



DEPARTMENT OF MECHANICAL AND AEROSPACE ENGINEERING

PROJECT REPORT for

Introduction to AI Applications in Engineering

Hand Gesture Recognition

By

***Jamal Lharri, Karim Poonawala, Aidan Pereira, Jay Revagar, Sharaf Ahamed
Shareef, Joji Kondo***

Student ID: 2114228, 2126350, 2137097, 2118629, 2104001, 2112147

Module Code: ME3624

Supervisor: Dr QingPing Yang

MAY 2025

1 INTRODUCTION

1.1 Context and Motivation Behind the Project

Hand gesture recognition (HGR) has emerged as one of the most dynamically and rapidly growing fields within artificial intelligence (AI), particularly in the domains of human-computer interaction (HCI) and collaborative robotics. The primary motivation behind this project stems from the increasing demand for more natural, intuitive, and contact-free methods of communication between humans and machines. Unlike traditional input devices such as keyboards, mice, or touchscreens, hand gestures offer a touchless, accessible, and often faster alternative that can improve user experience, safety, and control precision.

Gestures serve as a universal form of **non-verbal communication**. They are often the most direct and instinctive way of conveying information or issuing commands, especially in environments where voice input is impractical, or where hands-free control is needed. For example, in sign language, distinct static and dynamic hand movements are used to form words and phrases that substitute for spoken communication. A gesture recognition system with sufficient accuracy and flexibility could serve as the backbone for an automated sign language translator, providing real-time conversion of gestures into written language and speech. This would drastically improve communication for hearing-impaired users and bridge the accessibility gap in both social and professional settings.

From a technical perspective, the merging of deep learning, computer vision, and sensor technology has been the driving force behind recent advances in gesture recognition. **Convolution neural networks (CNNs)**, **recurrent neural networks (RNNs)**, and attention-based models have achieved remarkable performance in visual recognition tasks, including image classification, object detection, and action recognition. These models have demonstrated the capacity to learn hierarchical features from large datasets, enabling the recognition of subtle hand pose variations and movement patterns.

Building on this foundation, the goal of this project is to develop a proof-of-concept system capable of recognising a predefined set of static hand gestures using a CNN-based approach, with potential for future extension into dynamic gesture sequences. The system leverages transfer learning techniques to fine-tune a pre-trained ResNet-18 architecture for classifying 5 gesture classes: **'Peace'**, **'Thumbs Up'**, **'Okay'**, **'Heart'**, and **'Fist'**. This provides a strong baseline for real-time gesture recognition using static images, while establishing the groundwork for future development involving temporal model and multi-modal integration.

1.2 Applications

Hand gesture recognition has wide-ranging applications across AI, robotics, and interactive computing. Its ability to provide intuitive, non-contact control and communication makes it highly relevant in domains where conventional input methods may be inefficient or inaccessible.

1. Human-computer interaction (HCI)

Gesture-based control offers a natural and efficient alternative to traditional input devices such as keyboards, mice, and touchscreens. Users can issue commands or navigate digital environments through simple hand movements, allowing seamless interaction with virtual interfaces, 3D objects, and immersive systems. This is particularly valuable in settings such as public kiosks, medical environments, and immersive media where touch-free operation enhances hygiene, efficiency, and user experience.

2. Sign language interpretation

Gesture recognition systems can play a transformative role in assistive technology, especially for the hearing-impaired community. By translating sign language gestures into text or spoken language in real-time, these systems have the potential to significantly improve accessibility and communication. Accuracy and contextual understanding are vital in such applications to ensure that translated gestures convey the correct meaning and avoid misinterpretation.

3. Collaborative robotics

In industrial environments, gesture-based control enhances safety and efficiency by enabling contactless interactions between human operators and robots. Simple hand gestures can be used to pause, initiate, or modify tasks – allowing operators to issue commands without stepping away from their workspace or using physical input devices. This is especially important in cleanroom environments, hazardous areas, or high-paced production lines.

4. Gaming and virtual/augmented reality (VR/AR)

VR and AR platforms heavily rely on intuitive and responsive user interfaces to deliver immersive experiences. Gesture recognition enables users to interact with virtual elements through natural movements, improving realism and engagement. Low-latency gesture classification is crucial in these environments to prevent input lag and maintain immersion.

5. Smart home and internet of things (IoT)

Gesture-based interaction is increasingly being explored in smart environments for controlling lighting, appliances, and security systems. In scenarios where touch or voice control is not feasible (such as speech impairment or noisy settings), hand gestures provide an intuitive and silent control method. Gesture recognition can also support hygiene-sensitive applications, offering a contactless interface for home and healthcare settings.

Overall, the common thread across these applications is the potential for gesture recognition systems to enable natural, efficient, and hygienic interaction between users and machines. As a versatile control modality, gestures reduce cognitive load and facilitate seamless user engagement in both assistive and industrial domains.

1.3 Aims and Objectives

The solution will focus on recognising static hand gestures from still images using a convolution neural network. Through this implementation, the project seeks to demonstrate how AI-driven gesture classification systems can enhance interaction across various application domains, particularly in accessibility, robotics, and intelligent user interfaces.

Objectives:

1. **Critical Literature Review**

Conduct a comprehensive review of existing methods for hand gesture recognition, examining both traditional machine learning techniques and state-of-the-art deep learning architectures.

2. **Dataset Identification and Pre-processing**

Source a publicly available hand gesture dataset or compile a custom dataset if required. Apply pre-processing techniques such as resizing, normalisation, augmentation (e.g. rotation, shear, scale), and RGB correction to improve model robustness and generalisation.

3. **Model Design and Implementation**

Fine-tune a CNN with the use of transfer learning, for the classification of a defined set of static hand gestures. Adjust the networks final layers to balance training performance and inference speed.

4. **Experimental Testing and Evaluation**

Assess model performance using quantitative metrics such as classification accuracy, precision, recall, and F_1 score. Conduct validation using a separate dataset split and visualise outcomes through confusion matrices. Evaluate how the system performs under varying input condition such as lighting or orientation.

5. **Results Analysis and Recommendations**

Critically analyse the system's strengths and limitations in recognising static gestures. Provide insight into areas for improvement, such as dataset diversity, model generalisation, or dynamic gesture recognition. Recommend pathways for future enhancements, including real-time video gesture input and multimodal system.

1.4 Brief Overview of the Chosen AI/Robotics Application

This project focuses on the development of an **image-based hand gesture recognition system** by leveraging deep learning models, specifically convolutional neural networks. The aim is to classify static hand gestures with high accuracy, using pre-captured images as input. This approach is well-suited to scenarios that do not require live video inputs – such as accessibility tools for static sign language translation, command-based gesture control in collaborative robotics, or interaction in smart environments where discrete gesture inputs trigger specific responses.

The project adopts a classification model that processes RGB images through a deep CNN pipeline, optimised through data augmentation and transfer learning. By concentrating on static gestures, the system maintains high recognition speed and computational efficiency, making it practical for deployment in environments with limited hardware resources or real-time constraints. Although dynamic gesture recognition involving temporal sequences is an area of future work, this project establishes a strong foundational framework for gesture-based systems. It highlights how even still image input, when processed through intricate artificial neural networks, can serve a wide range of use cases – from industrial automation to accessibility enhancement.

In adopting this approach, the project demonstrates the practical value of applying AI to gesture recognition tasks. It validates how such systems can contribute to improved user experience, increased accessibility, and more intuitive machine control – ultimately offering robust, user-friendly solutions that operate independently of live camera feeds, yet remain expendable to dynamic systems in future iterations.

2 LITERATURE REVIEW

2.1 History and Evolution of Gesture Recognition

During daily interactions, non-verbal communication, such as hand gestures, accounts for 65% of messages, while verbal communication makes up the remaining 35% [1]. Non-verbal communication can include a variety of methods, with head movements, eye contact, facial expressions, and hand gestures being the most common [2]

Hand gestures have been used for millennia, with the latest studies showing that it was likely the first method of communication between early humans [3]. Hand gestures are used by many cultures to compliment verbal communication [4, 5] and studies suggest that communication exclusively using hand gestures would be greatly successful, even today [6, 7]. Gestures have even been regarded as a universal language and could potentially help modern civilisation communicate with indigenous people [8].

Gesture-based controls have been implemented in modern phones [9] to discretely perform tasks such as accepting and declining calls. Automotive manufacturers such as BMW have also implemented gesture-based controls for the infotainment systems in their vehicles to help make them less intrusive while driving – greatly improving safety [10].

Furthermore, hand gestures have evolved into structured communication systems such as **sign language**, enabling accurate interaction for individuals who are deaf and mute. However, many people do not understand sign language, which limits accessibility. A practical solution is the use of AI-driven systems that can recognise and translate hand gestures, thus broadening communication access across different user groups.

Recent advancements in HCI have highlighted the need for precise and accurate gesture recognition systems [11]. These advancements could ultimately reduce HCI

device dependence on traditional input devices like keyboards and mice by allowing gesture-based interfaces to take their place. This is particularly valuable in immersive environments such as virtual and augmented reality [12].

2.2 Static vs. Temporal (Dynamic) Gesture Recognition

Gestures can be broadly classified into two main types: **static** and **dynamic**. Static gestures refer to poses and shapes that involve no motion, captured in a single frame. These are defined by the orientation and position of the palm and finger [13]. A common example of static gestures is the use of fingers to represent numbers, as shown in *Figure 1*.

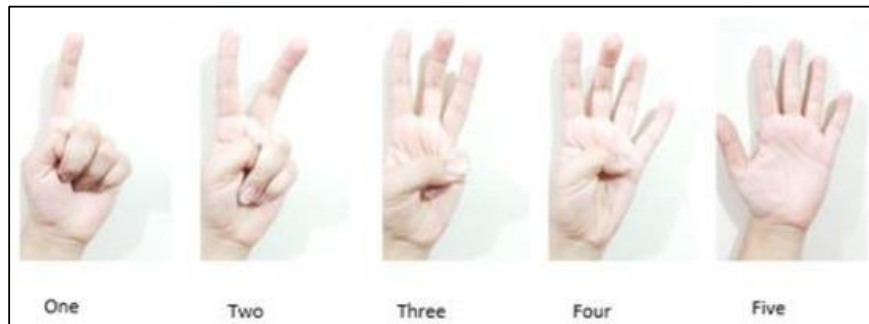


Figure 1: Static hand gestures [14]

In contrast, dynamic gestures are characterised by continuous motion, such as the movement of hands or fingers over time [13]. These gestures are more complex, requiring the model to analyse sequential data. One example includes motorist hand signals, often used by cyclists or horse riders to indicate changes in direction or speed, as illustrated in *Figure 2*.

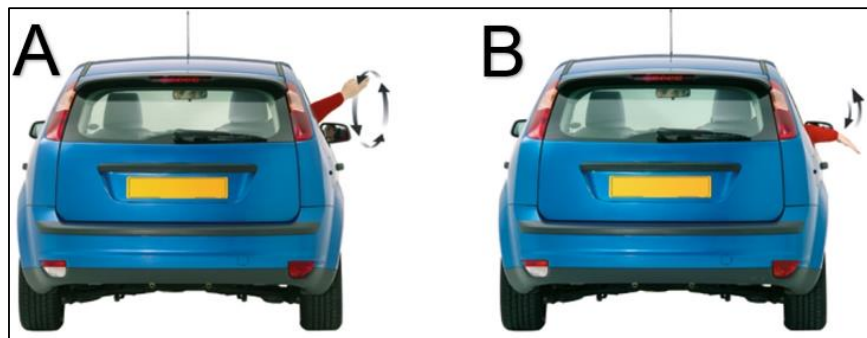


Figure 2: Dynamic hand gestures [15]

Understanding the distinction between these types is critical when designing machine learning models for hand gesture recognition, as different data structures and processing methods are required for static (image-based) versus dynamic (sequence-based) inputs.

2.3 Machine Learning and Deep Learning Techniques

Machine learning (ML) and **deep learning (DL)** algorithms are widely used for gesture recognition tasks. Both fall under the broader field of artificial intelligence, with DL considered a specialised subset of ML that leverages **Artificial Neural Networks (ANNs)** to replicate the learning capability of the human brain [16].

ML techniques enable machines to extract meaningful patterns from large datasets using algorithms. These algorithms process data to generate inferences, decisions, or classifications [16]. DL builds upon this foundation, using interconnected neural structures that allow the model to learn non-linear relationships through layered processing, much like a human brain.

DL typically requires significantly larger datasets to be effective, but offers improved performance for complex tasks such as image and video recognition. There are two main types of ML training approaches: **supervised learning**, which uses labelled datasets, and **unsupervised learning**, where models cluster or group unlabelled data based on similarity. Supervised learning also allows performance to be monitored over time using labelled inputs [17, 18]. These algorithms are combined with computing units known as **neurons** and to form ANNs. Nonlinearity allows for the model to function and learn like a brain, improving its results through repetition without the need for human intervention [19]. A side effect of DL is that it must be trained using very large datasets.

ML models can also be classified into 2 main types: **classification** and **regression**. Classification uses algorithms to accurately arrange test data into learned categories and supports vector trees, decision trees and random forests [17, 18]. Regression models use algorithms to determine the relation between dependent and independent variables, allowing it to predict values for different data points. Classification models are used for image-based data while regression models are typically used for numerical data.

Common classification algorithms are based on ANNs with recurrent neural networks and convolutional neural networks being the most common [20]. CNNs are feedforward neural networks that utilise filters and pooling layers. They have fixed input and output sizes and are better suited for spatial data such as images, where they can predict a label for each image along with a confidence level for the prediction.

RNNs on the other hand can have inputs and outputs of varying sizes and are able to feed results back into the network. This makes them better suited for temporal and sequential data, such as videos and text [20].

2.4 Gesture Recognition Datasets

ML models must be trained using adequately large datasets to ensure accuracy [21], with a larger dataset providing better results. This process of supervised learning can be made significantly quicker by dividing large datasets into smaller subsets that can be processed in parallel.

Quality of data also plays a significant role in the performance of ML models, as noisy or inconsistent data can make the model unreliable and lead to inaccurate predictions [21]. Unfortunately, large, high-quality datasets, especially including those of hand gestures, come with several financial and ethical issues as they would require the personal data of several thousands of people to sufficiently train a ML model. This data would also need to be securely stored in line with laws and regulations, such as GDPR (General Data Protection Regulation).

Robust hand gesture recognition systems must be trained on comprehensive datasets that include multiple gesture types and variations, including different sign languages such as American Sign Language (ASL). To address dataset limitations, **data augmentation techniques** (such as rotation, flipping and scaling) are often employed to artificially increase dataset diversity [22]. These techniques also improve real-world robustness by simulating different camera angles and orientations.

Another critical issue is class imbalance, where certain gesture classes are underrepresented. This can skew model predictions and lead to poor overall performance [23]. Addressing class imbalance using techniques such as weighted loss functions or oversampling is essential to ensure model fairness and reliability.

2.5 Machine Learning Using MATLAB and Limitations

MATLAB has many tools and commands that allow for the creation of AI models, and ML can be implemented to perform hand gesture recognition. It contains commands such as `ImageDatastore` and `splitEachLabel` to help split and organise data. Moreover, MATLAB contains several pre-trained deep learning networks such as **AlexNet** [24] and **ResNet-18** [25], both of which are based on a CNN model. AlexNet contains 8 layers and is pre-trained on over 1 million images, thus transfer learning can be used. The last few layers containing data about the classifier can be changed. This allows the model to retain knowledge about features such as pattern recognition and edge detection while modifying it to specialise in hand gesture recognition [26].

ResNet-18, contains 18 layers and is also pre-trained on over 1 million images. ResNet-18 can classify over a thousand object categories and has an image input size of 224×224 pixels [25]. The `imagePretrainedNetwork` command can be used to implement ResNet-18, while also allowing users to adapt and fine tune the neural network, allowing for transfer learning [27]. The additional layers of ResNet-18 means it runs slower than AlexNet but is better able to differentiate between subtle visible cues in an image. This makes it ideal for more complex recognition where other models such as AlexNet could struggle or even fail to correctly differentiate between fingers and their positions [28].

Additionally, ResNet architectures introduce **residual blocks** to mitigate the vanishing gradient problem commonly encountered in deeper neural networks. As networks grow in depth, gradient signals tend to diminish, making it increasingly difficult for the model to learn. Residual connections address this issue by allowing the network to bypass one or more layers using 'skip' connections, thereby preserving information flow and

enabling gradients to propagate more effectively during backpropagation. This architecture allows the model to focus on learning only the residual (or difference) between layers, which simplifies training and enhances model stability. As a result, ResNet-based models achieve improved generalisation and accuracy, particularly in complex image recognition tasks such as for hand gestures [29].

While deeper ResNet models (ResNet-50 with 50 layers) are available, they require significantly more computational resources. These deeper architectures are generally better suited for high-complexity tasks like facial recognition and may be unnecessarily resource-intensive for more constrained applications like hand gesture recognition [29].

To maximise performance, MATLAB allows for custom training configurations to fully utilise available computational power. For example, enabling GPU acceleration with a setting like `executionEnvironment auto` allows the system to use the GPU (if available), significantly speeding up training. In cases where a GPU is not present, MATLAB defaults to CPU processing, ensuring the program remains functional on a wide range of systems without causing compatibility issues.

MATLAB also streamlines the model evaluation process through built-in tools such as `classify`, `confusionchart`, and `plotroc`. Combined with an extensive library of documentation and support from The MathWorks, Inc., these features make MATLAB a convenient platform for developing machine learning-based hand gesture recognition systems.

However, a key limitation in achieving high prediction accuracy is the requirement for a labelled dataset. These datasets demand substantial storage and computational power for training, which can become a bottleneck in resource-limited environments. Consequently, the performance and scalability of such models are often constrained by the availability of data and hardware.

2.6 Critical Review

Throughout this project, relevant literature and recent research have been drawn upon to guide design decisions and justify the selected methodology. The reviewed studies highlight the increasing importance of hand gesture recognition systems, particularly in enhancing human-computer interaction and improving accessibility for individuals with hearing or speech impairments.

For this investigation, a deep CNN model was chosen, incorporating a residual architecture based on its strong performance in similar tasks. While some studies opt for models like AlexNet due to their simplicity and faster training times, ResNet-18 was also selected here for its greater depth and superior ability to extract complex spatial features. As noted in the literature, deeper CNN models tend to improve recognition precision – especially when distinguishing between subtle gesture variations, a common challenge in hand gesture classification.

In addition, data augmentation was employed as a core strategy to address dataset limitations. By increasing the diversity of training inputs, this technique improves the

model's generalisability while reducing the likelihood of **overfitting**. This aligns with best practices found in previous studies, which stress the importance of augmentation training data to simulate real world conditions.

Another frequently cited issue in gesture recognition literature is class imbalance, where unequal representation of gesture classes can bias the model. This study incorporated class balancing strategies to mitigate this challenge, in accordance with recommendations from existing research. Addressing class imbalance improves fairness and enhances the model's overall predictive performance.

Furthermore, a **multi-metric evaluation approach** was adopted to assess model performance. Rather than relying on a single accuracy metric, multiple indicators (such as precision, recall, and confusion matrices), were used to provide a more comprehensive understanding of the model's strengths and limitations. This approach reflects established evaluation practices in deep learning research.

In summary, the model designed in this project was grounded in proven strategies and informed by previously written academic literature. The combination of a deeper CNN model, data augmentation, class balancing, and robust performance evaluation aims to optimise the system's effectiveness in recognising static hand gestures and ensure its applicability to real-world scenarios.

3 MODEL DEVELOPMENT AND TRAINING (METHODOLOGY)

This section describes the complete technical pipeline for developing and evaluating a deep learning-based hand gesture recognition system. Two pre-trained CNN networks (ResNet-18 and AlexNet) were explored to assess their performance on a custom image dataset. In line with the project brief, the methodology includes dataset preparation, data augmentation, CNN architecture modification, training configuration, evaluation using performance metrics, and deployment through a real-time classification tool. While the broader goal includes recognition of both static and dynamic gestures, this implementation focuses on static, image-based gestures as a foundational baseline for future extension.

3.1 Dataset Identification and Preparation

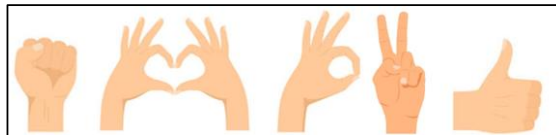


Figure 3: Five hand gestures evaluated in the model

A custom organised dataset HandGestures was constructed, containing **5,000 RGB images** sourced from a GitHub repository [30], equally distributed across five gesture classes: **'Thumbs Up'**, **'Heart'**, **'Peace'**, **'Okay'**, and **'Fist'** (shown above in *Figure 3*). Images were labelled automatically based on folder names using MATLAB's `imageDatastore` function (shown in *Figure 4*).

```
% assign labels based on folder names
imds = imageDatastore('HandGestures', ...
    'IncludeSubfolders', true, ...
    'LabelSource', 'foldernames');
```

Figure 4: Class categorisation using *imageDatastore*

Visual inspection was performed by displaying 20 randomly selected samples in a 4×5 grid to verify class diversity, orientation, and image quality.

3.2 Dataset Partitioning

To ensure fair and stratified learning, the full dataset was split into **80% training** and **20% validation** while maintaining class balance using the function `splitEachLabel`.

```
% Split dataset into 80% training and 20% validation
[trainImgs, valImgs] = splitEachLabel(imds, 0.8, 'randomized');
```

Figure 5: Code for dataset partitioning using *splitEachLabel*

This approach was used for both the ResNet-18 and AlexNet pipelines. An overview of the dataset structure is presented in *Table 1*.

Attribute	Description
Total images	5,000
Image resolution	640 x 480 pixel (RGB)
Number of classes	5
Class names	Thumb_up, Heart, Peace, Okay, Fist
Images per class	1,000
Dataset split	80% Training, 20% Validation
Labelling method	Folder name-based (automatic)

Table 1: Overview of dataset structure

3.3 Data Augmentation and Preprocessing

To enhance model strength, improve generalisation, and prevent overfitting on unseen data (especially under small sample sizes), strong real-time data augmentation was applied using `ImageDataAugmenter`. This simulated various spatial variations such as rotation, translation, scaling, and flipping.

```

% Define strong data augmentation settings
augmenter = imageDataAugmenter( ...
    'RandRotation', [-30 30], ...      % Random rotation between -30 and 30 degrees
    'RandXTranslation', [-15 15], ...  % Random horizontal shift up to ±15 pixels
    'RandYTranslation', [-15 15], ...  % Random vertical shift up to ±15 pixels
    'RandXScale', [0.7 1.3], ...      % Random scaling along X-axis (70% to 130%)
    'RandYScale', [0.7 1.3], ...      % Random scaling along Y-axis (70% to 130%)
    'RandXShear', [-20 20], ...       % Random horizontal shearing between -20° and 20°
    'RandYShear', [-20 20], ...       % Random vertical shearing between -20° and 20°
    'RandXReflection', true);         % Random horizontal flipping

```

Figure 6: Code for data augmentation

The validation set was not augmented, and all images were resized to 224×224×3 and 227×227×3 pixels, matching the input sizes of ResNet-18 and AlexNet respectively.

3.4 CNN Selection and Transfer Learning Setup

Transfer learning was used to train pre-trained networks for fine-tuning five-class classification. Final layers (fc1000, prob, ClassificationLayer_predictions) of the original ResNet-18 were overridden with a classification layer, SoftMax, and a fully connected layer – connected after the pool5 layer. The first 50 layers were frozen to keep low-level features intact.

For comparison, the same strategy was applied to AlexNet – the final layers (fc8, prob, output) were also substituted, with additional new layers inserted after the drop7 layers, with the first 10 layers being frozen. Input sizes adjusted accordingly (ResNet-18: 224×224, AlexNet: 227×227)

3.5 Training Configuration

Both the models were trained using the Adam optimizer, and an initial learning rate of 5E-5 for 20 epochs and mini-batch size of 64. The learning rate was reduced by 0.5 every 5 epochs, and L2 regularization (1E-4) was applied. Early stopping was enabled with a validation patience of 4. Training was performed on either CPU or GPU (auto setting), and the progress was monitored using MATLAB's training progress tool.

Table 2: Key hyperparameters used in training the hand gesture recognition model, along with their values and justification

Hyperparameter	Value	Purpose
Optimiser	Adam	Adaptive learning rate per parameter
Initial Learn Rate	5.00E-05	Slow learning to fine tune pretrained layers
Max Epochs	20	Balanced to prevent overfitting
Mini-batch Size	64	Reasonable trade-off between performance & speed
Learning Rate Drop	0.5 every 5	Piecewise schedule for stable convergence
Validation Patience	4	Early stopping to avoid overfitting

3.6 Performance Evaluation

Once training was complete, the model's ability to generalise unseen data was assessed using the validation set. MATLAB's `classify` function was used to generate classification results and compare with ground truth labels.

An in-depth performance analysis and detailed breakdown of the evaluation strategy was conducted – including a validation setup, performance metrics, confusion matrix analysis, and additional insights such as the F_1 score. Results are presented in the *Model Validation and Evaluation Metrics* section of the report.

The final model after performance validation was then serialised for re-use and integration into the real-time application tool.

3.7 Model Saving and Export

The trained model was serialised using the code shown in *Figure 7*.

```
% Save the trained ResNet18 model to a .mat file for future use
save('trainedHandGestureModel_ResNet_11.mat', 'trainedNet');
```

Figure 7: Code used to save model as a .mat file

The ResNet-18 model was saved as 'trainedHandGestureModel_ResNet_11.mat', and the AlexNet model was saved as 'trainedHandGestureModel_AlexNet.mat'. These models were then later uploaded into a separate script to perform real-time classification on unseen user uploaded images.

3.8 Real-Time Inference and User Interaction Tool

As part of demonstrating the application, a **real-time prediction tool** was implemented. This script (shown in *Figure 8*) opens the trained model, provides upload functionality for a single or group of gesture classes, and labels each image resized and enhanced.

Classification is performed using a **test-time augmentation (TTA)** approach, whereby original and horizontally flipped versions of each image are input into the model and their prediction scores are averaged to improve robustness. The three top classes and confidence levels are listed alongside each image, colour-coded by confidence:

- Green (>90%)
- Orange (70–90%)
- Red (<70%)

```

% ---- Load and preprocess image ----
imgPath = fullfile(path, files{i});
img = imread(imgPath);
img = imresize(img, inputSize(1:2));

% ---- Predict gesture using ResNet18 (original + flipped)
% Classification using Test-Time Augmentation (TTA)
tic;
[~, score1] = classify(trainedNet, img);
[~, score2] = classify(trainedNet, fliplr(img));
elapsedTime = toc;
finalScore = (score1 + score2) / 2;

```

Figure 8: Code for real-time inference script

The console output also prints processing time and predictions. The tool is applied to both the ResNet-18 and AlexNet models, with automatic input size adjustment.

4 MODEL VALIDATION AND EVALUATION METRICS

The section outlines the evaluation strategy used to assess the performance of the trained hand gesture recognition model. Given the multi-class nature of the problem, a combination of evaluation metrics was employed to offer a comprehensive performance analysis across all hand gesture categories. These include class-wise indicators such as accuracy, precision, recall, F₁-score, a confusion matrix and a Receiver Operating Characteristic (ROC) curve analysis.

4.1 Validation Strategy

Intermediate validation checks were conducted during training using MATLAB's `trainingOptions`, by specifying the `ValidationData` parameter - allowing constant monitoring of validation loss and accuracy. This could potentially help identify early signs of overfitting during the training process.

An ideal training process is characterised by a steady increase in validation accuracy and minimal divergence between training and validation performance. A significant early divergence, on the other hand, may indicate overfitting or insufficient data variability within the dataset.

4.2 Accuracy

Accuracy (*Equation 1*) is the simplest and most used performance metric. It represents the ratio of correctly classified samples to the total number of predictions made on the validation set.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \quad (\text{Equation 1})$$

In this project, accuracy was computed using the predicted class labels from the model (`classify`) and compared against the ground truth labels from the validation dataset.

Although accuracy is a useful general metric, it can be misleading in multi-class or imbalanced datasets. However, in this case, the dataset was evenly balanced with 1,000 samples per class, making accuracy a valid performance indicator.

4.3 Confusion Matrix

For a deeper insight into class-wise performance, a normalised confusion matrix was generated. Confusion matrices provide a visual summary of a classifier's predictions against its actual labels in a graphical performance, helping identify recurring misclassifications between gesture types.

In a confusion matrix:

- **Rows** represent actual classes
- **Columns** represent predicted classes
- **Diagonal elements** show correctly classified samples
- **Off-diagonal elements** indicate misclassifications

Therefore, this layout allows for the identification of specific gesture pairs that the model struggles to distinguish, such as visually similar signs like '**Peace**' and '**Okay**'.

The confusion matrix was generated using MATLAB's `confusionchart` function, with normalisation applied to enable comparison across classes regardless of sample size. High diagonal values and minimal off-diagonal entries indicate strong model performance. Conversely, frequent misclassification between certain class pairs may suggest the need for further data augmentation or additional training samples.

4.4 Precision, Recall, and F₁ Score

To complement accuracy and address class-specific performance, three additional evaluation metrics were employed:

- **Precision** (*Equation 2*): Measures the proportion of correctly predicted samples for a specific class out of all samples predicted for that class

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad (Equation\ 2)$$

- **Recall** (*Equation 3*): Measures the proportion of actual class samples that were correctly identified

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \quad (Equation\ 3)$$

- **F₁-Score** (*Equation 4*): The harmonic mean of precision and recall, providing a balanced metric that accounts for both false positives and false negatives

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (Equation\ 4)$$

These metrics were computed using MATLAB's `confusionmat` function and scaled to per-class values. Scores above 0.80 for precision, recall, and F₁ scores indicate good

model performance [31]. Imbalances between precision and recall may highlight over or under-prediction for certain gestures. Consistent values across all classes suggest stable and reliable model behaviour.

4.5 ROC Curve and AUC

To further assess the model's discriminative power, ROC curves were generated for selected gesture classes. This involved reframing the multi-class problem as **multiple binary classification tasks**, where each gesture class was treated independently against all others. The ROC plots illustrate the trade-off between the true positive rate (sensitivity) against the false positive rate ($1 - \text{specificity}$) across a range of classification thresholds. These plots provide insight not only into the model's overall performance but also into how confidently and consistently it can classify each specific gesture.

A key metric derived from the ROC curve is the area under the curve (AUC), which serves as a quantitative indicator of class separability. AUC values range from 0 to 1, with values closer to 1.0 indicating superior discrimination and values around 0.5 suggesting performance no better than random guessing. Typically, AUC values above 0.9 are considered excellent, reflecting a strong ability to distinguish between the selected gesture and all others [31].

Although ROC analysis is traditionally applied to binary classifiers, it was adapted here to assess subtle class-specific distinctions in gestures that often share similar visual features. For example, gesture such as 'Heart', 'Okay', and 'Thumbs Up' may exhibit overlapping hand positions or orientations, which can lead to misclassification. In such cases, ROC curves help identify these challenges more effectively than overall accuracy or confusion matrices alone.

These insights are particularly relevant for real-world applications, where misclassification of similar gestures could hinder usability or user experience. Lower AUC values highlight areas where the model may benefit from further refinement, whether through targeted data augmentation, architectural improvements, or the incorporation of additional discriminative features.

5 EXPERIMENTS AND RESULTS

This section presents the quantitative outcomes of the trained hand gesture classification models. As mentioned previously, performance for both models is measured in terms of key performance metrics such as overall validation accuracy, class-wise metrics, confusion matrix analysis, and ROC curves.

5.1 Overall Performance

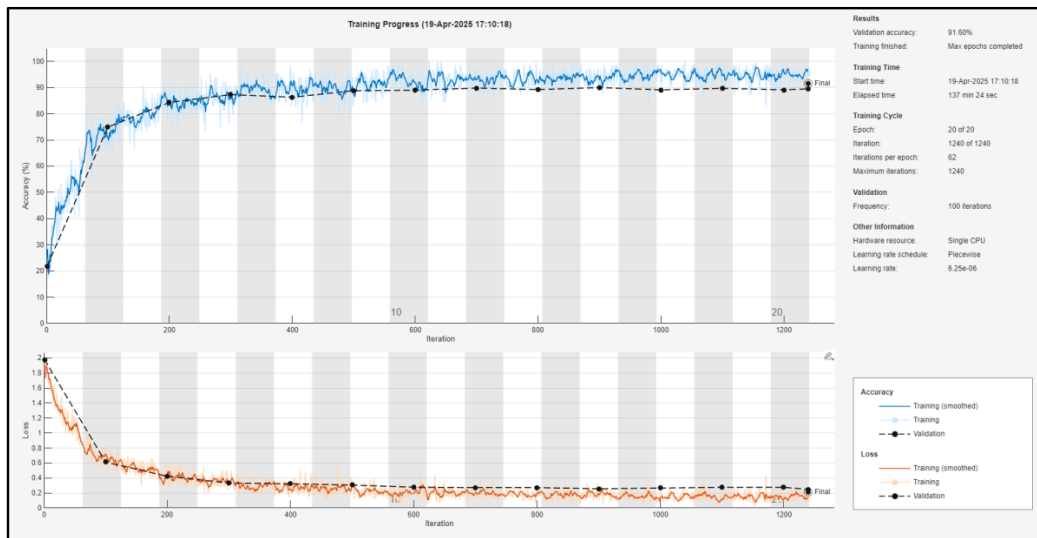


Figure 9: Training Accuracy Curve ResNet-18 (Training Validation Accuracy – 91.60%)

For the ResNet-18 model seen in Figure 9, training accuracy increased rapidly, exceeding 90% within 20 epochs, with validation accuracy reaching 91.60%. Training loss decreased steadily to below 0.3, and convergence was recorded around the 15th epoch. The relatively small gap between training and validation accuracy/loss for ResNet-18 suggests strong generalisation with minimal signs of overfitting.

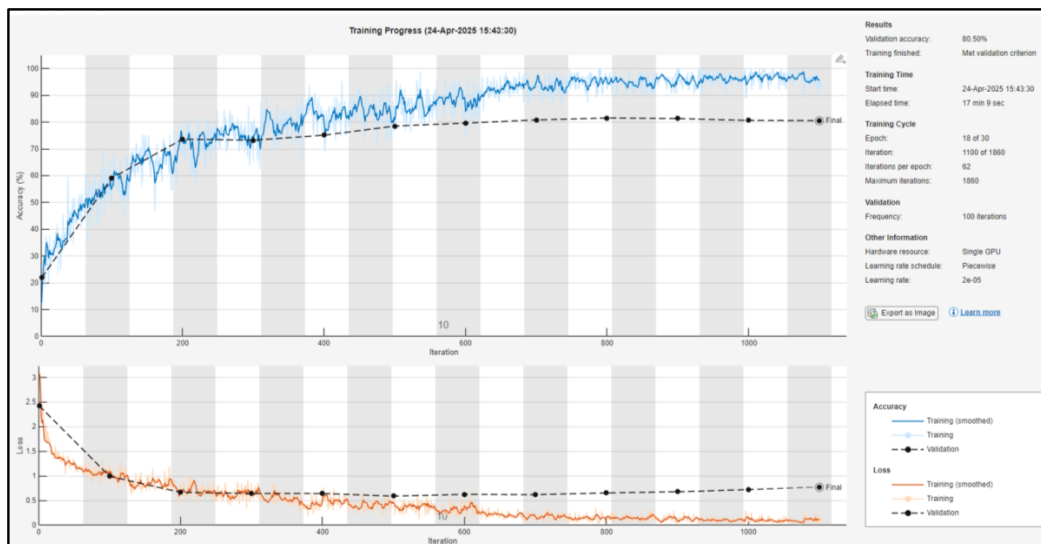


Figure 10: Training Accuracy Curve AlexNet (Training Validation Accuracy – 80.50%)

For the AlexNet model seen in Figure 10, the training accuracy rose more gradually, stabilising also around the 15th to 16th epoch, with a final validation accuracy of 80.50%. Training loss decreased to approximately 0.6, but validation showed greater

fluctuations as opposed to the ResNet-18 model. This fluctuation could be indicative of overfitting or limited generalisation capacity compared to ResNet-18.

5.2 Validation Accuracy and Confusion Matrix Results

Once the training phase was completed, the model was assessed using a separate validation dataset to test the performance of the model on unseen data. This resulted in an overall validation accuracy of 92.50%, again calculated through the `classify` function.

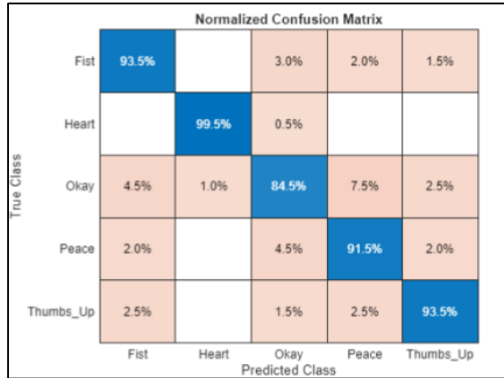


Figure 11: Confusion matrix for ResNet-18

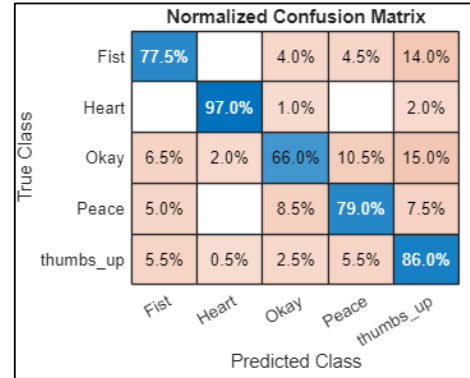


Figure 12: Confusion matrix for AlexNet

Table 3: Correct classifications from confusion matrices for ResNet-18 and AlexNet

Hand Gesture	ResNet18	AlexNet
Fist	93.50%	77.50%
Heart	99.50%	97.00%
Okay	84.50%	66.00%
Peace	91.50%	79.00%
Thumbs Up	93.50%	86.00%

Figures 11 and 12 present the full normalised confusion matrix for both the ResNet-18 and AlexNet models. The correct classification rates observed along the diagonal have been listed in Table 3 above. It is apparent from the results that the ResNet-18 model has more correct classifications than the AlexNet model. For example, the 'Okay' hand gesture was correctly classified for 66.0% of all cases using the AlexNet model in comparison to 84.5% of all cases for the ResNet model. On the other hand, the 'Heart' hand gesture was correctly classified above 95% of the time for both pre-trained models.

5.2.1 Misclassifications

The ResNet-18 confusion matrix shown in Figure 11 demonstrates strong overall performance, with most gestures classified correctly. However, notable misclassifications occurred between 'Okay' and 'Peace', with 7.5% of 'Okay' gestures

incorrectly predicted as '**Peace**'. Minor confusion was also observed between '**Fist**' and '**Heart**', though at a much lower rate.

In contrast, the AlexNet confusion matrix in *Figure 12* reveals a higher rate of misclassification, particularly for the '**Okay**' gesture. Specifically, 15% of '**Okay**' gestures were misclassified as '**Thumbs Up**', and 10% were incorrectly labelled as '**Peace**', indicating that the AlexNet model struggles more with gesture separability, especially when visual cues are ambiguous or overlapping. Further analysis suggests that the high visual similarity in hand orientation, finger curvature, and gesture outline (particularly between '**Okay**', '**Peace**', and '**Thumbs Up**') is a key contributor to these errors. These challenges are likely exacerbated under conditions of inconsistent lighting, partial occlusion, or variation in hand scale and rotation across samples. These results highlight the importance of training models on more diverse and augmented datasets to improve robustness to such variations.

5.3 Class-Wise Performance Metrics Results

A more detailed analysis of classification performance was conducted using **precision**, **recall**, and **F1-score** metrics for each individual hand gesture class. These metrics provide a deeper understanding of the model's ability to correctly identify each gesture type beyond overall accuracy. The results are summarised in *Tables 4* and *5* for ResNet-18 and AlexNet, respectively.

Table 4: Class-wise performance metrics for ResNet-18

Gesture Class	Precision	Recall	F1-Score
Fist	0.935	0.912	0.923
Heart	0.995	0.99	0.992
Okay	0.845	0.899	0.871
Peace	0.915	0.884	0.899
Thumbs Up	0.935	0.94	0.937

As shown in *Table 4*, the ResNet-18 model achieved consistently high precision, recall, and F1-scores across all gesture classes. The '**Heart**' gesture attained the best overall performance, with all three metrics nearing or exceeding 0.99. While the '**Okay**' gesture remained the most challenging to classify, its F1-score of 0.892 still reflects relatively strong performance. Compared to AlexNet, ResNet-18 demonstrated significantly better class-wise results across the board, particularly in handling ambiguous or similar gestures.

Table 5: Class-wise performance metrics for AlexNet

Gesture Class	Precision	Recall	F1-Score
Fist	0.775	0.82	0.797
Heart	0.97	0.975	0.972
Okay	0.66	0.805	0.725
Peace	0.79	0.794	0.792
Thumbs Up	0.86	0.691	0.766

In contrast, *Table 5* reveals that the AlexNet model exhibited lower class-wise performance overall. The '**Okay**' gesture once again had the lowest scores, with a notably reduced F₁-score of 0.725. Additionally, the '**Thumbs Up**' gesture showed the weakest recall (0.691), indicating a higher frequency of false negatives for that class. These lower metrics suggest that AlexNet struggled more with consistently recognising certain gestures, especially those with visual similarity or greater intra-class variability.

Overall, the ResNet-18 model demonstrated stronger and more balanced class-wise classification, highlighting its superiority in both precision and robustness across a range of gesture types. These findings are consistent with earlier observations from the confusion matrices and reinforce ResNet-18's reliability in distinguishing between subtle hand configurations.

5.4 ROC Curve and AUC Evaluation

To further test and quantify the classification performance of both models, Receiver Operating Characteristic curves were generated for each gesture class using a one versus rest approach. The plot is of the true positive rate against the false positive rate at varying classification thresholds.

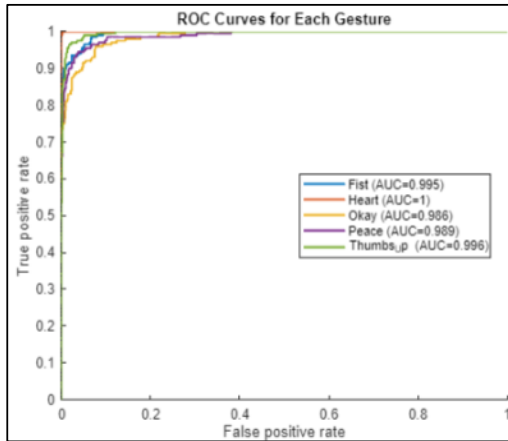


Figure 13: ROC curves for ResNet-18

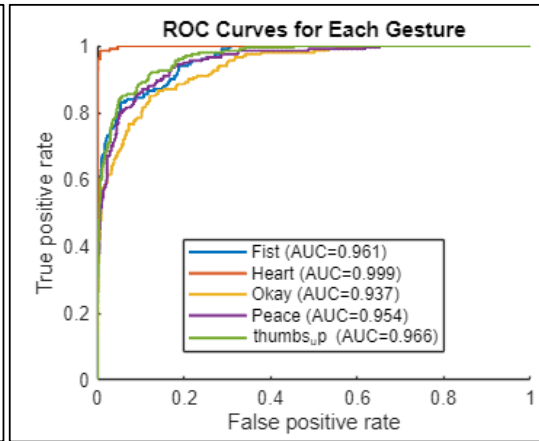


Figure 14: ROC curves for AlexNet

The area under the curve values, which represent a scalar measure of the model's ability to differentiate one class from the other, were extracted for each ROC curve for both models. *Figure 13* and *14* above each display all five ROC curves for the ResNet-18 and AlexNet models respectively, each curve representing one gesture class. It is

evident that for both cases AUC values are consistently above 0.9, however values for the ResNet-18 model appear slightly greater.

5.5 Real-Time Inference and Image Prediction Results

This section of the report will provide the results obtained from original images uploaded of a variety of the five hand gesture classes. This will assess both the AlexNet and ResNet-18 models against unseen and untrained data to see whether the model can correctly categorise the hand gestures. In total three images and their predictions for each respective model are displayed below.

ResNet18		AlexNet
Top Predictions: 1. Heart — 99.9% 2. Okay — 0.1% 3. Peace — 0.0%		Top Predictions: 1. Heart — 100.0% 2. Okay — 0.0% 3. Fist — 0.0%

Figure 15: Image and model predictions for 'Heart' gesture


ResNet18		AlexNet
Top Predictions: 1. Okay — 57.0% 2. Thumbs _u p — 42.6% 3. Peace — 0.3%		Top Predictions: 1. thumbs_up — 88.9% 2. Fist — 10.2% 3. Okay — 0.9%

Figure 16: Image and model predictions for 'Thumbs Up' gesture


ResNet18		AlexNet
Top Predictions: 1. Heart — 63.0% 2. Thumbs _u p — 14.2% 3. Okay — 11.5%		Top Predictions: 1. Heart — 51.7% 2. Okay — 42.7% 3. Peace — 4.1%

Figure 17: Image and model predictions for 'Okay' gesture

The results above showcase three different unseen hand gestures fed into image prediction models each trained by ResNet-18 and AlexNet. The results show that the images displaying the 'Heart' gesture were correctly classified, with both models predicting above 99%. On the other hand, both models struggled to guess the image

displaying an '**Okay**' gesture, with the ResNet-18 model predicting only a 11.5% chance of the gesture being the '**Okay**' sign.

6 DISCUSSION

6.1 ResNet-18 vs. AlexNet

To compare the use of ResNet-18 with AlexNet for transfer learning, the model was built using both CNNs. As shown in the performance graph of the AlexNet model (*Figure 10*), the **model accuracy converges later** than ResNet at around 800 iterations. The validation accuracy for AlexNet is 80.5%, much lower than ResNet at 91.6%. Furthermore, using AlexNet, the validation accuracy lags around 15% lower than the training accuracy, suggesting overfitting of data.

6.1.1 Class-Wise Performance

As shown in *Figures 11* and *12*, both models displayed the highest accuracy when classifying the '**Heart**' gesture, with the ResNet-18 model featuring a validation accuracy of 99.5%, higher than the 97.0% accuracy of the AlexNet model. The ResNet-18 model misclassified gestures such as '**Okay**' at a rate of 0.5%, while the AlexNet model reproduced this at a higher rate of 1.0%, additionally misclassifying the gestures as a '**Thumbs Up**' 2.0% of the time. Rates of correct classification for the '**Fist**', '**Okay**', and '**Peace**' gestures were **significantly higher** with the ResNet-18 model than with the AlexNet model, with each being greater by more than 10%. Similarly, for both models the lowest accuracy was demonstrated for the '**Okay**' classification. For the ResNet-18 model this was 84.5%, with the AlexNet model achieving even lower at 66.0%, misclassifying over a third of samples.

Modes of misclassification were **mostly consistent** between models, except for the AlexNet model misclassifying the '**Heart**' as a '**Thumbs Up**' (as mentioned previously), and the '**Thumbs Up**' as a '**Heart**' 0.5% of the time. The highest rate of misclassification presented by the ResNet-18 model was 7.5%, where it misclassified '**Okay**' as '**Peace**', while for the AlexNet model, it was 14.0% and was due to the misclassification of '**Fist**' as '**Thumbs Up**'. For this model, three modes of misclassification exceeded rates of 10%, including the mode most prevalent in the ResNet-18 model. Other areas of discrepancy include the AlexNet model featuring noticeably higher rates for misclassifications as a '**Thumbs Up**', incorrectly predicting inputs (especially a '**Fist**') as a '**Thumbs Up**' to a higher degree. Otherwise, each mode of misclassification exhibited by the ResNet-18 model was **similarly represented** by the AlexNet model, albeit at a higher rate.

When comparing other performance metrics (as shown in *Tables 4* and *5*), the ResNet18 model featured a higher value for precision, recall and F₁-score than the AlexNet model. For both models, the highest value for recall was for the '**Heart**' gesture, with 0.975 for the AlexNet model and a higher value of 0.990 for the ResNet-18 model. The lowest value however differed between models, with the ResNet-18 one featuring a value of 0.884 for the '**Peace**' classification, while the AlexNet one featured a **drastically lower** value of 0.691, for the '**Thumbs Up**'. The AlexNet model therefore

showed a much wider range of values for both precision and recall. The F₁-scores for the AlexNet model were lower and of a wider range than those for the ResNet18 model, however both featured the highest values for the '**Heart**' classification at 0.990 and 0.972, for ResNet-18 and AlexNet, respectively.

As shown in *Figure 13*, ROC curves generated for each gesture classification again demonstrated higher accuracy for the ResNet-18 model, for all classifications. AUC values similarly were also shown to be higher for the ResNet-18 model, however to a lesser degree. Once more, the '**Heart**' classification showed the highest accuracy for both models, while the '**Okay**' classification featured the lowest.

Throughout the results of this experiment, the ResNet-18 model consistently performed with higher accuracy than the AlexNet model, demonstrating its suitability for the development of this model. While both models conducted similar errors, the ResNet-18 model exhibited these at a **noticeably lower rate**. Such behaviours evidence the deeper architecture of ResNet-18 and supports the implementation of residual learning in an image classification model. As a result, the model developed using ResNet-18 was regarded as the primary model for this experiment.

6.2 Model Accuracy

Employing ResNet-18 as the pre-trained CNN architecture, the model overall presented high performance, achieving high gesture identification accuracy upwards of 80% for all classes. Across gesture classifications, the '**Heart**' sign demonstrated the highest validation accuracy at 99.5%, with only 0.5% of predictions misclassified as an '**Okay**'. This behaviour may be attributed to the visual distinctiveness of the '**Heart**' sign, as it is the only gesture using **two hands** to form a clear, open shape. The '**Okay**' was the only other gesture to feature an open shape, albeit with one hand, which would explain its misclassification. More significantly, the model employed performs image recognition, rather than solely the recognition of the hands. Therefore, it would be expected to recognise and form patterns of the image area beyond just the intended hand gesture, such as the presence and positioning of the arms – making a gesture that uniquely employs both hands and arms (such as the '**Heart**' sign) easily distinguishable. This behaviour highlights how the placement of more than just the hands performing the gesture should be considered in recognition.

The lowest validation accuracy was exhibited for the '**Okay**' gesture, at 84.5%, noticeably lower than the accuracy of other classifications which exceeded 90%. The most significant misclassifications for this gesture were the '**Peace**' and '**Fist**', with confusion rates of 7.5% and 4.5% respectively. The '**Peace**' gesture resembles the '**Okay**' in featuring the hand positioned upwards, with a few fingers outstretched. While the '**Fist**' does not feature fingers outstretched, for all three gestures beyond the hands, the arms would be similarly oriented upwards. Such congruence in shape contributed to these misclassifications. The '**Okay**' was also the only gesture to be misclassified as every other, at a noticeable rate (>0.5%), with a rate of 1.0% for the least prevalent

case, where it was misclassified as a '**Heart**'. Validation accuracy for all other classifications was between 90-95%.

When measuring recall, the highest value was shown by the '**Heart**' classification, of 0.990, while the lowest was shown by the '**Peace**' classification, of 0.884. For F₁-scores, the '**Heart**', again, produced the highest score, of 0.992, however in this case the '**Okay**' classification gave the lowest score of 0.871, following the behaviour of previous metrics. Apart from the '**Okay**' classification, each gesture produced a score close to or above 0.9. As the harmonic mean of both precision and recall values, the F₁-score can be used to symmetrically represent both metrics in one.

Compared to existing hand gesture recognition models, the results from this project demonstrate impressive performance, particularly for visually distinctive gestures. The use of static image inputs processed through a deep CNN framework illustrates a robust and scalable approach for gesture recognition, offering a solid platform for future advancements toward dynamic recognition and more complex interaction systems.

6.2.1 Image Prediction

A real-time prediction tool was implemented and used to evaluate original generated images with both ResNet-18 and AlexNet models. This exercise was beneficial for the purposes of trialling the models for **manual testing**, as they would be employed in an application.

Both models were trialled for three cases, of which the AlexNet model correctly predicted two, and the ResNet-18 model predicted one. For the '**Heart**' classification, both models **strongly** predicted the correct gesture, with the ResNet-18 model giving 99.9% and the AlexNet model giving 100.0%. In this case, both models provided **absolute** predictions, demonstrating the ease in classifying the '**Heart**' gesture as shown in validation. For the '**Thumbs Up**' classification, the AlexNet model correctly identified the gesture with a strong prediction of 88.9%, while the ResNet-18 model incorrectly identified an '**Okay**' gesture with a moderate prediction of 57.0%, giving a prediction of 42.6% for the correct classification. For this gesture, the ResNet-18 model displayed **weaker** performance compared to that shown in validation. Finally, for the '**Okay**' classification, both models incorrectly predicted the gesture as a '**Heart**', with the ResNet-18 model giving a greater prediction of 63.0%, and the AlexNet model giving a prediction of 51.7%. The AlexNet model gave a stronger prediction of 42.7% for the correct gesture compared to the ResNet-18 model, which gave a weak prediction of 11.5%. The difficulty in correctly identifying the '**Okay**' gesture further exemplifies the behaviour demonstrated in the validation stage, where the classification exhibited the **lowest** accuracy amongst classes.

While in the stage of validation the ResNet-18 model outperformed the AlexNet model, in this instance of evaluation, the AlexNet model demonstrated more consistent performance, with predictions leaning more towards the correct classification. This experiment highlighted differences in behaviour in mass testing of the model against individual, discrete applications. A reason for such distinctions could be the gross

generalisation of results derived from mass testing, with individual trialling of discrete data representing real world performance more accurately.

6.3 Challenges Faced

The obstacles encountered during development of this project are listed below:

1. **Large Dataset Requirements**

Achieving high accuracy required a large and diverse dataset, which demanded significant storage space and manual effort to collect, organise, and label images across different conditions.

2. **Computational Limitations**

Training the deep learning model, even with transfer learning, required substantial computational resources. Systems without high-end GPUs faced longer training times, relying instead on slower CPU processing.

3. **Model Overfitting Risks**

With a limited custom dataset (5,000 images), there was a risk of the model overfitting to training data. Data augmentation strategies were crucial.

4. **Gesture Similarities Leading to Misclassification**

Certain hand gestures like '**Peace**' and '**Okay**' had subtle visual differences, making them harder for the model to distinguish, resulting in occasional confusion and lower class-wise precision/recall for these categories.

5. **Fine-Tuning Hyperparameters**

Choosing the right combination of learning rate, batch size, and augmentation methods required iterative testing. Overly aggressive training settings could lead to unstable training behaviour or convergence issues.

6. **Generalisation to Real-World Data**

Despite good validation accuracy, the model's ability to handle new, real-world images comprised of different lighting, backgrounds, and hand sizes was always a potential limitation.

6.4 Relevance to Real-World Applications

Hand gesture recognition systems have significant real-world impact across various sectors. One major application is in **sign language translation**, enabling effective communication between deaf and non-deaf individuals. Additionally, non-contact communication through hand gestures is highly valuable in healthcare settings, where minimising physical contact is important for hygiene and patient safety, or in situations where verbal communication is restricted.

In the field of security and intelligence surveillance, gesture recognition technology can enhance monitoring systems by detecting suspicious or dangerous hand movements. Furthermore, smart home systems benefit from gesture control, allowing users to operate lighting, appliances, and other devices through intuitive, touch-free

commands, promoting a more automated and accessible living environment. Automotive companies such as the BMW group have also implemented gesture recognition in their vehicles to allow drivers to control the car's radio and infotainment systems by simply performing hand gestures, without the need to take their eyes off the road [10].

Overall, hand gesture recognition bridges the gap between humans and machines, improving accessibility, safety, and convenience across a range of industries.

While the achieved accuracies are sufficient for **recreational and amateur applications** such as gaming or casual device control, they may not meet the stringent requirements of **industrial or safety-critical systems**, which typically demand recognition accuracies of 95% or higher such as an automated bartender setup that recognised static hand gestures [32]. In applications where user input has a direct effect on safety or critical operations, the risk of catastrophic failure mandates higher factors of safety and reliability. Consequently, although hand gesture recognition is an effective and efficient tool for convenience-driven systems, its use must be carefully considered and validated when integrated into environments with profound consequences for error.

Application of the model in a real-world scenario was verified by conducting a manual testing phase, authenticating the process throughout, from the user capturing an original image from a digital camera rather than employing gesture images from available datasets, to classification using the developed model.

6.5 Limitations

Despite achieving satisfactory validation accuracy, several limitations were present in this study:

1. Limited Gesture Set

Only five distinct hand gestures were used for training and testing. While this was sufficient for demonstrating the model's core functionality, real-world applications often require a broader range of gestures, which could introduce additional complexity and reduce classification accuracy.

2. Static Image Dataset

The system relied solely on static image data rather than live video input or dynamic gesture sequences. This restricts the model's applicability in real-time control environments where gestures may be performed with motion.

3. Environmental Consistency

Images in the dataset may have been captured under similar lighting conditions and backgrounds, which can lead to overfitting. In real-world settings, variation in lighting, background, and hand orientation could significantly affect performance.

4. Model Generalisability

The use of transfer learning with a pre-trained ResNet-18 model was beneficial for accuracy and training efficiency. However, ResNet-18 was originally designed for object classification and may not optimally capture subtle hand gesture features, limiting its generalisability to more complex or nuanced gestures.

5. **Lack of User Diversity**

The dataset likely involved a limited number of individuals performing gestures. Variations in hand size, shape, and skin tone across different users could influence model accuracy and fairness.

6. **No Temporal Analysis**

Temporal aspects of gesture motion (e.g. speed and order) were not analysed, as the system processed individual frames. This could be a constraint in use cases where gesture dynamics are important.

7 **CONCLUSION**

7.1 **Conclusion to Project**

This investigation successfully explored the structure and application of hand gesture recognition technology, using the **ResNet-18** and **AlexNet** models and five distinct hand gestures to do so. Comparative analysis between the two CNN models highlighted that the ResNet-18 model consistently **outperformed** AlexNet across all tested performance metrics. This includes key metrics such as validation accuracy, precision, recall, and F1 scores. The deeper architecture, higher layer count, and residual learning of the ResNet-18 model, thus contributes to lower misclassification rates, particularly for gestures with subtle differences.

Regarding the hand gesture dataset itself, it was concluded that both models showed very similar patterns in how accurately they determined certain gestures. The **'Heart'** gesture was by far the most accurately recognised with the least being the **'Okay'** sign. Surprisingly, the AlexNet model also had poor recognition of the **'Fist'** gesture, often confusing it with **'Thumbs Up'** – an issue not observed with the ResNet-18 model, highlighting limitations of AlexNet.

Despite showing high performance, challenges such as gesture similarity, dataset limitations, computational constraints, and generalisation to real-world data brought to light the complexity involved with developing and optimising hand recognition technology. Strategies used to perform data augmentation were essential in mitigating risks such as overfitting and allowing the model to be more robust.

Interestingly, manual validation tests performed on both models suggested that for real-time classification, the AlexNet model is more effective. Across all gestures, including those misclassified, the AlexNet model was found to predict the correct gesture more accurately. However, for both models, validation of the **'Okay'** sign was unsuccessful as it resulted in a confident misclassification. This was concluded to be likely due to issues with the validation images used. Nevertheless, the results prove that the models still require further careful optimisation.

The relevance of hand gesture recognition technology was substantiated through their wide-ranging applications, from accessibility and healthcare tools to their use in smart homes and industrial settings. This model, however, was found to only meet the needs of recreational and casual applications as more technical fields require even more accurate and precise gesture detection – applications where misclassification can lead to catastrophic failure.

This investigation demonstrated the effectiveness of static hand gesture recognition systems, with this CNN model in particular acting as a strong foundation for future improvements and optimisation, and possible implementation into environments where precise touchless control is necessary.

7.2 Suggestions for Further Work

A key limitation in the development of this hand gesture recognition system was the narrow classification scope – only five distinct gestures were included in the training and testing phases. While this small set demonstrates the foundational capability of the system, it limits scalability in the real-world applications. For instance, simple gesture-controlled systems (e.g. robotic control on assembly lines) may function with limited gesture types. However, more advanced implementations such as real-world sign language interpretation would require recognition of a far more extensive and nuanced gesture vocabulary.

To address this, future developments should focus on expanding the classification range by incorporating a broader and more diverse dataset. Increasing the number of gesture classes (especially those used in formal or informal sign languages), would allow the system to evolve into a fully featured recognition engine suitable for accessibility tools and multi-language support. Additionally, although the current model processes static images, integrating temporal information from video sequences would allow recognition of dynamic gestures. A dynamic system could capture the transitions between gestures and context within sequences, critical for applications like sign language interpretation or gesture-based control in interactive environments.

The system also shows strong potential for **multi-modal expansion**. For example, combining gesture recognition with facial expression analysis could enhance communication interpretation, especially for emotional or conversational contexts. This multi-modal approach could be explored further using existing facial recognition frameworks in conjunction with hand gesture classification. Furthermore, this gesture recognition system could be adapted to control digital interfaces (such as virtual keyboards, touchless control panels, or smart home systems), enabling hands-free interaction for accessibility and convenience. It could also be extended to detect broader non-verbal cues such as body posture or movements, expanding its use into areas like physical therapy, gaming, or security.

One compelling avenue for development is the application of this system in real-time sign language translation. This would involve processing continuous hand gestures from video input and mapping them into spoken or written language. Implementation

of a basic fingerspelling interpreter (manually coded languages) could be achieved with modest additional development. With further enhancement, a more comprehensive sign language translator could be created, facilitating communication for users with hearing impairments and supporting accessibility across multiple regional sign languages.

3D pose information of hand joints could provide more robust interpretation of hand gestures. Experimental tests using sensors to monitor real time digit and joint position could help reduce noise in the validation results.

In summary, this project lays a solid foundation for gesture-based input systems, with future development paths including gesture set expansion, dynamic sequence processing, multi-modal recognition, real-time translation, and embedded deployment. These improvements would enhance the utility, inclusiveness, and practicality of the system across a wide range of real-world applications.

REFERENCES

- [1] P. K. Pisharady and M. Saerbeck, "Recent Methods and Databases in Vision-based hand gesture recognition: A review," *Computer Vision and Image Understanding*, vol. 141, pp. 152-165, 2015.
- [2] C. Miller and M. Poston, "Types of Nonverbal Communication," 01 08 2020. [Online]. Available: <https://cod.pressbooks.pub/communication/chapter/4-2-types-of-nonverbal-communication/>. [Accessed 14 04 2025].
- [3] N. Fay et al., "Gesture is primary modality for language creation," *Proceedings of the Royal Society B: Biological Sciences*, vol. 289, 2022.
- [4] P. Feyereisen and J.-D. De, *Gestures and Speech: Psychological Investigations*, Cambridge: Cambridge University Press, 1991.
- [5] D. McNeill, "Hand and Mind," in *Advances in visual semiotics : the semiotic web 1992-93*, Berlin, Mouton de Gruyter, 1995, pp. 351-374.
- [6] M. Bohn, G. Kachel, and M. Tomasello, "Young children spontaneously recreate core properties of language in a new modality," *Proceedings of the National Academy of Sciences*, vol. 116, no. 51, pp. 26072-26077, 2019.
- [7] S. Goldin-Meadow, D. McNeill and J. Singleton, "Silence is liberating: Removing the handcuffs on grammatical expression in the manual modality.," *Psychological Review*, vol. 103, no. 1, pp. 34-55, 1996.
- [8] S. M. Mankar and S. A. Chhabria, "Review on hand gesture based mobile control application," in *2015 International Conference on Pervasive Computing (ICPC)*, Pune, 2015.
- [9] K. Cooperrider, "Universals and diversity in gesture," *Gesture*, vol. 18, no. 2-3, pp. 209 - 238, 2019.
- [10] F. Althoff, R. Lindl and L. Walchshausl, *ROBUST MULTIMODAL HAND- AND HEAD GESTURE RECOGNITION FOR CONTROLLING AUTOMOTIVE INFOTAINMENT SYSTEMS*, Munich: BMW Group Research and Technology, 2005.
- [11] M. Yaseen and S. Jusoh, "A systematic review on hand gesture recognition techniques, challenges and applications," *PeerJ Computer Science*, 2019.
- [12] M. Stoerring, T.B.Moeslund, Y. Liu and E. Granum, "Computer vision-based gesture recognition for an augmented reality interface," in *4th IASTED*

International Conference on VISUALIZATION, IMAGING, AND IMAGEPROCESSING, Marbella, 2004.

- [13] M. Oudah, A. Al-Naji, and J. Chahl,, "Hand Gesture Recognition Based on Computer Vision: A Review of Techniques," *Journal of Imaging*, vol. 6, no. 8, p. 73, 2020.
- [14] A. Mujahid et al., "Real-Time Hand Gesture Recognition Based on Deep Learning YOLOv3 Model," *Applied Science*, vol. 11, no. 9, p. 4164, 2021.
- [15] Department of Transport, "Signals to other road users - The Highway Code - Guidance," www.gov.uk, 01 10 2015. [Online]. Available: <https://www.gov.uk/guidance/the-highway-code/signals-to-other-road-users>. [Accessed 14 04 2025].
- [16] Coursera, "Deep Learning vs Machine Learning: A Beginner's Guide," 19 09 2024. [Online]. Available: <https://www.coursera.org/gb/articles/ai-vs-deep-learning-vs-machine-learning-beginners-guide>. [Accessed 16 04 2025].
- [17] J. Delua, "Supervised versus unsupervised learning: What's the difference?," 12 03 2021. [Online]. Available: <https://www.ibm.com/think/topics/supervised-vs-unsupervised-learning>. [Accessed 16 04 2025].
- [18] Lozano-Murcia, F. P. Romero, Jesús Serrano-Guerrer and Á. Olivas, "A Comparison between Explainable Machine Learning Methods for Classification and Regression Problems in the Actuarial Context," *Mathematics*, vol. 11, no. 14, pp. 3088-3088, 2023.
- [19] P. Grieve, "Deep learning vs. machine learning: what's the difference?," 2023 09 22. [Online]. Available: <https://www.zendesk.co.uk/blog/machine-learning-and-deep-learning/>. [Accessed 16 04 2025].
- [20] L. Craig, "CNN vs. RNN: How are they different?," TechTarget, 29 07 2024. [Online]. Available: <https://www.techtarget.com/searchenterpriseai/feature/CNN-vs-RNN-How-they-differ-and-where-they-overlap#:~:text=The%20main%20differences%20between%20CNNs,and%20RNNs%20have%20different%20architectures>. [Accessed 16 04 2025].
- [21] S. T. Boppiniti, "Big Data Meets Machine Learning: Strategies," *International Journal of Creative Research in Computer Technology and Design*, 2020.
- [22] S. Wong, A. Gatt, V. Stamatescu and M. McDonnell, "Understanding data augmentation for classification: when to warp?," 2016.

- [23] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of artificial intelligence research*, vol. 16, no. 1, pp. 321-357, 2002.
- [24] I. S. a. G. E. H. A. Krizhevsky, "ImageNet Classification with Deep Convolutional," *Communications of the ACM*, vol. 60, no. 6, pp. 84-90, 2012.
- [25] MathWorks. Inc., "resnet18," [Online]. Available: <https://uk.mathworks.com/help/deeplearning/ref/resnet18.html>. [Accessed 16 04 2025].
- [26] MathWorks, Inc., "Transfer Learning Using AlexNet - MATLAB & Simulink - MathWorks United Kingdom," MathWorks, Inc., [Online]. Available: <https://uk.mathworks.com/help/deeplearning/ug/transfer-learning-using-alexnet.html>. [Accessed 16 04 2025].
- [27] MathWorks. Inc., "imagePretrainedNetwork," MathWorks. Inc., [Online]. Available: <https://uk.mathworks.com/help/deeplearning/ref/imagepretrainednetwork.html>. [Accessed 16 04 2025].
- [28] A. Ullah, H. Elahi, Z. Sun, A. Khatoon and I. Ahman, "Comparative Analysis of AlexNet, ResNet18 and SqueezeNet with Diverse Modification and Arduous Implementation," *Arabian Journal for Science and Engineering*, 2021.
- [29] S. Yahya, A. Ramli, M. Nordin and H. Basarudin, "Comparison of Convolutional Neural Network Architectures for Face Mask Detection," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 12, 2021.
- [30] A. Kapitanov, "GitHub - hukenovs/hagrid: HAnd Gesture Image Recognition Dataset," 2025. [Online]. Available: <https://github.com/hukenovs/hagrid>. [Accessed 2025].
- [31] J. Terra, "simplilearn.com," 04 05 2025. [Online]. Available: <https://www.simplilearn.com/what-is-a-roc-curve-and-how-to-use-it-in-performance-modeling-article>. [Accessed 05 05 2025].
- [32] M. S. C. L. M. W. Babak Toghiani-Rizi, "Static Gesture Recognition using Leap Motion," Uppsala University, 2017.