Deep Learning: Final Project Report

Aryaman Khandelwal, Hriday Rathi, Shreevardhan Shah,
Tushar Garg, Tushar Goyal

May 10 2024

Contents

1 Introduction

Dynamic hand gesture recognition is at the forefront of advancing human-computer interaction technologies. By capturing and interpreting the complex movements of the hand over time, this field holds the potential to revolutionize the way users engage with machines, offering a more natural and responsive interface.

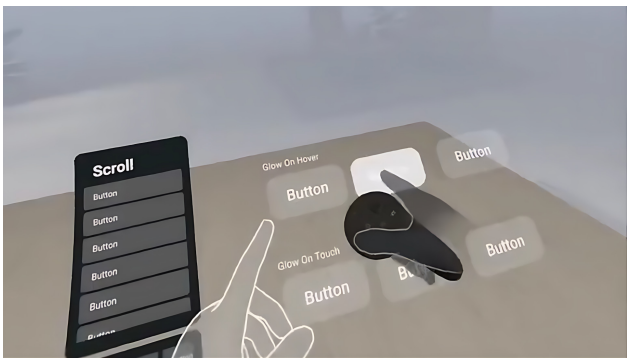

Figure 1: Hand Tracking in Meta Quest

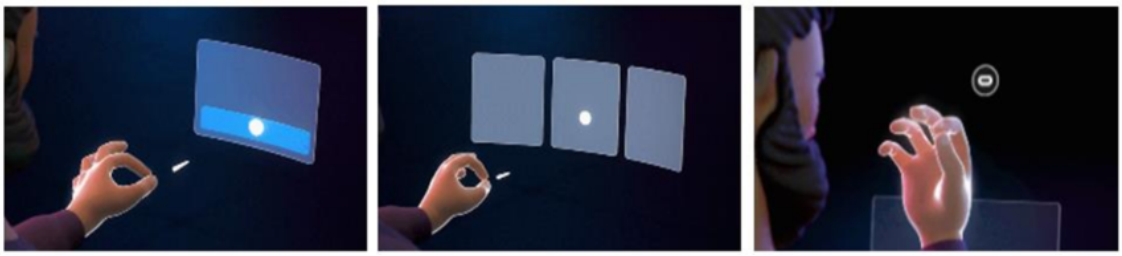

Figure 2: Available Gestures in Meta Quest
(a) Point and Pinch; (b) Pinch and Scroll; (c) Palm Pinch

Our project directly addresses the technical challenges involved in accurately classifying these gestures, which is crucial for the practical application of gesture-based systems in real-world scenarios.

2 Current Approaches to Dynamic Hand Gesture Detection

2.1 Skeleton-Based Recognition:

- **Method:** This method involves tracking the skeletal structure of the hand, identifying key points like knuckles and fingertips, and analyzing their movements over time.
- **Tools:** Technologies such as depth cameras and software frameworks like **MediaPipe** are commonly used to capture the skeletal data.
- **Advantages:** High accuracy in gesture detection due to the detailed structural data.
- **Limitations:** Often requires specific hardware and can struggle with occlusions where the camera view is blocked or the hand orientation changes drastically.

2.2 Optical Flow and Motion Capture:

- **Method:** Optical flow techniques track the motion between frames of video to infer the direction and velocity of moving objects (i.e., hands).
- **Advantages:** Useful in capturing the dynamic aspects of gestures, providing real-time tracking capabilities.
- **Limitations:** Highly sensitive to lighting conditions and background movement, which can lead to inaccuracies in gesture recognition.

2.3 Machine Learning and Deep Learning Models:

- **Method:** Models such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), including Long Short-Term Memory networks (LSTMs), are employed to recognize patterns in gesture sequences.
- **Advantages:** Can learn complex gesture patterns and improve over time as more data is gathered.
- **Limitations:** Requires extensive labeled datasets for training, and performance can degrade if the gestures deviate from the training data.

3 Gaps and Limitations in Current Methods

- **Generalization:** Many existing systems are trained on specific datasets under controlled conditions and fail to perform well in diverse, real-world environments where background conditions, lighting, and user variations are more complex.
- **Real-Time Processing:** There is often a trade-off between accuracy and speed. High-accuracy systems may not operate in real-time, which is crucial for applications such as virtual reality or interactive gaming.
- **Complex Gesture Recognition:** Recognizing gestures involving multiple hands or interacting with objects remains challenging due to the increased complexity and potential occlusions.
- **Adaptability:** Systems generally lack adaptability to new gestures or variations of known gestures without retraining, which is not feasible in dynamic settings.
- **Dependency on Hardware:** High performance often depends on specialized hardware, which can limit the deployment of gesture recognition technologies in mobile or low-resource settings.

4 Related Work

4.1 Double-feature Double-motion Network (DD-Net)

- **Introduction:** Skeleton-based action recognition is crucial for applications ranging from multimedia to assistive technologies, but **traditional methods** have often been hampered by **large model sizes** and **slow processing speeds**. DD-Net addresses these issues through a streamlined architecture that leverages unique feature sets tailored for dynamic action recognition.

- **Proposal:** DD-Net distinguishes itself by proposing a **lightweight** and **fast-performing** model that overcomes the limitations of previous skeleton-based recognition systems. Traditional approaches generally suffer from the trade-off between accuracy and performance, particularly in real-time applications. DD-Net introduces two key innovations:

  - **Joint Collection Distances (JCD):** This feature captures spatial relationships between skeleton joints in a way that is invariant to changes in location and viewpoint, which enhances the model's robustness and applicability in diverse scenarios.
  - **Two-scale Global Motion Feature:** This feature incorporates both slow and fast motion scales, allowing the model to accurately capture a wide range of human activities by acknowledging the varying dynamics in human movement.

  Both the above features, ensure that DD-Net not only remains compact but also retains high accuracy and speed, making it suitable for real-time applications.

- **Architecture:** DD-Net employs a 1D CNN architecture tailored for processing sequential data, which is particularly effective for extracting temporal dynamics from skeletal movements. The architecture is optimized to handle the sequence and timing of joint movements efficiently, which are critical for action recognition. By using 1D convolutions, DD-Net focuses on temporal feature extraction across the sequence of frames, thereby reducing computational complexity compared to 2D CNNs used for spatial data.

  The network utilizes two primary features: the Joint Collection Distances (JCD) and a two-scale global motion feature. JCD captures **spatial relationships** between joints, providing a robust, viewpoint-invariant measure that's crucial for consistent action recognition across different scenarios. The two-scale global motion feature, **calculating temporal differences at varying intervals**, captures both slow and fast movements, enriching the model's understanding of dynamic actions. These features are processed through separate convolutional layers and then concatenated to form a comprehensive representation that combines spatial configurations and temporal dynamics.
  Following the concatenation of spatial and temporal features, the combined data passes through additional convolutional layers to fully integrate and refine the information for accurate action classification.

  DD-Net's design **minimizes** computational overhead by **strategically reducing feature dimensionality** and optimizing the network's depth and width. This efficient processing allows DD-Net to achieve very high processing speeds, suitable for real-time applications where rapid response is essential.

- **Dataset:**

  - **SHREC:** A dataset focusing on hand gestures, derived from depth cameras capturing detailed 3D skeletal data.
  - **JHMDB:** A more general dataset that includes body actions derived from 2D skeletal data interpreted from video.

- **Performance Analysis:** DD-Net achieves state-of-the-art performance on both datasets, outperforming existing methods in terms of both accuracy and speed:

  - **On SHREC:** DD-Net achieves up to 96.3% accuracy with a processing speed of 2,200 FPS on a GPU, showcasing significant improvements over other methods that either lack speed or accuracy.

  - **On JHMDB:** It similarly excels by reaching an accuracy of 77.2% while maintaining a high processing rate, further demonstrating its applicability to different types of skeletal data.

- **Benefits:**
  - **Efficiency:** Extremely low computational requirements allow for high-speed processing on standard hardware.
  - **Accuracy:** Maintains high accuracy across diverse actions and conditions by effectively capturing and integrating multiple motion scales.
  - **Versatility:** Suitable for both 2D and 3D skeleton data, enhancing its applicability in various real-world scenarios.

- **Disadvantages:**
  - **Dataset Dependency:** The performance might vary significantly with less standardized datasets or those with poor skeletal tracking.
  - **Complexity in Training:** While the model itself is designed for efficiency, the dual-feature approach requires careful tuning during training to optimize performance.

4.2 Temporal-Decoupling Graph Convolution Network for Skeleton-Based Gesture Recognition (TD-GCN)

- **Introduction**: Skeleton-based gesture recognition has emerged as a crucial technique in various applications such as virtual reality, sign language translation, and human-computer interaction. Traditional methods leverage Graph Convolutional Networks (GCNs) which model the skeletal structure as a graph, treating joints as vertices and the bones between them as edges. However, a significant limitation of existing approaches is the use of a **static adjacency matrix** to represent joint connections, which **does not account** for the **dynamic nature of human movements** where the spatial relationships between joints can vary significantly across different frames of a gesture. This static representation restricts the ability of GCNs to effectively model the temporal variations inherent in skeletal data.

- **Proposal:** To address this limitation, the authors of the paper propose a novel architecture, the Temporal Decoupling Graph Convolutional Network (TD-GCN),which introduces **temporal-dependent adjacency matrices** that vary across different frames.

- **Architecture:** TD-GCN allows us capture the dynamic spatial-temporal relationships between joints more accurately. By decoupling the spatial and temporal dependencies, TD-GCN can adapt the graph topology dynamically over time, enhancing the ability to learn more discriminative features for gesture recognition.

  The network employs a modular design comprising Coupling Feature Learning (CFL) for high-level feature extraction, Channel Decoupling Learning (CDL) and Temporal Decoupling Learning (TDL) for generating channel-specific and frame-specific adjacency matrices, and Temporal-Channel Fusion (TCF) to integrate these features effectively.

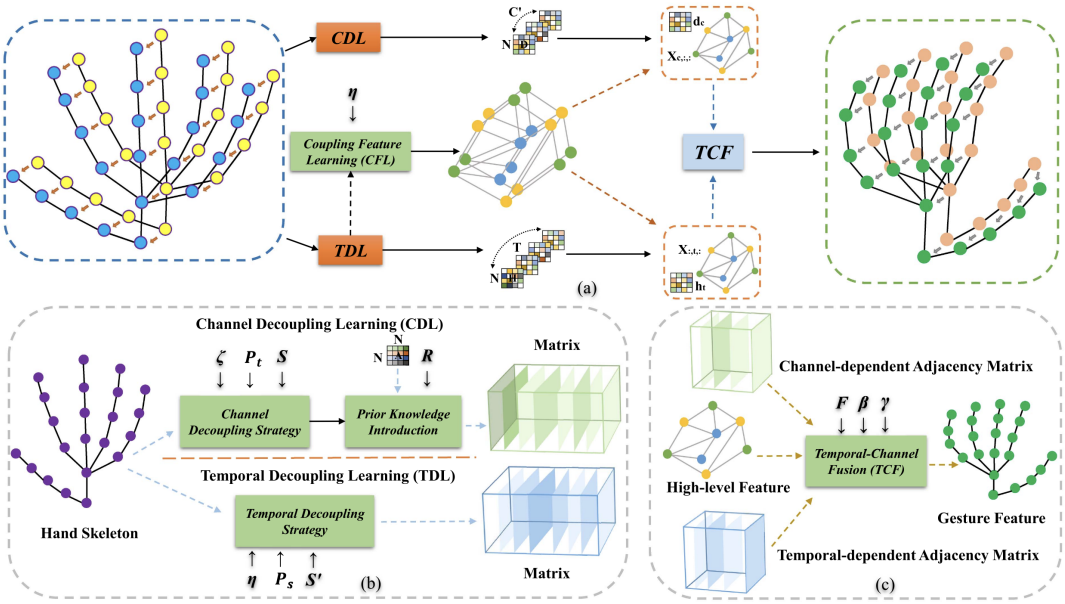

Figure 3: TD-GCN Architecture

- **Dataset:** Model has been tested on: **SHREC'17 Track**, **DHG-14/28**.
- **Performance Analysis:**
  - **SHREC'17 Track Dataset:**
    - Achieved an accuracy of 94.6% for recognizing 14 gestures, which is an improvement of approximately 3.2% over the prior state-of-the-art models.
    - For the more complex set of 28 gestures, the model recorded an accuracy of 91.9%, surpassing previous benchmarks by nearly 4.5
  - **DHG-14/28 Dataset:**
    - The model achieved an accuracy of 91.5% across both simpler and more complex gesture sets, showcasing its adaptability to different complexity levels in gestures.
  - **Comparison and Incremental Updates**
    - Compared to traditional GCN models, TD-GCN showed enhanced recognition rates, particularly in scenarios with high variability in gesture execution speed and style.
    - Notably, the model's performance in low latency settings (important for real-time applications) was consistently above 90%, highlighting its practical effectiveness in dynamic environments.
- **Disadvantages:** While the TD-GCN demonstrates superior accuracy and adaptability, its high computational requirements may restrict scalability. Additionally, the use of dynamic adjacency matrices increases computational overhead, thus potentially reducing training efficiency.

5 Baseline MLP Model Architecture

The baseline MLP model is designed to classify gesture data transformed into a flat feature vector. The architecture of the MLP is straightforward yet robust, ensuring a solid foundation for evaluating the effectiveness of more intricate models that will be introduced.

5.1 Model Description

The MLP model is constructed using **Keras**, with the following key features:

- **Input Layer:** The input layer accepts preprocessed data reshaped into a one-dimensional vector, ensuring compatibility with the dense layers of the model.
- **Dense Layers:** Multiple fully connected layers are employed, each consisting of 256 neurons and utilizing ReLU activation functions. This setup is chosen to introduce **non-linearity** into the model, enhancing its ability to learn complex patterns in the data.
- **Dropout:** A **dropout rate of 0.5** is applied after each dense layer to prevent **over-fitting** by randomly omitting a subset of features at each iteration during training.
- **Output Layer:** The final layer is a dense layer with a softmax activation function, designed to output a probability distribution across the predefined gesture classes.

5.2 Model Compilation and Training

The model is compiled with the **Adam optimize**r and **categorical crossentropy loss function**, focusing on classification accuracy as the primary metric. The training process involves:

- Using a concatenated feature set from two data sources, X_0 and X_1, as inputs.
- Conducting a training session over 500 epochs with full batch processing, which is particularly suitable given the manageable size of the dataset.
- Employing validation data to monitor and mitigate overfitting throughout the training phase.

5.3 Results

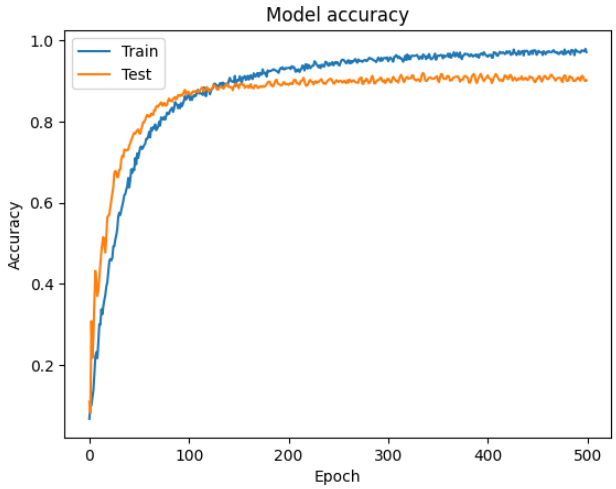

Figure 4: Train vs Test Model Accuracy

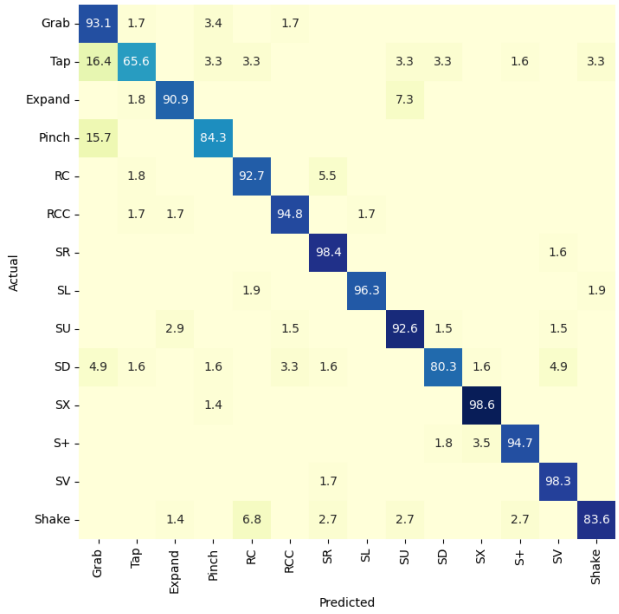

Figure 5: Confusion matrix (Actual vs Predicted gesture)

6 Novel Claim 1: Integration of Bone Modality Data into DDNet Architecture

Claim: Enhancing the DD-Net architecture by **integrating bone modality data** improves the model's ability to accurately classify dynamic hand gestures through a deeper understanding of **angular and distance changes** between connected joints over time.

Intuition: The development of this proposal was significantly inspired by the Temporal Decoupling Graph Convolutional Network (TD-GCN), the state-of-the-art architecture known for its ability to model the spatial topology of skeletons dynamically across different frames. While TD-GCN has shown high success in gesture recognition, its accuracy remains slightly below that of DD-Net, particularly in scenarios requiring rapid processing speeds and low computational overhead. Recognizing that TD-GCN's strength lies in its handling of temporal changes in bone connections—something not originally emphasized in DD-Net—we **hypothesized** that integrating a similar bone modality feature into DD-Net could capture more detailed gesture dynamics without compromising the network's efficiency. This integration aims to combine DD-Net's high performance and efficiency with the advanced skeletal dynamics modeling of TD-GCN, potentially setting a new standard for accuracy in dynamic hand gesture recognition systems.

6.1 Supporting Experiment:

Objective: To test the effectiveness of integrating bone modality data into the DD-Net architecture for improved gesture recognition accuracy.

6.2 Training Data:

- **Dataset Used:** SHREC'17 Track dataset, known for its dynamic hand gesture sequences which include a variety of hand movements captured via 3D skeletal data.
- **Details:** The dataset consists of multiple sequences of hand gestures, each labeled with specific gesture types. These sequences capture the 3D positions of hand joints over time, making it suitable for training models to recognize dynamic gestures.

Test Data:

- **Separation:** A separate test set from the SHREC'17 dataset was used to evaluate the model, ensuring that the training and testing data were distinct to avoid overfitting and to test the model's generalizability.

6.2.1 Feature Engineering and Architectural Adaptations

Enhancements to DD-Net's architecture were focused on integrating additional bone modality data, which involves understanding both the angular and distance dynamics of the skeleton structure. These changes are designed to refine the model's input features and accommodate the complexities of dynamic hand gestures:

- **Angular Changes:** We have introduced a feature that captures the angular variations between bones across consecutive frames. This metric is crucial for understanding the **rotational dynamics** of each joint, providing insight into how the orientation of hand parts changes through time. These angular measurements are expected to add a crucial layer of detail that assists in distinguishing gestures with similar trajectories but different rotational behaviors.

  The angle θ between two consecutive joint vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ is calculated using the formula:

  $$\theta = \cos^{-1}\left(\frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{\|\mathbf{v}_1\|\|\mathbf{v}_2\|}\right)$$

  where $\mathbf{v}_1 \cdot \mathbf{v}_2$ is the dot product of the vectors, and $\|\mathbf{v}_1\|$ and $\|\mathbf{v}_2\|$ are the magnitudes of the vectors.

- **Distance Changes:** In addition to angular data, we measure changes in the distances between joint pairs that form the skeletal structure of the hand. This feature helps in identifying gestures by analyzing the stretching or compression between bones, offering a direct measure of how the spatial configuration of the hand evolves during a gesture. This spatial dynamic is particularly useful for recognizing gestures that involve significant hand shape changes.

  The change in distance d between the positions of joints across frames is defined by:

  $$d = \|\mathbf{v}_2 - \mathbf{v}_1\| - \|\mathbf{v}_1 prev - \mathbf{v}_2 prev\|$$

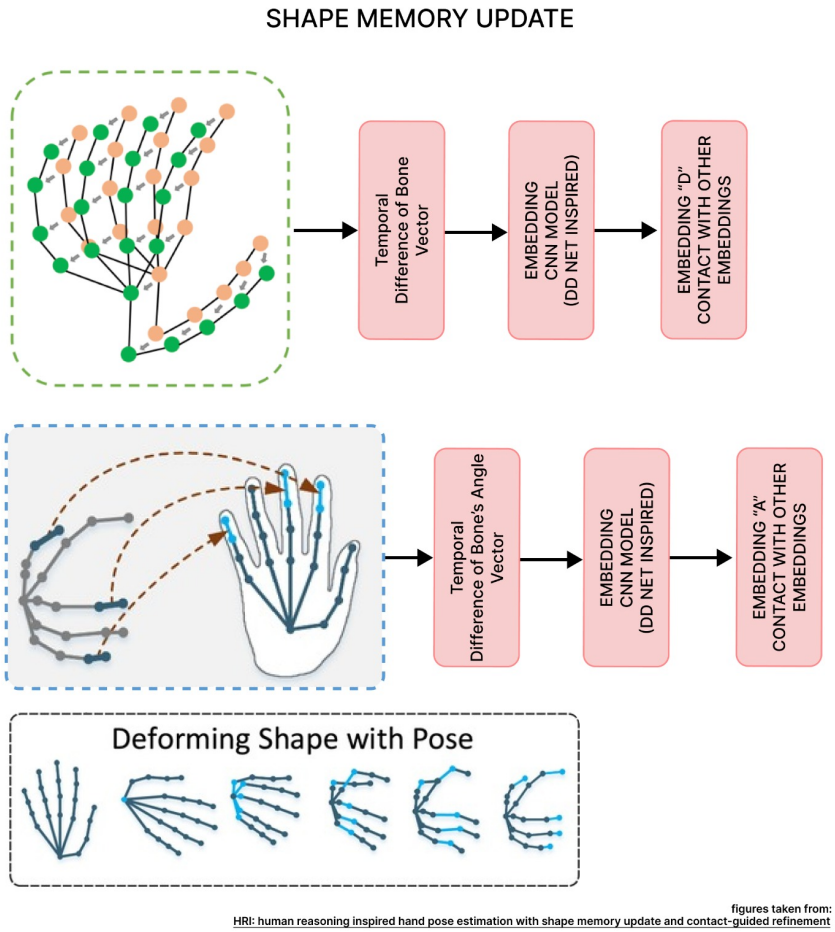

Figure 6: Bone vector and angle translation

- **Architectural Modifications:** To incorporate these new features, we modified the input layer of DD-Net to process and integrate both angular and distance changes alongside the traditional Joint Collection Distances (JCD). Significant adjustments were also made to the feature processing pipelines, which now include normalization steps to scale the new features appropriately for network training. These adjustments ensure that the new bone modality data are well-integrated into the network's learning process, enhancing the overall gesture recognition capabilities of DDNet without disrupting the pre-existing architecture's efficiency.

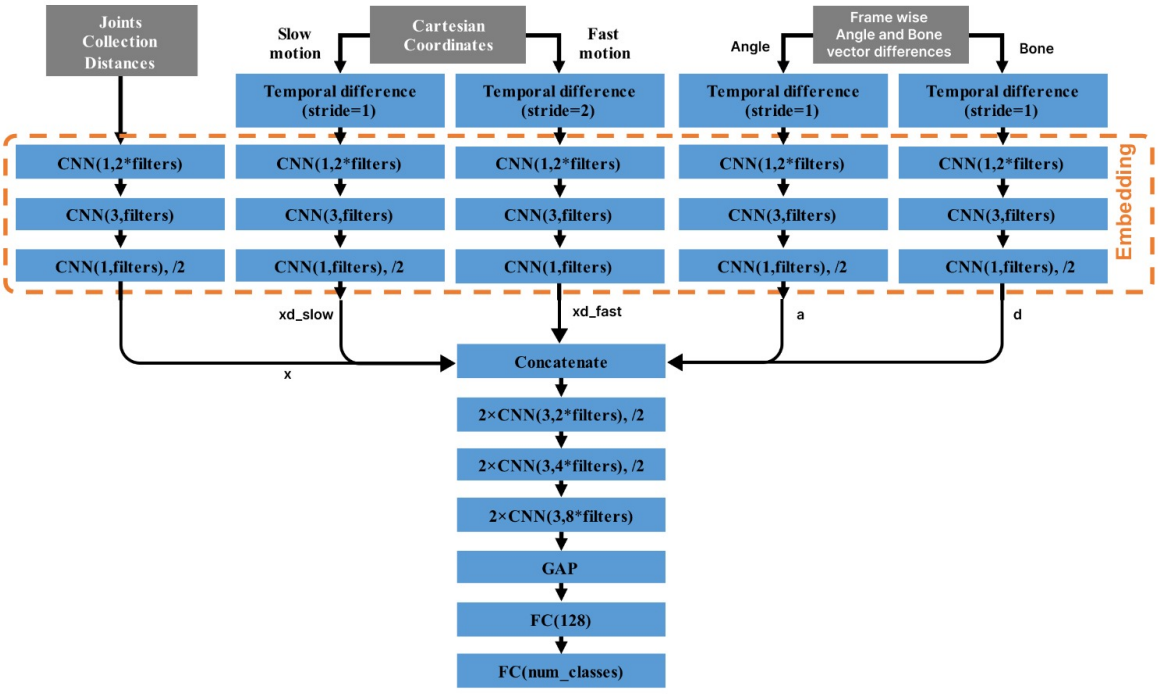

Figure 7: Network architecture after adding bone modality

Results:

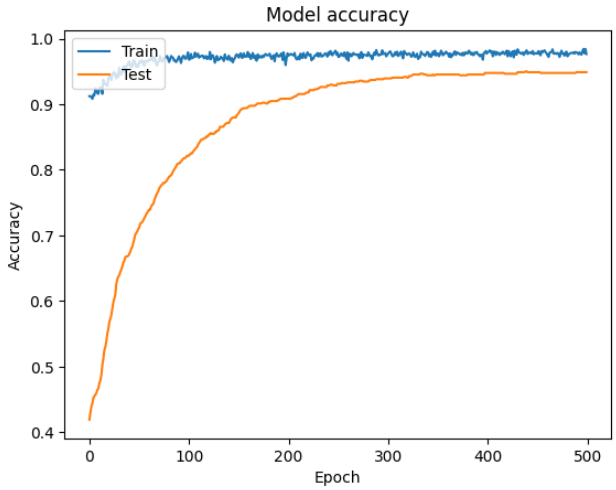

Figure 8: Train vs Test Model Accuracy

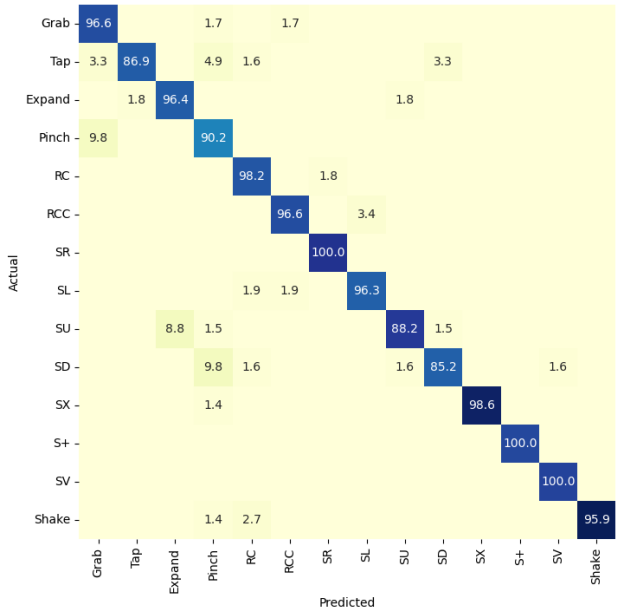

Figure 9: Confusion matrix (Actual vs Predicted gesture)

DD-NET Original model accuracy: 96.3 %
DD-NET model accuracy after bone modality addition : 94.8%

6.3 Illustrations and Lessons Learnt

- **Overall Performance:** The integration shows mixed effects on gesture recognition accuracy. Some gestures like *Expand* and *Grab* displayed improvements, while others such as *Tap*, *Swipe Up*, and *Swipe Down* saw decreases in accuracy.
- **Significant Improvements:** The accuracy for *Expand* increased from 92.7% to 96.4%, and *Grab* improved from 94.8% to 96.6%. These improvements suggest that the bone modality data helps in capturing more distinct bone movements associated with these gestures.
- **Decreases in Performance:** There were notable decreases in *Swipe Up* (from 92.6% to 88.2%), *Swipe Down* (from 93.4% to 85.2%), and *Shake* (from 97.3% to 95.9%). This might indicate that the additional features introduced noise or complexity that did not contribute positively to the model's ability to classify these particular gestures.
- **Consistent High Performers:** Gestures such as *Swipe Right* and *Stretch* maintained a 100% recognition rate, showing that these gestures are well-defined by both the original and the enhanced feature sets.

6.4 Conclusion

The integration of bone modality data has provided beneficial insights into specific gestures, improving the recognition accuracy for complex movements. However, it also introduced challenges, including potential overfitting or the inclusion of non-discriminative features for certain gestures. These findings suggest a need for further model tuning, possibly incorporating feature selection techniques to optimize the contribution of new data elements to the model's overall predictive capabilities.

7 Novel Claim 2: Improving Modality Embedding using Siamese Network

7.1 Claim

After observing the original confusion matrix, we notice that the similar gestures such as Tap and Pinch gets misclassified into Grab because of the similarity between the gestures in the dataset. All three are done with only the index finger and thumb, and they almost follow the same trajectory across time. This could be because the embedding being learnt across modalities for similar gestures, might be **overlapping** hence causing the network to misclassify. We propose instead of learning embedding for the the complete task, we would **pre-learn** embedding for a given modality using **Siamese Network** using **triplet loss**, hence learning to embed the similar gestures together and dissimilar far apart. After learning these embedding we train a classifier for the gesture classification.

7.2 Intuition

In considering the challenge of misclassifying similar gestures such as Tap and Pinch as Grab due to dataset similarities, we drew parallels to the domain of face recognition. Face recognition systems often encounter difficulties due to the subtle variations in facial structures among individuals, leading to misclassifications. To address this, researchers have successfully employed Siamese Networks, which are trained to generate distinct embeddings for different individuals. These networks leverage the concept of triplet loss to effectively learn embeddings by considering sets of examples with similar and dissimilar attributes.

In our context, where **similar gestures share common features and trajectories**, employing a Siamese Network with triplet loss offers a promising solution. By providing the network with a diverse set of examples encompassing both similar and dissimilar gestures, we can train it to learn embeddings that effectively differentiate between gestures. This approach ensures that the embeddings learned are sufficiently distinct, enabling the classifier to accurately classify gestures even amidst dataset similarities.

Furthermore, this approach not only enhances gesture classification accuracy but also contributes to the broader task of embedding creation. By training the network to make similar embeddings closer and dissimilar embeddings farther apart, we create a more generalized embedding creation layer. This layer captures features that the previous embedding creation method, trained solely on the classification task, might have overlooked. Thus, by leveraging the capabilities of Siamese Networks with triplet loss, we enhance both the specificity of **gesture classification** and the **generalization of embedding creation.**

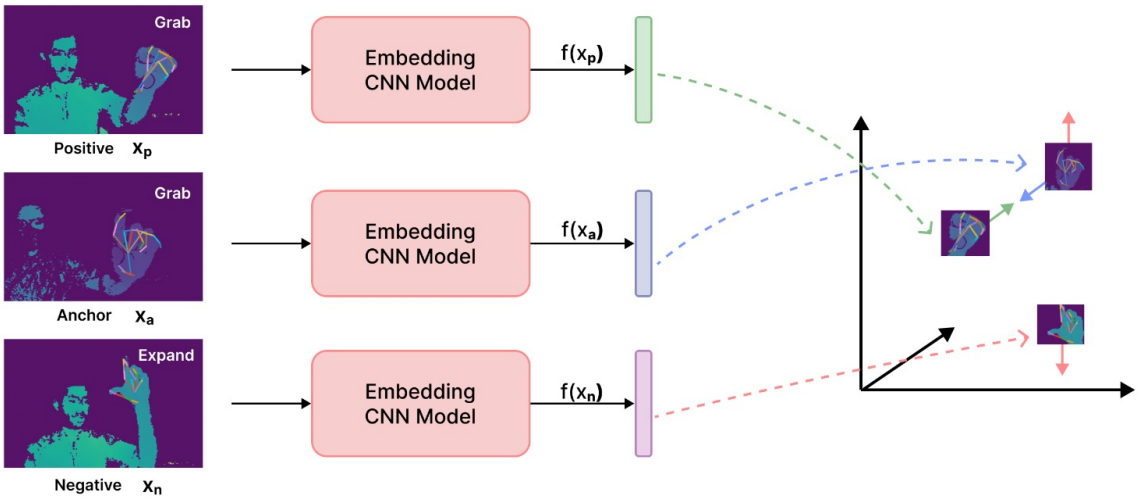

Figure 10: Siamese Network: To learn better embedding b/w gestures

7.3 Supporting Experiment

Objective: To evaluate the impact of using Siamese Networks with triplet loss to learn embeddings for Joint Collection Distances (JCD) and Two-scale Global Motion Features separately, and subsequently integrating them into the DDNet architecture for improved gesture recognition accuracy.

7.4 Training Data:

- **Dataset Used:** SHREC'17 Track dataset, renowned for its dynamic hand gesture sequences captured via 3D skeletal data.
  - **Details:** The dataset comprises numerous sequences of hand gestures, each annotated with specific gesture types. These sequences record the 3D positions of hand joints over time, making it suitable for training models to recognize dynamic gestures.

7.5 Experimental Procedure:

1. **Preprocessing:**
   - Extract Joint Collection Distances (JCD) and Two-scale Global Motion Features from the 3D skeletal data in the SHREC'17 Track dataset.
   - Generate a dataset with 3 elements: anchor gesture, positive gesture and negative gesture.
2. **Siamese Network Training:**
   - Design and train Siamese Networks with triplet loss separately for JCD and Two-scale Global Motion Features.
   - The Siamese Networks aim to learn embeddings that effectively differentiate between similar and dissimilar gestures within each feature.
3. **Evaluation:**
   - Assess the performance of the learned embeddings through classification tasks.
   - Train classifiers using the learned embeddings for JCD and Two-scale Global Motion Features.

- Measure classification accuracy on a separate test set from the SHREC'17 dataset, ensuring the model's generalizability.

4. **Integration into DD-Net Architecture:**
   - Integrate the learned embeddings from both modalities into the DD-Net architecture.
   - Assess the impact on gesture recognition accuracy by combining the features from JCD, Two-scale Global Motion Features, and the original features used in DDNet.
5. **Comparison:**
   - Compare the classification accuracies achieved by the classifiers trained on individual features and the combined feature set.
   - Analyze the improvement in gesture recognition accuracy achieved by incorporating embeddings learned using Siamese Networks with triplet loss.

7.6 Triplet Loss

The triplet loss compares the embeddings of three images: an anchor (A), a positive (P), and a negative (N) sample. The goal is to minimize the distance between the anchor and the positive sample while maximizing the distance between the anchor and the negative sample. This encourages the model to map similar images close together in the embedding space and dissimilar images far apart.

7.7 Equation

Mathematically, the triplet loss is often defined as follows:

$$L(A, P, N) = \max\{\|f(A) - f(P)\|_2^2 - \|f(A) - f(N)\|_2^2 + \alpha, 0\}$$

Where:

- A, P, and N are the anchor, positive, and negative samples, respectively.
- $f(\cdot)$ represents the embedding function (in this case, the Siamese network).
- $\|\cdot\|_2$ denotes the Euclidean distance.
- α is the margin, a hyperparameter that specifies the minimum difference between the distance of the anchor-positive pair and the anchor-negative pair.

In summary, the triplet loss enforces a geometric constraint on the embedding space, promoting meaningful representations where similar examples are clustered together, and dissimilar examples are separated by a margin.

7.8 Results

7.8.1 Embedding trained for Feature: JCD - Jacobian Collection Distances

Architecture	Accuracy (Test)	Param
SOTA Original DDNet (JCD only)	50.7%	0.34M
(Ours) Modified DDnet with Siamese Network (JCD only)	**69.8%** ↑ 19.1%	0.34M

Table 1: Performance Results for JCD Feature

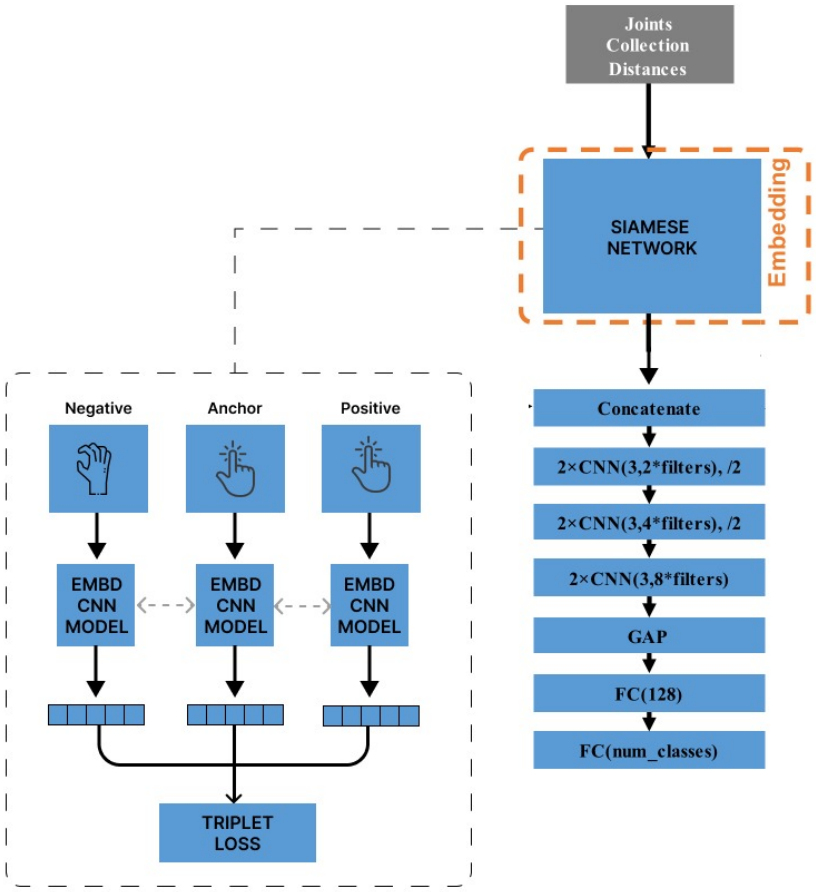

Figure 11: Siamese Network - Learning JCD Embedding

When the JCD Embeddings are trained in Siamese Network, **the classification accuracy increases by a whooping 19.1% beating the baseline reported in the state of the art DDNet from just a single feature while keeping the number of parameters same.** The training time for the network was 1000 seconds on Tesla T4 GPUs with 200 epochs, with 100,000 training examples. This shows the robustness of the Siamese network to distinguish between gestures from the baseline DDNet architecture that just learns for the cross entropy loss.

To extend this discovery we tried to learn embeddings for the rest of the features as well.

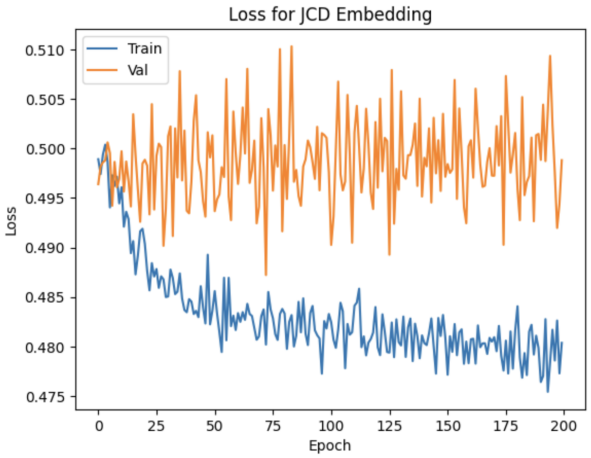

Figure 12: Siamese Network - JCD Embedding

Embedding trained for Other Features: Fast Global Motion & Slow Global Motion

Architecture	Accuracy (Test)
SOTA Original DDNet (Fast Motion & Slow Motion)	92.2%
(Ours) Modified DDnet with Siamese Network (Fast Motion & Slow Motion)	91.9%

Table 2: Performance Results for Fast Global Motion & Slow Global Motion

When employing a Siamese Network to train embeddings for Fast Global Motion and Slow Global Motion features, our modified DD-Net demonstrates a marginal decrease in accuracy of 0.3% compared to the state-of-the-art baseline. This slight reduction suggests that for these particular features, the Siamese Network may not provide significant improvement over the baseline DDNet architecture, which already achieves high accuracy. Unlike the substantial boost observed with JCD embeddings, the performance for Fast Global Motion and Slow Global Motion remains largely unchanged, indicating potentially different characteristics in feature representation and discriminative power.

7.8.2 Combined Performance from all Features

Architecture	Accuracy (Test)
SOTA Original DDNet (JCD, Fast Motion & Slow Motion)	96.3%
(Ours) Modified DDnet with Siamese Network (JCD, Fast Motion & Slow Motion)	94.3%

Table 3: Performance Results for JCD, Fast Global Motion & Slow Global Motion

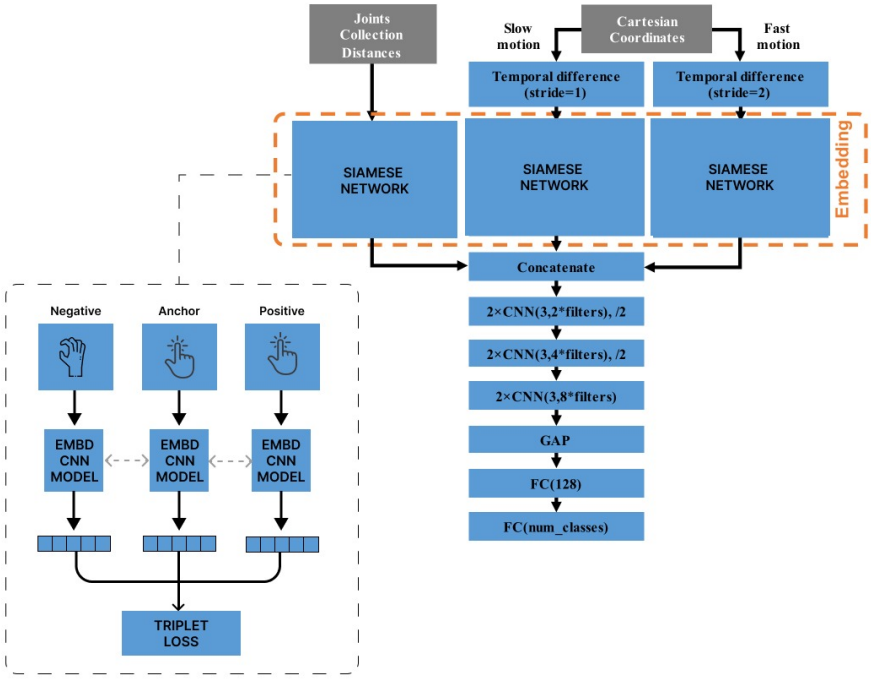

Figure 13: Siamese Network for all features

Incorporating Siamese Networks for training embedding across multiple features, including JCD, Fast Global Motion, and Slow Global Motion, our modified DD-Net showcases a high accuracy of 94.3%. While this represents a slight reduction compared to the state-of-the-art baseline of 96.3% but might showcase better one-shot learning.

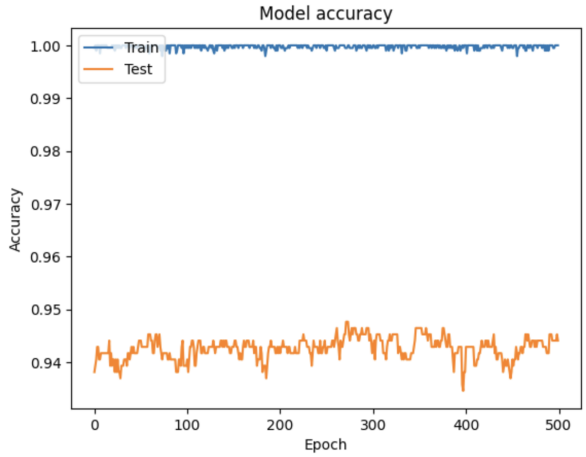

Figure 14: Accuracy of Classifier on Modified DDnet with Siamese Network

8 Testing Model on Gestures

To test for new gestures, apart from those available in the **SHREC** dataset, we created a pipeline that allows us to dynamically create a gesture using **MediaPipe**.

Figure 15: Creating new gestures using MediaPipe

When performing gestures, we used relative coordinates for drawing the node data points and their edges. While storing the temporal movement the data, we used the world coordinates of the skeleton frame.

After performing several repetitions of the same gesture (say 15,or 30 times), the data along with its label was converted into pickle format.

The MediaPipe library segments the hand into 21 nodes, whereas the model trained on the SHREC dataset segments the hand into 22 nodes. To address this discrepancy, we retrained the model to use 21 nodes by removing the palm node. Subsequently, we used this adjusted model to predict gestures that it had already been trained to recognize.

9 GitHub Repository

GitHub Link: `https://github.com/thetushargoyal/deep-learning-project`

10 References:

Temporal Decoupling - Graph Convolutional Network

Make Skeleton-based Action Recognition Model Smaller, Faster and Better

Siamese network with Keras, TensorFlow, and Deep Learning