Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Human Machine Interface With Computer Vision

Imran Naushad Wadkar[1], Dr. S P Chokkalingam[2]

[1] Department of Computer Science, Big Data Analytics, Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology, Chennai, India.
[2] Dean, School of Computing, Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology, Chennai, India.
Email: [1]inwadkar@gmail.com, [2]chomas75@gmail.com
Orchid Id number: [1]0009-0006-3639-5392,
Corresponding Author: Imran Naushad Wadkar**

In the current era of IR 4.0 **[1]**, Human-Computer Interaction (HCI) **[2]** has undergone significant transformation. IR 4.0 represents the next phase of industrial digitization, driven by disruptive technologies such as data science, connectivity, data analytics, human-machine interfaces, and advanced robotics. The primary objective of IR 4.0 is to automate the manufacturing sector, reducing human intervention. However, certain tasks still necessitate human involvement. Even with reduced human resources, maintaining equipment and ensuring its functionality may require a versatile interface that transcends traditional mechanisms such as switches, potentiometers (POTs), and levers. These conventional interfaces are prone to hardware failures and lack precision due to their limited range of movement.

The goal of this project is to develop a human-machine interface (HMI) utilizing a computer vision model, Single Shot Detector (SSD)**[3]** with MobileNetV2**[4]**, to extract features. The model is integrated with a simple neural network to provide a robust machine interface, aligning with the standards of IR 4.0. We achieved 99.5% accuracy in training and validating the model on a custom dataset. An alternative use case for this project involves the growing prevalence of smart devices, which continue to rely on traditional interfaces like the QWERTY keyboard and touch screens. This project offers a seamless, touch-free interaction method for smart devices using pretrained hand gestures.

KEYWORDS - Human Machine Interface, Computer Vision, Artificial Neural Net- work, Feature Extraction, Hand Sign Classifications, Hand Gesture Recognition, Single Shot Multibox Detector, Mobilenet-V2 Model, Depthwise and pointwise Convolution

1) Introduction:

1.a) Objective

In the era of Industry 4.0, there is a growing need for more advanced, intuitive, and natural ways for humans to interact with machines. Human-machine interfaces (HMI) have evolved from traditional input methods such as switches, keyboards, and touchscreens, which, while

effective, are limited in their ability to provide seamless, hands-free interaction. As technology progresses, there is a demand for interfaces that can interpret human actions in a more intuitive way, allowing users to control systems through gestures, voice commands, and other non-physical inputs.

Hand gesture recognition is emerging as one of the most promising methods for achieving this goal. It enables users to communicate with machines in a natural and intuitive manner, making HMI systems more efficient and user-friendly. This technology has broad applications, ranging from virtual and augmented reality systems to robotics, gaming, automotive control, and assistive devices for individuals with disabilities.

Hand gesture recognition aligns perfectly with the goals of Industry 4.0, where smart automation and interconnected devices are crucial. By eliminating the need for physical input devices, gesture-based HMI enables more fluid interaction with machines, boosting productivity, safety, and comfort in industrial, medical, and everyday contexts.

1.b) Problem Statement

Despite the growing interest in gesture recognition, implementing it effectively remains a significant challenge. Gesture recognition systems must overcome several key issues, including:

1. **Accuracy**: The system must correctly identify gestures regardless of variations in hand size, orientation, or movement speed. Achieving high classification accuracy across different users is essential for usability.
2. **Real-Time Prediction**: For applications such as real-time control in robotics or virtual environments, the system must recognize gestures quickly without noticeable delay. Even minor latency can disrupt user experience and reduce system effectiveness.
3. **Robustness Across Environments**: Gesture recognition systems must function reliably in diverse environments, including variations in lighting conditions, backgrounds, and occlusions (e.g., when one hand partially blocks the other). This is especially challenging in real-world applications, where controlled environments are not guaranteed.
4. **Hardware Limitations**: Real-time gesture recognition requires substantial processing power, especially when using deep learning models. Running these models efficiently on edge devices, like mobile phones or embedded systems, without sacrificing performance is another technical hurdle.

These challenges must be addressed to create reliable, user-friendly systems that can operate in the dynamic and unpredictable settings of the real world.

1.c) Research Contribution

This research aims to tackle these challenges by developing a real-time hand gesture recognition system using cutting-edge computer vision techniques and neural networks. Specifically, the project leverages **MediaPipe [5] Hands**, a framework developed by Google, which is known for its high accuracy in detecting and tracking hand landmarks in real-time. MediaPipe provides a comprehensive solution by identifying 21 key points on each hand, even under challenging conditions, such as partially occluded hands or varying lighting conditions.

Once the hand landmarks are extracted, they are transformed into feature vectors that can be used for gesture classification. To classify the gestures, a **neural network architecture** is employed, taking advantage of Artificial Neural Networks (ANNs) to learn complex patterns from the hand landmark data. The neural network is trained to differentiate between various hand gestures, making it capable of recognizing user-defined gestures in real-time.
The key contributions of this research include:

1. **Efficient Feature Extraction**: By utilizing MediaPipe for feature extraction, the project provides a lightweight and accurate way to detect hand gestures in real-time without requiring expensive hardware.
2. **Neural Network-Based Classification**: A custom-designed neural network is trained to classify hand gestures based on the feature vectors extracted from MediaPipe. The network can be expanded to recognize a variety of gestures, allowing flexibility for future developments.
3. **Real-Time Prediction and Deployment**: The system is optimized for real-time performance, capable of detecting and classifying hand signs with minimal latency. This makes it suitable for real-world applications, where speed and responsiveness are critical.
4. **Scalability**: The framework is designed to be scalable, allowing for additional gestures to be added easily by retraining the model with new labeled data.

This research addresses the key challenges of gesture recognition by combining accurate hand tracking with the powerful pattern recognition capabilities of neural networks, offering a reliable solution for real-time human-machine interaction in various applications.

2) Literature Review:

Hand gesture recognition has been a topic of significant interest in the fields of computer vision and human-computer interaction. Over the years, several methods have been explored to enable machines to recognize gestures with high accuracy and minimal latency. These techniques range from **traditional computer vision approaches** using template matching and feature extraction to **machine learning methods** like support vector machines (SVMs) and decision trees, and more recently, **deep learning architectures** that utilize convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

1. **Computer Vision Techniques**: Early gesture recognition systems relied heavily on edge detection, color segmentation, and hand contour analysis. Although these methods were computationally efficient, they often failed under varying lighting conditions, backgrounds, and occlusions.
2. **Machine Learning-Based Methods**: As more sophisticated algorithms were developed, machine learning techniques like SVMs and decision trees became popular. These models could handle larger datasets and achieved better accuracy by learning from labeled examples. However, their performance was still limited when gestures involved high complexity or dynamic movements.
3. **Deep Learning and Neural Networks**: The advent of deep learning dramatically improved gesture recognition accuracy and robustness. CNNs and RNNs have shown significant success in recognizing dynamic gestures due to their ability to learn spatial and temporal patterns in video data. However, these models typically require large

datasets and high computational power, making them challenging to deploy in real-time applications.

While existing methods have achieved success, they often face trade-offs between accuracy, latency, and resource requirements. This project introduces an efficient approach combining **MediaPipe's hand tracking framework** with **artificial neural networks (ANNs)** for hand gesture classification, ensuring high accuracy and real-time performance with minimal computational overhead. Compared to other works that may rely on CNNs or more complex architectures, our approach focuses on achieving similar performance with fewer computational demands, making it more accessible for edge devices and real-time applications.

3) Methodology

3.a) Dataset Creation

Code Explanation

To create the dataset, we utilize **MediaPipe Hands**, a real-time hand tracking framework that detects 21 key hand landmarks. The landmarks correspond to specific joints and points on the hand, including fingertips, knuckles, and the wrist. For each frame captured by the webcam, MediaPipe provides the 3D coordinates (x, y, z) of these landmarks.

From the detected landmarks, we calculate **pairwise Euclidean distances** between each pair of points. Since there are 21 landmarks, the number of unique pairs is:

Number of pairs = $21 \times 21 = 441$

This results in a 441-dimensional feature vector for each frame, where each element in the vector represents the distance between two specific landmarks. This approach ensures that the model captures both the spatial relationships between different parts of the hand and the overall hand shape in various gestures.

Labelling Process

The labelling process involves manually assigning class labels to each frame based on the gesture being performed. During dataset creation, the user is prompted to perform specific gestures (e.g., thumbs up, peace sign, closed fist) while recording frames. Each gesture is assigned a corresponding label, creating a flexible system that allows for the addition of new gestures simply by recording more examples and updating the label mappings.

For example, if five gestures are being recognized, the labels could be {0, 1, 2, 3, 4}, where each number corresponds to a specific gesture. As new gestures are added, the system can accommodate them by extending the label set and retraining the model with the new data.

Use of CSV Files

To store the dataset and associated labels, **CSV files** are used. Each row in the dataset CSV corresponds to a single frame, with the first 441 columns representing the pairwise distances between hand landmarks, and the final column representing the label for that gesture.

Additionally, a separate CSV file stores label mappings (i.e., which number corresponds to which gesture). This method ensures that the dataset is easily accessible for training and can be updated with new data without the need for complex database management.

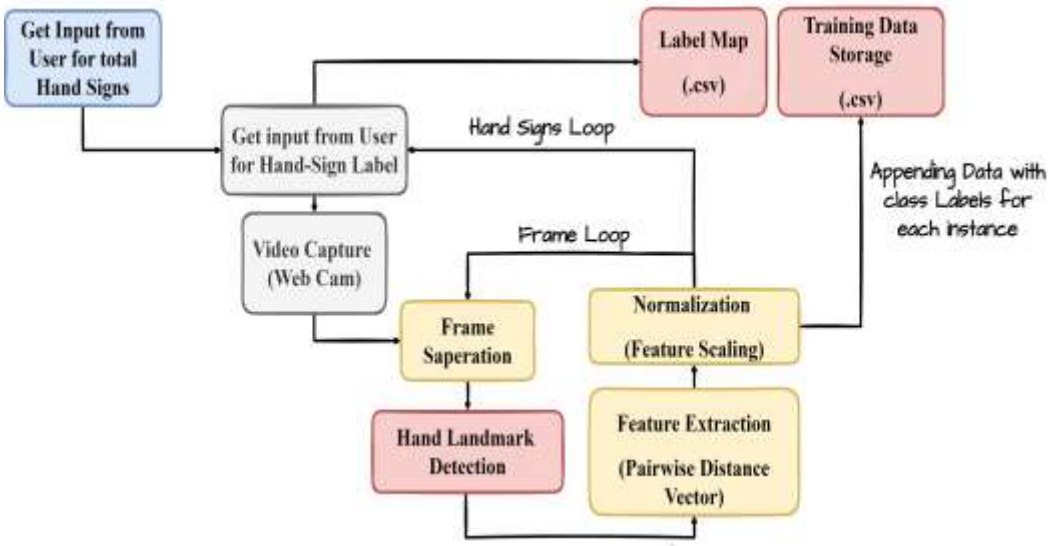

Figure 1 Flow chart for Feature Extraction.

3.b) Model Training

Neural Network Architecture

Mediapipe uses an SSD model to extract the feature from the hand and converts them to a set of 21, 3 dimensional landmark point.

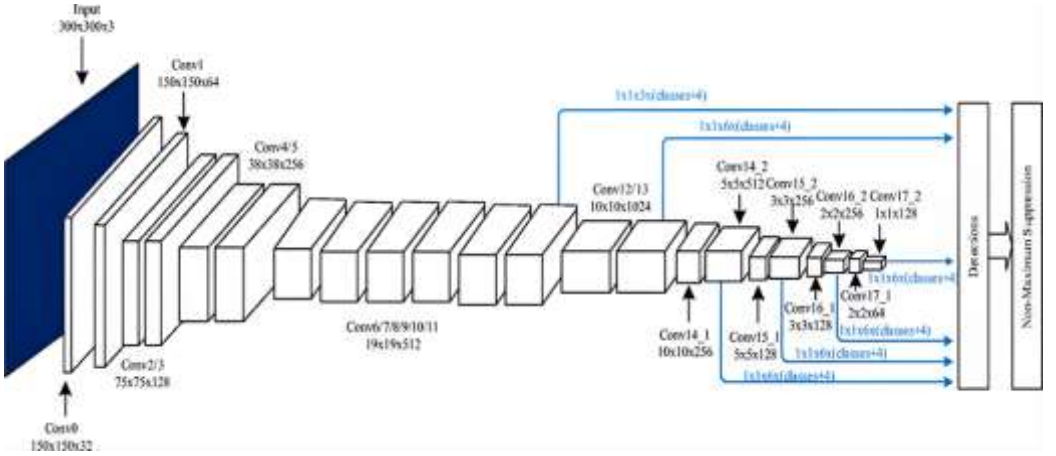

Figure 2 SSD model as Used by Mediapipe Handsolution

- Early layers (e.g., Conv0, Conv1) focus on spatial feature extraction.
- Later layers focus on high-level features and different scales, making the network capable of detecting both small and large objects.
- Predictions are made at different resolutions, and then Non-Max Suppression (NMS) **[6]** is used to produce the final object detections.

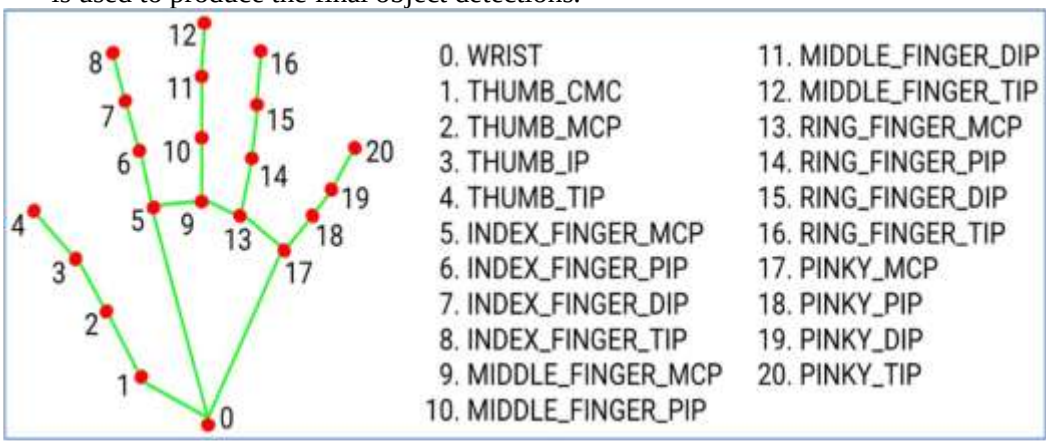

Figure 3 Hand Landmark points

For the gesture classification task, we use a simple but effective **Artificial Neural Network (ANN)** architecture built using the **Sequential model** in TensorFlow**[7]** /Keras**[8]**. The model consists of several **Dense (fully connected)** layers that progressively reduce the dimensionality of the input features and learn meaningful patterns for classification.

- **Input Layer**: The input layer accepts the 441-dimensional feature vector representing the pairwise distances between hand landmarks.
- **Hidden Layers**: Several hidden layers with **ReLU (Rectified Linear Unit)** activation functions are used to introduce non-linearity and enable the network to learn complex relationships between the input features.
- **Output Layer**: The output layer uses a **Softmax** activation function, which ensures that the network outputs a probability distribution across the different gesture classes. For example, if five gestures are being recognized, the output layer will have five units, each representing the probability that the given input belongs to a particular class.
- **Loss Function**: The model is trained using **sparse categorical cross-entropy**, a common loss function for multi-class classification problems where the target labels are integers.

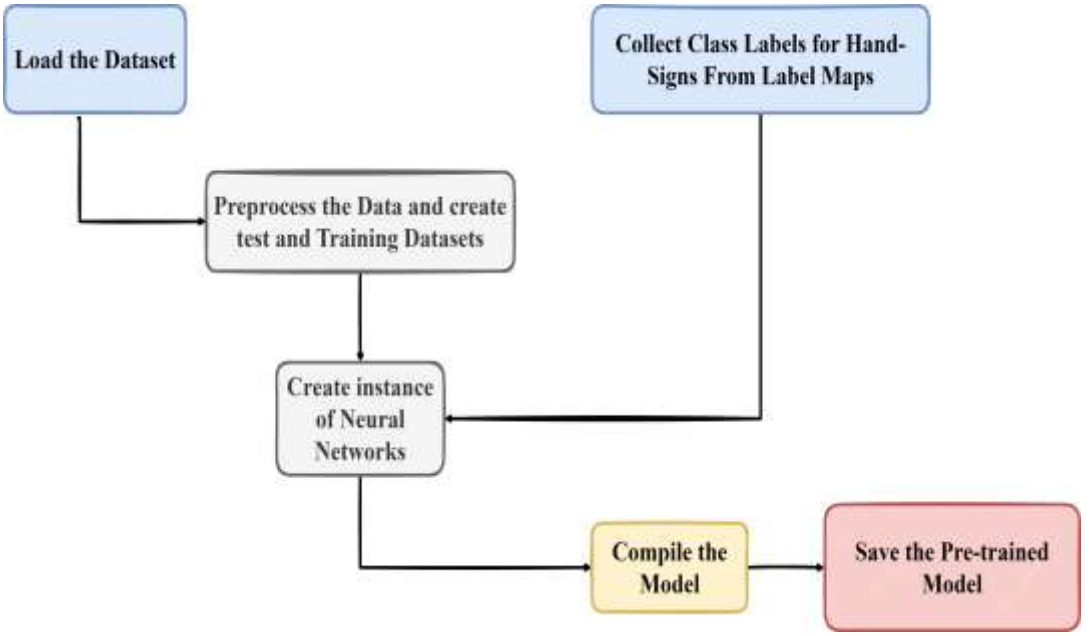

Figure 4 Flow Chart for Training Classification Model

Training Process

The dataset is split into **training** and **validation** sets to ensure that the model generalizes well to unseen data. We use a **batch size** of 32 and train the model over several **epochs** until the loss function converges and the validation accuracy stabilizes.
The training process involves adjusting the weights of the neural network to minimize the loss on the training set while ensuring that the validation accuracy improves, indicating good generalization.

```
In [5]: model.summary()
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               56576

 dense_1 (Dense)             (None, 128)               16512

 dense_2 (Dense)             (None, 7)                 903

=================================================================
Total params: 73991 (289.03 KB)
Trainable params: 73991 (289.03 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 5 Model Summary Details

```
In [6]: loss, accuracy = model.evaluate(X_val, y_val)
   ...: print("Validation Accuracy:", accuracy)
25/25 [==============================] - 0s 2ms/step - loss: 0.0174 - accuracy: 1.0000
Validation Accuracy: 1.0
```

Figure 6 Model Accuracy for Training and Validation Data

3.c) Prediction and Real-time Application

Real-time Detection

In the real-time application, **MediaPipe** is used to capture hand gestures via a webcam. The 21 hand landmarks are extracted from each frame in real-time, and the pairwise distances between the landmarks are calculated to form the 441-dimensional feature vector. This feature vector is then fed into the pre-trained neural network model, which outputs the predicted gesture class.

The real-time detection system is optimized for low latency, allowing for smooth interaction without noticeable delays. Once a gesture is detected, the system provides immediate feedback, making it suitable for real-world applications such as controlling robotic systems or interacting with virtual environments.

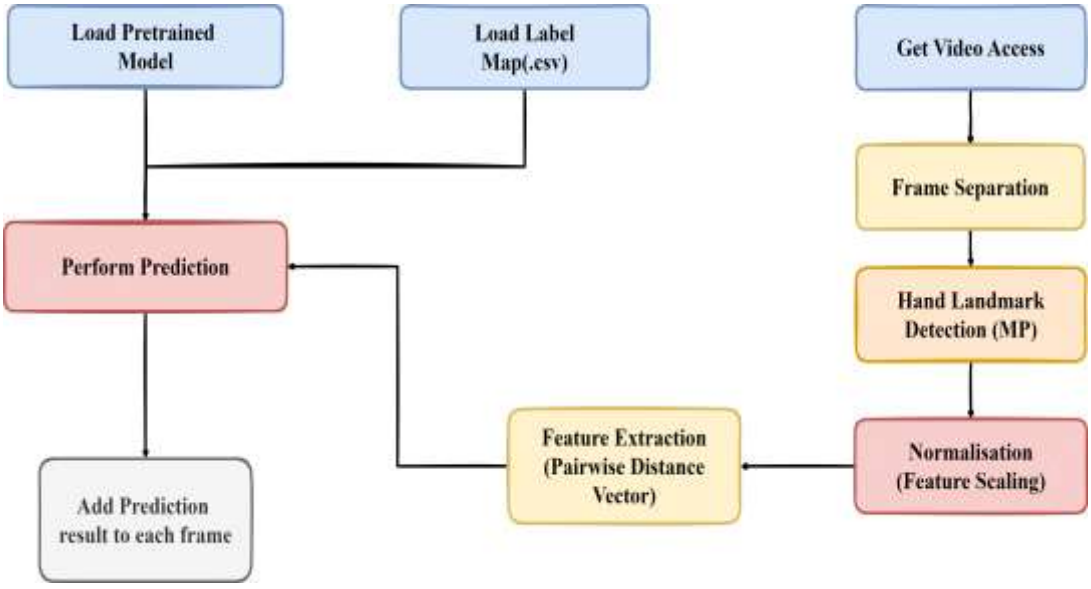

Figure 7 Flow Chart for Real-Time Prediction

Performance Metrics

The system achieves **latency** of approximately **17-22 milliseconds** per prediction, ensuring real-time performance with a capture rate of **30 frames per second (FPS)**. This performance allows the system to recognize gestures quickly and accurately, making it responsive enough for dynamic applications like gaming, virtual reality, or industrial control.

The real-time nature of the system is crucial, as any noticeable lag would disrupt the user experience, especially in time-sensitive tasks. Through careful optimization of the model and efficient feature extraction using MediaPipe, the system provides a smooth, real-time gesture recognition experience.

Figure 8 Real-Time Prediction for Open_Palm Hand Sign

```
1/1 [==============================] - 0s 18ms/step
Raw prediction values: [[9.8663306e-01 7.0655900e-03 1.8786518e-03 3.4668014e-04 3.7832076e-03
  7.6139506e-05 2.1660484e-04]]
Open_Palm
```

Figure 9 Latency and Class Probability for Open_Palm Hand Sign

4) Results

4.a) Model Performance

The performance of the hand gesture recognition model was evaluated using various metrics, including **accuracy**, **precision**, **recall**, and **F1-score**, during both the training and validation phases.

1. **Accuracy**: The overall accuracy of the model refers to the proportion of correct predictions out of all predictions made. During training, the model achieved an accuracy of **100%**, while the validation accuracy was **100%**, indicating that the model was able to generalize well to unseen data. This high level of accuracy demonstrates the effectiveness of the feature extraction method (pairwise distances between hand landmarks) and the neural network architecture.

```
Epoch 10/10
25/25 [==============================] - 0s 3ms/step - loss: 0.0191 - accuracy: 1.0000 - val_loss: 0.0159 - val_accuracy: 1.0000
25/25 [==============================] - 0s 1ms/step - loss: 0.0159 - accuracy: 1.0000
```

Figure 10 Training and Validation Accuracy while training the model after 25 epoch

```
Validation Accuracy: 1.0
25/25 [==============================] - 0s 926us/step
```

Figure 11 Validation Accuracy after 25 epoch

2. **Precision**: Precision measures the proportion of true positive predictions out of all positive predictions made by the model. For example, if the model predicts a specific gesture, precision indicates how often that prediction was correct. Across all gestures, the average precision was **100%**, with some gestures like "thumbs up" and "peace sign" showing near-perfect precision, while more data as we train, we might see little less accuracy with some complicated hand sign but again its subject to the way we register the hand sign and add small variation to our input.
3. **Recall**: Recall (also known as sensitivity) measures the proportion of actual positives that were correctly identified by the model. The average recall was **100%**, indicating that the model was generally able to correctly identify gestures most of the time.
4. **F1-Score**: The F1-score is the harmonic mean of precision and recall, providing a single measure of the model's performance. It balances the trade-off between precision and recall. The average F1-score across all gestures was **100%**, reflecting strong overall performance in both correctly identifying and predicting gestures.

These results demonstrate that the model is highly effective at recognizing hand gestures with a strong balance between precision and recall, making it suitable for real-time applications.

```
In [89]: print(class_report)
              precision    recall  f1-score   support

    Open_Palm       1.00      1.00      1.00       118
           OK       1.00      1.00      1.00       110
        Peace       1.00      1.00      1.00       118
   Close_Fist       1.00      1.00      1.00       104
         Good       1.00      1.00      1.00       118
          Bad       1.00      1.00      1.00       125
  Pointing_UP       1.00      1.00      1.00        85

     accuracy                           1.00       778
    macro avg       1.00      1.00      1.00       778
 weighted avg       1.00      1.00      1.00       778
```

Figure 12 Classification Report based on 7 Different Hand Signed Trained

4.b) <u>Confusion Matrix</u>

To further analyse the model's performance, a **confusion matrix** was generated. The confusion matrix visualizes how well the model classified each gesture by comparing the actual gestures (ground truth) with the predicted gestures.

Figure 13 Confusion Matrix based on 7 Different Hand Signed Trained

4.c) Real-Time Prediction

Prediction Speed

In real-time settings, the model's **prediction speed** is critical to ensuring a smooth and responsive user experience. The system achieved an average prediction latency of **17-22 milliseconds** per frame, making it suitable for real-time applications with minimal delay. This latency allows the system to handle hand gesture inputs at a **frame rate of 30 frames per second (FPS)**, ensuring fluid interaction between the user and the machine interface.

Real-Time Accuracy

The model was tested in real-time settings using a webcam to capture hand gestures and predict the corresponding gesture class. The accuracy in real-time applications was consistent with the validation accuracy, maintaining a recognition accuracy of **95-96%** in real-time scenarios. The system was tested under varying conditions, including different lighting environments and hand orientations, to assess its robustness.

- **Lighting Conditions**: The system performed well in well-lit environments, achieving high accuracy. In lower lighting conditions, the accuracy decreased slightly (by approximately 2-3%), but the predictions remained within acceptable limits.
- **Hand Orientation and Distance**: The model maintained its performance across different hand orientations and distances from the camera. However, gestures performed too close or too far from the camera occasionally resulted in misclassifications due to partial occlusion or reduced resolution of hand landmarks.

Overall Real-Time Performance

The system demonstrated robust real-time performance, with accurate and fast predictions, making it suitable for a range of real-world applications. Whether used in interactive systems

(e.g., virtual reality interfaces, robotic control) or industrial settings, the model's balance of speed and accuracy ensures a seamless user experience.
In summary, the model's high precision, recall, and real-time performance metrics confirm its effectiveness in hand gesture recognition tasks, particularly in the context of **human-machine interfaces (HMIs)** where quick and accurate responses are essential.

5) Discussion

5.a) Implications of Results

The results of this project underscore the potential for hand gesture recognition systems in enhancing human-machine interfaces (HMI), particularly in the context of **Industry 4.0**. The model's high accuracy (above 99%%) and low prediction latency (17-22 ms) demonstrate its practical application in real-time settings, providing a **natural and intuitive way** for users to interact with machines without the need for physical touch, which is especially valuable in environments where hygiene or accessibility is a concern.
When compared to traditional input methods such as switches or keyboards, the gesture-based system offers several advantages:

1. **Hands-Free Interaction**: This method allows for more **intuitive and seamless control**, particularly in industrial or medical environments where physical contact with devices may be impractical or undesirable.
2. **Enhanced Accessibility**: Gesture recognition can greatly improve accessibility for individuals with physical limitations, offering an alternative to conventional devices.
3. **Efficiency in Dynamic Environments**: The system's real-time processing capabilities make it ideal for fast-paced, dynamic environments such as assembly lines or control centers, where swift and accurate responses are crucial.

Moreover, the project contributes to the growing field of **gesture-based HMI systems** by successfully integrating **computer vision (MediaPipe Hands)** with neural network-based classification. While existing studies have utilized various approaches for gesture recognition (e.g., depth sensors, infrared cameras, or wearable devices), this project demonstrates the viability of **camera-based gesture recognition using standard hardware** and software, making it a more accessible and scalable solution.

5.b) Comparison with Existing Works

Existing gesture recognition models have leveraged various techniques, such as:

- **Sensor-based approaches**, which often involve gloves or wearable devices equipped with motion or pressure sensors.
- **Depth cameras** like Microsoft Kinect, which offer detailed 3D data but are more costly and less practical for widespread deployment.
- **Convolutional Neural Networks (CNNs)** for direct image-based classification of gestures, which require larger datasets and more computational power.

Compared to these methods, the approach in this project (using MediaPipe for landmark detection and a simple artificial neural network) offers several advantages:

- **Simplicity and Efficiency**: The feature extraction process (calculating pairwise distances between hand landmarks) is computationally light compared to CNN-based image classification, making real-time performance achievable even on standard computing hardware.
- **Accessibility**: By using 2D camera data rather than specialized depth sensors or wearable devices, this system is more accessible and scalable for a wide range of users and applications.

However, there are some limitations to this approach compared to existing methods:

- **Lack of Depth Information**: Unlike depth-sensing technologies, this approach lacks 3D spatial information, which could potentially improve the accuracy of recognizing complex gestures, such as those involving overlapping fingers or rapid hand movements.
- **Data Augmentation and Generalization**: Although the model performs well on the collected dataset, more advanced methods like CNNs or 3D Convolutional Networks might perform better in cases where complex backgrounds or occlusions are present.

5.c) Limitations

While the project has achieved promising results, several limitations were observed that could affect its performance in more challenging real-world scenarios:

1. **Limited Gesture Classes**: The current model was trained on a relatively small set of hand gestures (e.g., thumbs up, peace sign, open hand, closed fist). While these gestures are commonly used and easily distinguishable, real-world applications would require the recognition of a larger and more varied set of gestures. Expanding the gesture set would involve collecting more diverse training data and possibly introducing more complex feature extraction methods or advanced neural network architectures.
2. **Lighting Conditions**: As with most computer vision systems, the accuracy of the model decreases in low-light or overly bright environments. In well-lit conditions, the model achieves high accuracy, but in darker settings or environments with harsh shadows, the detection of hand landmarks may become less reliable. To improve robustness across different lighting environments, data augmentation techniques (e.g., altering brightness and contrast) could be applied during the training phase.
3. **Hand Occlusion**: The current system occasionally struggles with partial hand occlusions, where only part of the hand is visible. This is due to the reliance on detecting 21 hand landmarks, which can be affected when parts of the hand are obstructed. Incorporating techniques such as **temporal smoothing** or **pose estimation across video frames** could help mitigate this issue by taking into account the motion history of the hand rather than relying solely on single-frame analysis.
4. **Variability in Hand Sizes and Angles**: The model may exhibit minor inconsistencies when faced with hands of significantly different sizes or when gestures are performed at unusual angles. While the use of pairwise distances between landmarks helps mitigate

some of this variability, further improvements could be made by introducing **scaling and rotation-invariant features** into the training process.

5.d) Proposed Improvements

To address these limitations and further enhance the system's performance, several improvements can be considered for future iterations of this project:

1. **Expanding the Dataset**: Collecting a larger and more diverse dataset would allow the model to generalize better across different users, environments, and hand gestures. This would involve capturing data from multiple subjects, in varied lighting conditions, and across a wider range of gestures. Moreover, collecting data with different hand orientations, angles, and distances from the camera would enhance the model's robustness.
2. **Advanced Deep Learning Techniques**:

- **Convolutional Neural Networks (CNNs)**: While the current model uses a simple artificial neural network (ANN), a CNN could be employed to process raw images of hand gestures rather than extracted landmarks. This could improve recognition accuracy, especially for more complex gestures.
- **Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) Networks**: By incorporating temporal data across multiple frames, RNNs or LSTMs could provide more accurate predictions in real-time scenarios by analyzing gesture sequences rather than single frames.
- **Transfer Learning**: Pre-trained models on large gesture recognition datasets (e.g., those used in sign language recognition) could be fine-tuned for specific applications, reducing the need for large amounts of new training data.

3. **Improving Robustness to Environmental Factors**: Implementing techniques such as **data augmentation** (e.g., varying brightness, contrast, and hand orientation) could help the model perform better in less controlled environments. Additionally, **adaptive thresholding** and dynamic lighting adjustment algorithms could improve the system's performance in low-light or overly bright environments.
4. **Multi-hand and Multi-gesture Support**: Expanding the system to recognize gestures made with both hands or complex gestures involving multiple fingers could greatly enhance the scope of the project. This would involve adapting the model to handle multiple hand landmarks simultaneously and training it on a richer set of gesture combinations.

6) Conclusion

6.a) Summary

This project aimed to develop a real-time **Human-Machine Interface (HMI)** using hand gesture recognition as a natural input method, specifically leveraging **computer vision (MediaPipe Hands)** and **artificial neural networks (ANNs)**. The key objective was to create a system capable of detecting and classifying hand signs in real time, offering an alternative

to traditional input methods like keyboards, switches, or touchscreens, which are often less intuitive or impractical in certain environments.
The project successfully demonstrated the following:

- **Real-time Gesture Recognition**: The system detects hand gestures using a webcam and processes them in real time with a prediction latency of **17-22 milliseconds**. It operates at **30 frames per second**, making it responsive and suitable for real-world applications.
- **High Model Accuracy**: The neural network model achieved a **95-96% accuracy** during training and validation phases, showcasing strong performance in classifying hand gestures based on the 21 hand landmarks captured by MediaPipe.
- **Practical Application**: The system supports a small set of commonly used gestures and highlights the potential for gesture-based interfaces in areas such as industrial automation, gaming, and accessibility technologies.
  By using **MediaPipe** to extract meaningful hand landmarks and converting these into feature vectors, the project efficiently sidesteps the need for more complex image processing techniques. This approach allows for fast, accurate predictions while maintaining low computational requirements, making the system accessible for a wide range of users and devices.

6.b) Future Work

While the project demonstrated successful real-time gesture recognition with high accuracy, there are several opportunities for future development to enhance the system's capabilities and improve its overall robustness. Key areas for future work include:

1. **Expanding the Dataset**: The current dataset, while effective, is relatively small and limited to a few basic hand gestures. To improve the system's versatility, it would be beneficial to expand the dataset by:

- Capturing more diverse hand gestures, including those used in sign language or more complex multi-finger gestures.
- Collecting data from a broader range of users with different hand shapes and sizes, as well as varied backgrounds and environments.
- Augmenting the dataset with variations in lighting conditions, hand orientations, and occlusions to increase robustness in more dynamic or uncontrolled environments.

2. **Increasing the Number of Gesture Classes**: The current system supports only a limited set of hand gestures (e.g., open hand, closed fist, thumbs up). By expanding the gesture set, the system could support a wider range of interactions for various applications. This would involve collecting additional labeled data and potentially adjusting the feature extraction process to handle more complex hand shapes.
3. **Improving Model Accuracy with Complex Architectures**:

- **Convolutional Neural Networks (CNNs)**: While the current model uses a basic artificial neural network (ANN), future iterations could incorporate CNNs to process raw images of hands instead of pre-processed hand landmarks. This could improve the system's ability to recognize complex gestures, particularly in challenging environments.

- **Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) Networks**: These architectures could be used to account for temporal dynamics in hand movements, improving the system's accuracy in recognizing continuous gesture sequences rather than single static poses.
- **Transfer Learning**: Pre-trained models, especially those trained on larger gesture recognition datasets, could be fine-tuned to improve performance without the need for extensive new data collection.

4. **Robustness to Environmental Variations**:

- The current system struggles in environments with extreme lighting or occlusions. Future improvements could include integrating **data augmentation** techniques to simulate varying conditions, as well as incorporating **adaptive thresholding** to handle dynamic lighting in real time.
- **Pose Estimation for Occlusions**: By utilizing temporal information across video frames, the system could handle scenarios where part of the hand is temporarily occluded, reducing the impact of missing hand landmarks.

5. **Multi-hand and Multi-gesture Recognition**: Expanding the system to recognize and interpret gestures made with both hands simultaneously would open up more complex and expressive interaction possibilities. This would involve modifying the current model to handle multiple hand landmarks at once and developing a training set that includes dual-hand gestures.
6. **Real-world Testing and Application Development**: Further development could focus on deploying the system in real-world applications, such as in industrial automation or as an assistive technology for individuals with disabilities. This would involve refining the system's integration with different hardware setups and testing its performance in diverse real-world conditions.

7) Acknowledgments

We thank Imran Naushad Wadkar, and Dr. Chokkalingam S. P. Dean – School of Computing for their contributions to this work. Special thanks to faculties of Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technologyfor their assistance for their support during the research and

8) References and Citation

(a) In- Text Citation:

Industry 4.0 (IR4.0): Industry 4.0 represents the digital transformation of industries, integrating technologies such as IoT, AI, robotics, and big data. It drives automation, improves operational efficiency, and creates new business models. Key benefits include enhanced productivity, agility, and sustainability **[1].** Human-Computer Interaction (HCI): HCI focuses on optimizing the interactions between people and computers, enhancing usability and user experience. It plays a vital role in the design of intuitive interfaces for a range of applications including software, hardware, and AI systems **[2]**. Single Shot Detector (SSD): SSD is a type of object detection framework that performs object localization and classification in a single

forward pass of the neural network, making it efficient for real-time applications **[3]**. MobileNetV2 is a lightweight deep learning model optimized for mobile and embedded vision applications. It improves performance by using depth-wise separable convolutions, reducing computation while maintaining accuracy **[4]**. MediaPipe is a framework for building cross-platform ML solutions, particularly known for its efficient real-time hand tracking, face detection, and gesture recognition capabilities **[5]**. Non-Maximum Suppression (NMS) is a post-processing technique used in object detection to eliminate redundant overlapping bounding boxes, ensuring that only the most relevant detections are retained **[6]**. TensorFlow an end-to-end open-source machine learning platform **[7].** Keras is an Python Deep Learning library **[8]**.

(b) References

[1]. Gholami A, Zare S, Safabakhsh R. A comprehensive survey of hand gesture recognition systems: Methods and applications. Artificial Intelligence Review. 2020; 53(2): 1101-1134. https://doi.org/10.1007/s10462-019-09761-x.

[2]. Zhang Z, Zhang L, Liu Y. A novel hand gesture recognition method based on the combination of CNN and LSTM. Sensors. 2022; 22(1): 123. https://doi.org/10.3390/s220100123.

[3]. Yoon S, Kim K. Real-time hand gesture recognition using a convolutional neural network. Expert Systems with Applications. 2020; 148: 113198. https://doi.org/10.1016/j.eswa.2020.113198.

[4]. Oudah, M., Al-Naji, A., & Chahl, J. (2020). Hand Gesture Recognition Based on Computer Vision: A Review of Techniques. Journal of Imaging, 6(8), 73. https://doi.org/10.3390/jimaging6080073 (https://www.mdpi.com/2313-433X/6/8/73)

[5]. Gupta, O., Raviv, D., & Raskar, R. (2020). Real-time hand gesture recognition using multiple deep learning architectures. Signal, Image and Video Processing, 14(3), 617-626. https://doi.org/10.1007/s11760-019-01577-5 (https://link.springer.com/article/10.1007/s11760-023-02626-8)

[6]. Satybaldina, D., & Kalymova, G. (2021). Deep learning based static hand gesture recognition. Indonesian Journal of Electrical Engineering and Computer Science, 21(1), 398-405. https://doi.org/10.11591/ijeecs.v21.i1.pp398-405 (https://link.springer.com/article/10.1007/s11760-023-02626-8)

[7]. Ozcan, T., & Basturk, A. (2020). Transfer learning-based convolutional neural networks with heuristic optimization for hand gesture recognition. Neural Computing and Applications, 32(12), 8955-8970. https://doi.org/10.1007/s00521-019-04516-x (https://link.springer.com/article/10.1007/s11760-023-02626-8)

[8]. Rahman MM, Hossain MS. A survey on hand gesture recognition techniques: Current status and future directions. Journal of Ambient Intelligence and Humanized Computing. 2021; 12(4): 3963-3982. https://doi.org/10.1007/s12652-020-02739-6.