**Hand Gesture Recognition System and Survey**

**Submitted for**

**Statistical Machine Learning CSET211**

Submitted by:

**(E23CSEU0504) ABHIK GHOSH**

Submitted to

**DR. Amit Soni**

**July-Dec 2024**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

A close-up of a logo

Description automatically generated

**INDEX**

Contents

[Abstract 3](#_Toc182694715)

[Introduction 3](#_Toc182694716)

[Related Work 4](#_Toc182694717)

[Methodology 4](#_Toc182694718)

[Hardware/Software Required 7](#_Toc182694719)

[Experimentation and Results: 8](#_Toc182694720)

[Conclusions: 9](#_Toc182694721)

[Future Scope 10](#_Toc182694722)

[GitHub Link 10](#_Toc182694723)

# Abstract

The project aims to develop a Hand Gesture Recognition System using Convolutional Neural Networks (CNN). The system includes multiple models, each trained to recognize various hand gestures. A mobile application is integrated with the system, enabling users to test and evaluate each model. The primary goal is to gather feedback from a wide range of users through the app, determining which model performs best in real-world scenarios. This survey-based approach provides insights into the most effective model for practical deployment.

# Introduction

Image classification is a fundamental task in computer vision with applications spanning autonomous vehicles, healthcare, and security systems. The development of deep learning models, particularly Convolutional Neural Networks (CNNs), has significantly improved classification accuracy. This report discusses an image classification project using a CNN architecture on the Hagrid 150k dataset, exploring existing research and external findings to contextualize and validate the approach.

This purpose of this project is to compare custom classification CNN architecture with standard models used for most research applications in this field and utilize a mobile app-based survey to find the perfect balance between complexity and accuracy for gesture recognition.

# Related Work

Previous work on hand gesture recognition has explored techniques ranging from traditional computer vision to machine learning and deep learning. Early approaches relied on feature extraction methods like Histogram of Oriented Gradients (HOG) and Support Vector Machines (SVM). Recent studies have demonstrated the effectiveness of CNNs, known for their ability to automatically learn spatial hierarchies from images. This project builds on previous research by comparing multiple CNN architectures for optimal gesture recognition in real-world conditions.

Citations:

<https://openaccess.thecvf.com/content/WACV2024/html/Kapitanov_HaGRID_--_HAnd_Gesture_Recognition_Image_Dataset_WACV_2024_paper.html>

<https://ieeexplore.ieee.org/abstract/document/10543590/>

<https://ieeexplore.ieee.org/abstract/document/10484421/>

# Methodology

1. Dataset Preparation

<https://www.kaggle.com/datasets/innominate817/hagrid-classification-512p-no-gesture-150k>



The Hagrid 150k dataset is a large-scale collection of hand gesture images designed for classification tasks. This dataset includes 150,000 images with varied lighting conditions, hand postures, and backgrounds, offering a robust benchmark for assessing model performance in practical settings. The diversity in the data presents challenges typical of real-world applications, such as noise and occlusion.

It has 19 distinct classes with multiple people from varying surroundings in order to adapt to any environment, a smaller dataset is used for some models containing only 6 classes.

1. Model Development

The existing models based off the Hagrid dataset involve training well know image classification models, Hence this project aims to experiment with alternative model architectures :

* Simple CNN involving few layers
* Complex CNN involving multiple layers

For comparison we have also trained a ResNET50 model on the same dataset to compare our architecture and finding the perfect balance between complexity and accuracy.

The general pipeline for our custom models is as follows:

* **Data Preprocessing**: Images were normalized and resized to 256x256 pixels for uniform input. Data augmentation (flipping, rotation, zoom) was applied to enhance generalization.
* **Model Architecture**: A custom CNN architecture comprises of:
  + **Convolutional Layers**: For feature extraction, followed by ReLU activation.
  + **Pooling Layers**: Max-pooling was employed for down-sampling.
  + **Fully Connected Layers**: For classification, concluding with a softmax activation.
* **Training and Optimization**: The model was trained using categorical cross-entropy as the loss function and Adam as the optimizer, with an initial learning rate of 0.001. Early stopping and dropout regularization were employed to prevent overfitting.

These custom models are compared to existing architectures via the mobile app and collect survey data.

1. Mobile App Integration

A Flutter-based mobile app is created to facilitate user interaction with the system. The app provides an interface to select different models, test them, and vote for the best-performing model.

The app has a initiative UI and hopes to reach a wide audience.

The latest version of the android app can be downloaded here:[ADD LINK]

# Hardware/Software Required

* **Hardware**:

GPU-enabled system for training models (NVIDIA GTX 1080 or higher recommended)

Smartphones (Android) for app testing

* **Software**:
  1. **Model Training**
* **Python 3.12.x**
* **Tensorflow / Keras**
  1. **Mobile App**
* **Android Device**
* **Mobile App([LINK HERE])**

# Experimentation and Results:

The model was trained using the following parameters:

* Optimizer: Adam
* Loss Function: SparseCategoricalCrossentropy
* Epochs: 15
* Batch Size: 128

The model was evaluated using a separate validation set. The performance metrics used include accuracy, precision, recall, and F1-score.

The final model achieved the following results on the validation set:

**Simple CNN Metrics**

* Accuracy: 82%
* Precision: 80%
* Recall: 79%
* F1-score: 80%

**Simple Model Size**

* keras file: 96.4 MB
* tflite: 32.1 MB

**Complex CNN Metrics**

* Accuracy: 74%
* Precision: 72%
* Recall: 71%
* F1-score: 71%

**Complex Model Size**

* keras file: 385 MB
* tflite: 128 MB

**Compareable ResNET50 Model**

* Accuracy: 93%
* Precision: 93%
* Recall: 92%
* F1-score: 92%

**ResNET50 Model Size**

* keras file: 271 MB
* tflite: 89.8 MB

**Compareable ResNET50 Model**

* Accuracy: 91%

**Teachable Machine Model Size**

* keras file: 2.34 MB
* tflite: 1.99 MB

# Conclusions:

The project successfully demonstrates the use of CNN for hand gesture recognition. The user survey through the Flutter app hopes to establish a comprehensive evaluation of multiple models, highlighting the strengths and weaknesses of each.

Based on only validation Data the ResNET model proves to be the most promising architecture, we can use the survey data to confirm or deny this hypothesis.

# Future Scope

Future improvements could include:

* Expanding the dataset to encompass more diverse gestures and environments.
* Ability to load and benchmark User Made Models
* Refining the Flutter app interface for enhanced user experience.
* Incorporating advanced CNN architectures and hybrid deep learning models for better accuracy.
* Exploring real-time gesture recognition capabilities for AR/VR applications.
* Validating model performance over extended periods with a larger user base.

# GitHub Link