were still several noticeable misclassifications. For example, it had trouble distinguishing between gestures like wearing glasses and take off glasses which likely look similar without a clear sense of direction. These kinds of mistakes suggest that while the Transformer does a good job capturing how joints move over time, it struggles to understand spatial structure between joints. From this, we thought adding spatial information or structure could help the model make better predictions.

A graph with blue and orange lines

AI-generated content may be incorrect.

*Figure 5. Accuracy Results of 2S-STGCN*

Explanation of Figure 5. Accuracy Results of 2S-STGCN

After 40 epochs, the model achieved a final test accuracy of approximately 78% which is the highest accuracy in our experiments. On the left, the training loss steadily decreased until around epoch 30, then slightly increased, suggesting early signs of overfitting as the model learns the training data too closely. The middle graph shows how the training accuracy continues to climb while the test accuracy plateaus and then slightly drops after epoch 35. On the right, the learning rate follows a cosine annealing schedule which starts high to explore broadly, then gradually decreases to help the model converge. Toward the end, it slightly increases again. Despite some overfitting, the model maintained strong generalization and delivered the highest overall performance on the gesture recognition task.

A graph with a blue line

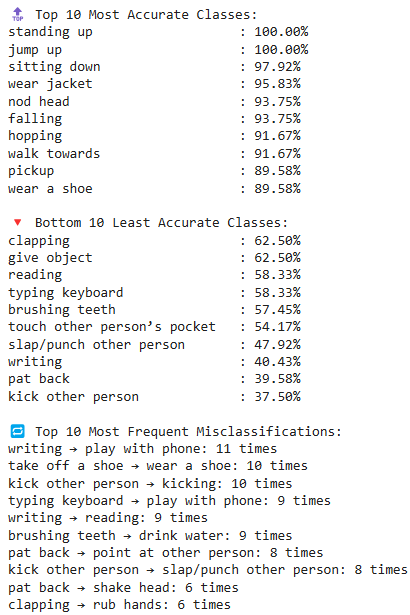
AI-generated content may be incorrect.

*Figure 6. Confusion Matrix of 2S-STGCN*

*Explanation of Figure 6. Confusion Matrix of 2S-STGCN*

The confusion matrix for the 2S-STGCN model shows strong performance as indicated by high accuracy. Compared to the other models, this one made fewer major mistakes, especially on gestures that involve full-body movement. For example, there is an occasional mix-up between reading and writing. These are common across models because the motions are very similar and can look nearly identical in skeleton form. That said, 2S-STGCN handled these cases better than the others, likely because it combines both spatial and temporal information. This gives us more confidence that this model is the most effective at capturing complex motion patterns.

## Top Model Performance



*Figure 7. Accuracies by Label of 2S-STGCN*

Explanation of Figure 7. Accuracies by Label of 2S-STGCN

The figure shows per-class accuracy breakdown for 2S-STGCN. While it performs well on gestures involving clear and distinct body movements, such as standing up or jumping. In contrast, the model struggles with more subtle gestures that share similar movement patterns, particularly those involving hand or arm actions like writing or clapping. This suggests that while 2S-STGCN effectively captures spatiotemporal features, it has a difficult time distinguishing between gestures with overlapping motions.

## Model Performance Comparison

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Type** | **Accuracy** | **Pros** | **Cons** |
| 2S-TCN | Temporal CNN  (Joint + Bones) | ~ 63% | Simple, captures temporal patterns well | No spatial modeling, complex to tune |
| Transformer | Encoder Only | ~ 65% | Handles long sequences, flexible architecture | Lacks spatial bias |
| 2S-STGCN | Spatial-Temporal GCN  (Joint + Bones) | ~ 77.8% | Best performance, models spatial and temporal features | Sensitive to overfitting, higher complexity |

# Business Implications & Recommendations

## Business Values

Based on our analysis, we found that deep learning models can accurately recognize human gestures, which makes gesture recognition a strong alternative to traditional ways of interacting with robots. This is important because in busy or noisy workplaces, gesture recognition will be more reliable and efficient because it allows workers to communicate with robots in a faster and more natural way.

For business stakeholders, this offers both practical and financial advantages. Gesture recognition systems use computer vision instead of expensive hardware like sensors which means they are more affordable to set up and easier to maintain. This can help reduce costs over time. Additionally, making robots more responsive to human gestures improves workflow efficiency, reduces errors, and can help prevent workplace accidents. As companies like Amazon and Kroger continue investing in automation, this could help increase productivity, improve worker safety, and make robotic systems more adaptable to different challenges.

## Recommendations

We believe gesture recognition can offer real benefits in workplace settings when implemented thoughtfully. Here are some recommendations for moving forward:

1. *Train on video data if possible*

If resources and capacity are allowed, we recommend training on full video data in addition to skeleton sequences. Video inputs can capture more subtle cues like hand shape, posture, and motion context that may not be fully represented in skeletal data.

1. *Explore more advanced models*

We can use AGCN instead of STGCN to learn optimal spatial connections between joints, rather than relying on a fixed skeleton graph. Recent research has also introduced more advanced approaches such as CTR-GCN, SGN or MSG3D. These models are specifically designed to better capture both spatial and temporal dynamics of skeletal data. If resources and time allow, we recommend exploring these models in future projects to further improve accuracy.

1. *Customize for each workplace setting*

Every workplace is different, so companies should plan to retrain or fine-tune models to fit their own environment and common gestures. This will help maintain accuracy and ensure the system continues to work well overtime.

1. *Apply reinforcement learning*

While our models were trained using supervised learning, reinforcement learning will come in handy as robots continue to learn and improve from interaction feedback. This would be very useful in unpredictable settings where gestures vary and change overtime.

# References

* Feldhorst, S., Dohrmann, L., & Toth, D. (2021). Pose estimation and gesture recognition for human-robot collaboration in assembly. Frontiers in Robotics and AI. Retrieved from  
  <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8649894>
* Fortune Business Insights. (2024). Gesture Recognition Market Size, Share & COVID-19 Impact Analysis. Retrieved from  
  <https://www.fortunebusinessinsights.com/industry-reports/gesture-recognition-market-100235>
* Kokic, M., Novák, S., & Nemecek, P. (2023). Hand gesture communication in human-robot cooperation using computer vision. ECMS 2023. Retrieved from  
  <https://www.scs-europe.net/dlib/2023/ecms2023acceptedpapers/0373_ltma_ecms2023_0043.p>
* Shahroudy, A., Liu, J., Ng, T. T., & Wang, G. (2016). NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1010–1019.GitHub: <https://github.com/shahroudy/NTURGB-D>
* Williams, A. (April 24, 2024). Amazon’s $10 billion robot revolution aims to cut fulfillment costs. Financial Times. <https://www.ft.com/content/31ec6a78-97cf-47a2-b229-d63c44b81073>