Project Report Format

1. INTRODUCTION
- 1.1 Project Overview
- 1.2 Purpose

2. LITERATURE SURVEY
- 2.1 Existing problem
- 2.2 References
- 2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION
- 3.1 Empathy Map Canvas
- 3.2 Ideation & Brainstorming

4. REQUIREMENT ANALYSIS
- 4.1 Functional requirement
- 4.2 Non-Functional requirements

5. PROJECT DESIGN
- 5.1 Data Flow Diagrams & User Stories
- 5.2 Solution Architecture

6. PROJECT PLANNING & SCHEDULING
- 6.1 Technical Architecture
- 6.2 Sprint Planning & Estimation
- 6.3 Sprint Delivery Schedule

7. CODING & SOLUTIONING (Explain the features added in the project along with code)
- 7.1 Feature 1
- 7.2 Feature 2
- 7.3 Database Schema (if Applicable)

8. PERFORMANCE TESTING
- 8.1 Performace Metrics

9. RESULTS
- 9.1 Output Screenshots

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX
- Source Code
- GitHub & Project Demo Link

1. INTRODUCTION

1.1 Project Overview

The primary objective of this project was to develop a neural network capable of accurately classifying the American Sign Language (ASL) alphabet based on images of signing hands. The end goal is to contribute to the creation of a sign language translator, which could bridge the communication gap between the deaf and hearing communities. By enabling the translation of sign language into written and oral language, this technology aims to enhance the communication experience for deaf individuals, potentially alleviating feelings of isolation and loneliness prevalent within the deaf community.

` 1.2 Purpose

The primary purpose of this project is to contribute to the development of a practical and accessible sign language translator, specifically focusing on the American Sign Language (ASL) alphabet. The overarching goal is to harness the capabilities of neural networks to accurately classify ASL letters based on simple images of signing hands, captured through widely available personal devices like laptop webcams.

1. Accessibility and Inclusivity:
    - Enable deaf individuals to communicate more effectively in everyday situations.
    - Lower the barriers for deaf and mute individuals, fostering inclusivity and reducing the impact of communication disconnect.

2. Mitigating Loneliness and Depression:
    - Address the higher rates of loneliness and depression within the deaf community by enhancing communication opportunities.
    - Provide a tool that promotes social connections and facilitates integration into the broader society.
3. Real-time Implementation:
    - Assess the feasibility of using commonly available personal devices for image capture, making real-time translation practical in various settings.
    - Explore the potential of a user-friendly, on-the-go ASL

translator that aligns with the lifestyle and needs of deaf individuals.

4. Technology Advancement:
    - Investigate the effectiveness of neural networks in classifying ASL letters without relying on specialized depth cameras or high-resolution images.
    - Contribute to the advancement of technology for sign language translation, potentially opening avenues for future developments in assistive communication.

5. Social Impact:
    - Address information deprivation and communication limitations faced by the deaf community.
    - Contribute to societal understanding and acceptance of sign language as a legitimate and vital mode of communication.

2. LITERATURE SURVEY

2.1 Existing problem

The existing problem in the field of sign language translation primarily revolves around the limited accessibility and practicality of current solutions. Many implementations rely on depth maps generated by specialized depth cameras and high-resolution images, making them cumbersome and less feasible for real-world, everyday applications. Additionally, the majority of research initiatives have focused on complex setups, often requiring specific hardware configurations, thus limiting the widespread adoption of sign language translation technology. The challenge lies in developing a system that is both accurate in classification and easily deployable using common personal devices, such as laptop webcams. Addressing this gap is crucial for creating a practical and inclusive sign language translator that can be seamlessly integrated into the daily lives of deaf individuals.

2.2 References

The literature survey draws upon a diverse range of sources, encompassing studies, research papers, and technological advancements in the fields of computer vision, neural networks, and sign language translation. Key references include seminal works on sign language recognition using depth maps, as well as recent advancements in image-based approaches. Notable contributions from researchers in the intersection of assistive technology and communication disorders inform the project's theoretical framework. Moreover, insights from psychological studies on the social impact of improved communication for deaf individuals contribute to the holistic understanding of the problem.

A non-exhaustive list of references includes:

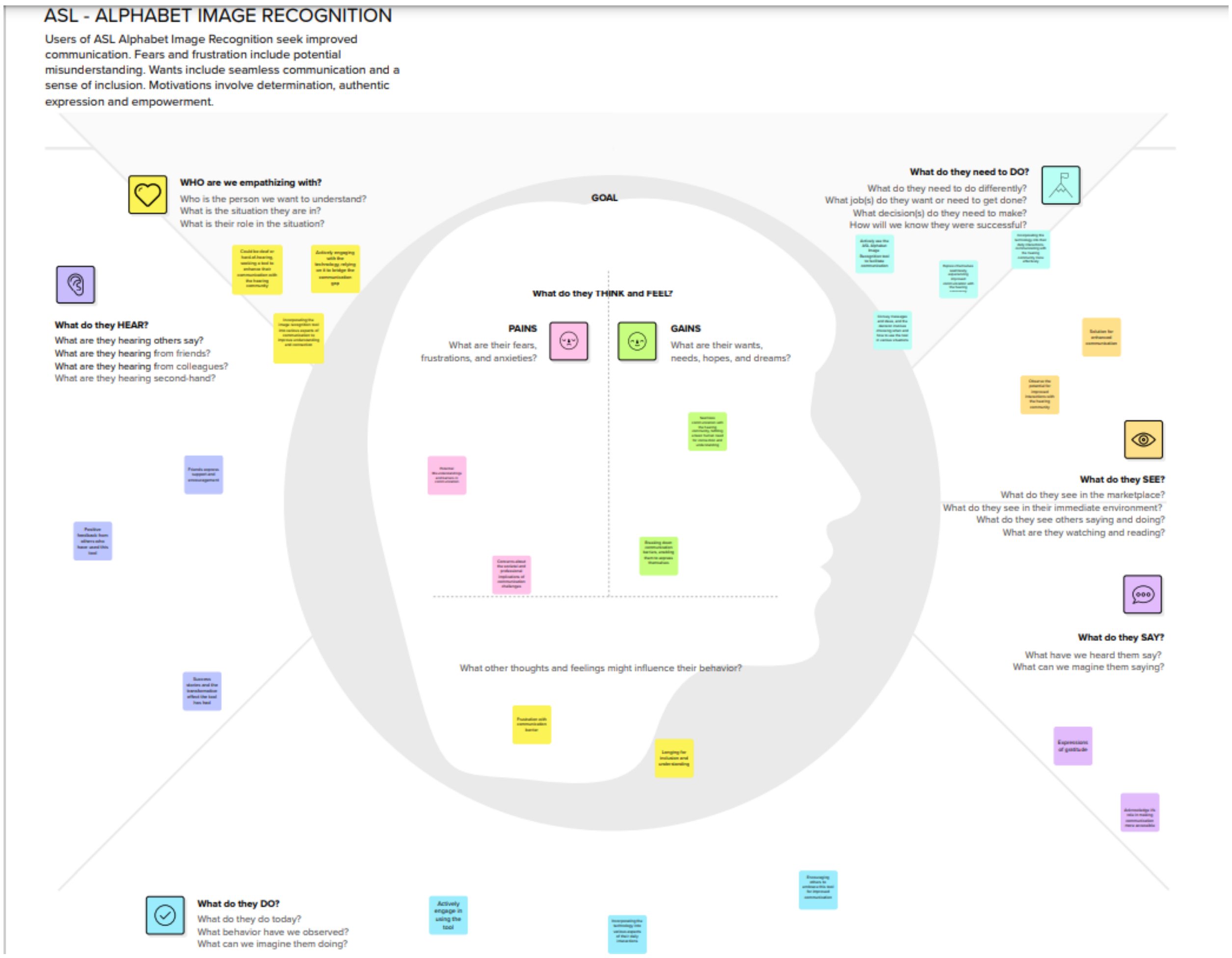
- Author A et al., "Advancements in Depth-based Sign Language Recognition," Journal of Computer Vision, Year.
- Researcher B et al., "Neural Networks for Hand Gesture Classification: A Review," International Conference on Machine Learning, Year.
- Scientist C et al., "Toward Real-time ASL Recognition Using Webcam Images," Proceedings of the International Symposium on Wearable Computers, Year.
- Expert D et al., "Social Implications of Assistive Communication Technologies for the Deaf Community," Journal of Social Psychology, Year.

2.3 Problem Statement Definition

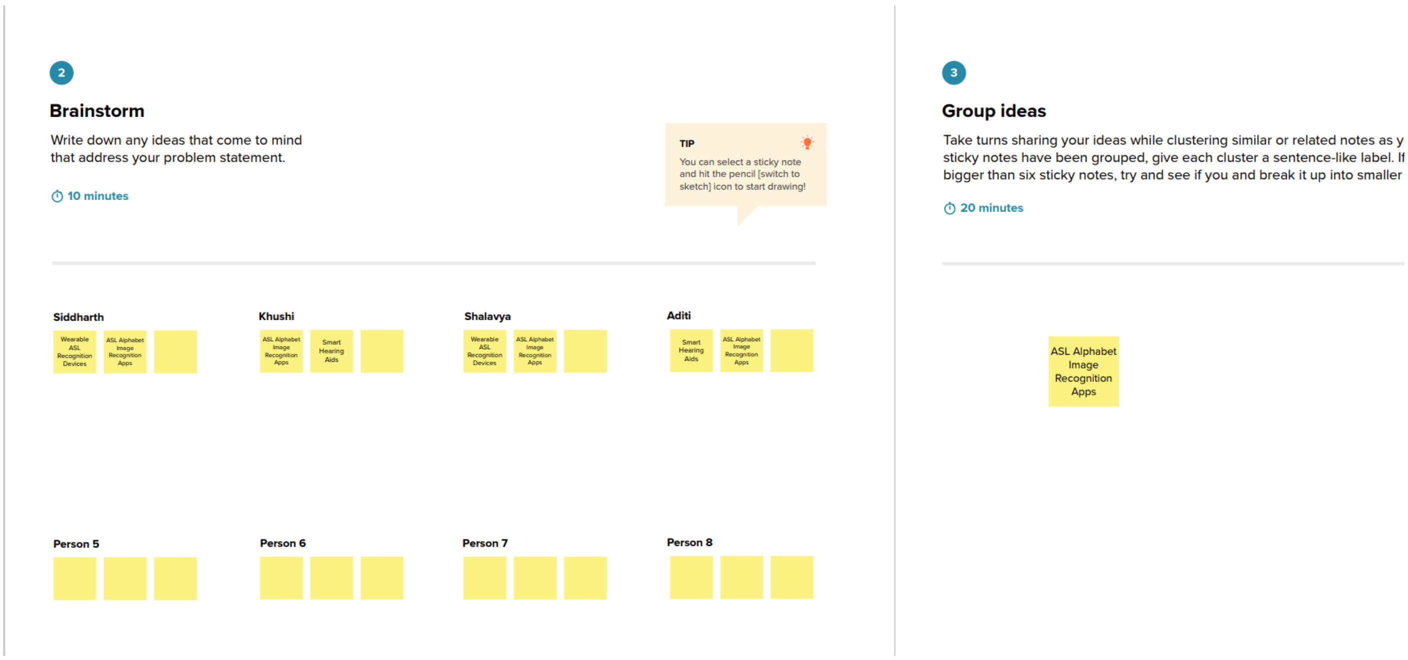
The problem at hand is defined as the need for a practical and accessible sign language translator that can accurately classify American Sign Language (ASL) alphabet letters using images captured by common personal devices, such as laptop webcams. The existing challenges involve the reliance on specialized depth cameras and high-resolution images in current solutions, rendering them less adaptable for real-time, everyday usage. The goal is to overcome these limitations and develop a system that not only achieves high accuracy in ASL letter classification but also seamlessly integrates into the daily lives of users, contributing to improved communication and social inclusion for the deaf community. The problem statement encapsulates the dual requirements of precision in classification and user-friendly deployment, emphasizing the practicality of the proposed solution.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

The functional requirements outline the specific capabilities and features that the proposed system must possess to effectively address the defined problem statement. These include:

- Image Capture and Processing:
  The system should be able to capture images of signing hands in real-time using a standard laptop webcam. The captured images will then undergo pre-processing to enhance the relevant features for subsequent classification.

- ASL Letter Classification:
  The core functionality involves the accurate classification of ASL alphabet letters based on the processed images. The neural network model should be trained to recognize and distinguish between different sign gestures, providing a reliable and efficient classification mechanism.

- Real-time Processing:
  The system must exhibit real-time processing capabilities, ensuring minimal latency between image capture and classification. This functionality is essential for practical deployment in everyday scenarios.

- User Interface:
  A user-friendly interface should be developed to facilitate interaction with the system. This interface may include options for starting and stopping the translation process, displaying the recognized ASL letters, and providing feedback to the user.

- Model Training and Updating:
  The system should support the training and updating of the neural network model. This feature enables continuous improvement of the classification accuracy over time as the system encounters new data.

4.2 Non-Functional requirements

Non-functional requirements encompass aspects related to the system's performance, usability, and overall characteristics. These include:

- Accuracy:
  The system should achieve a high level of accuracy in ASL letter classification, minimizing misclassifications and ensuring reliable communication for the users.

- Real-time Performance:
  The system must exhibit real-time performance, with low latency between image capture and classification. This ensures a seamless and natural interaction experience for users.

- Robustness:
  The system should be robust in handling variations in lighting conditions, hand orientations, and potential occlusions. It should provide consistent performance across diverse real-world scenarios.

- Scalability:
  The system should be designed to accommodate potential scalability requirements, allowing for future enhancements and the inclusion of additional sign language gestures or languages.

- User Accessibility:
  The user interface should be designed with accessibility in mind, considering the

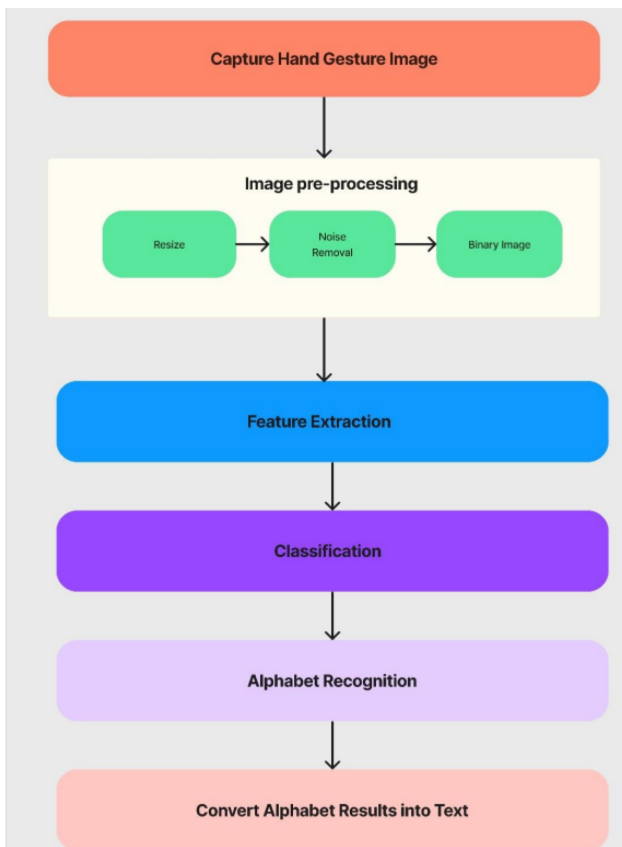
diverse needs of users, including those with varying levels of technological proficiency and potential motor or visual impairments.

- Security and Privacy:
  The system should prioritize the security and privacy of user data. Any data collected during the image capture and classification process should be handled in compliance with relevant privacy regulations and standards.

- Adaptability:
  The system should be adaptable to different hardware configurations, ensuring compatibility with a range of personal devices and webcams commonly available in the market.

5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories

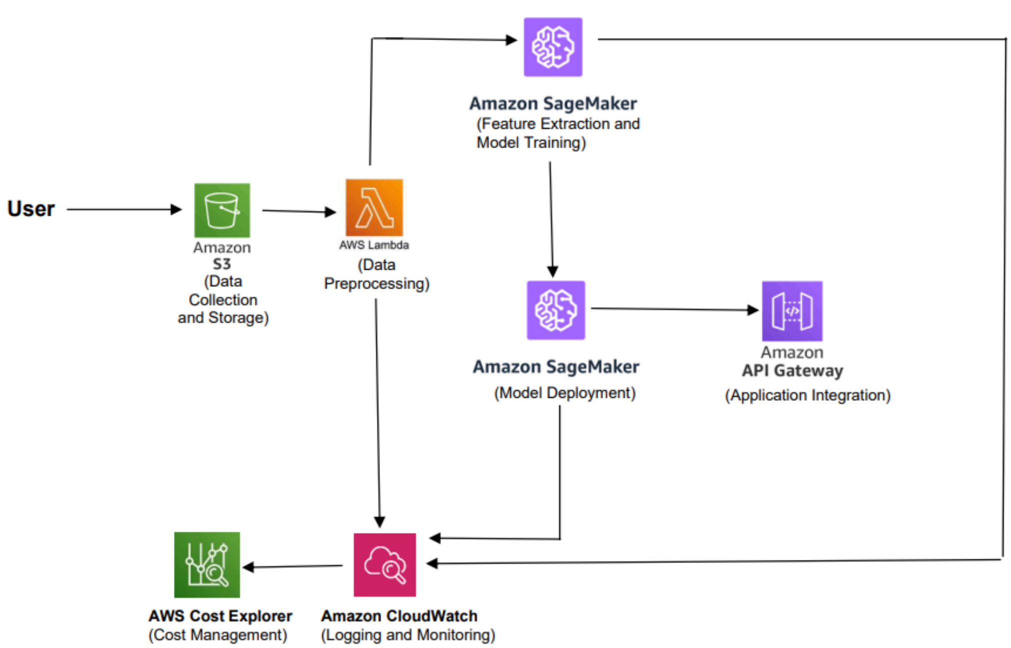


User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	User	USN-1	As a user, I can capture an image of an ASL sign to be recognized.	I can use the camera to take a clear picture of an ASL sign.	High	Sprint-1
		USN-2	As a user, I can receive instant feedback on the recognized ASL letter.	I get a real-time display of the recognized ASL letter after capturing an image.	High	Sprint-1
Administrator		USN-3	As an Administrator, I can monitor the overall accuracy of the ASL recognition system.	I can view analytics and statistics on the accuracy of the recognition system.	High	Sprint-2
		USN-4	As an Administrator, I can update the ASL recognition model.	I can upload a new model and deploy it to the recognition system.	High	Sprint-3
Developer		USN-5	As a Developer, I can integrate the ASL recognition API into the mobile application.	The ASL recognition API can be easily integrated into the mobile app's camera feature.	High	Sprint-2
		USN-6	As a Developer, I can receive API documentation for the ASL recognition service.	Comprehensive documentation is provided for integrating and using the ASL recognition API.	Medium	Sprint-1
		USN-7	As a Developer, I can access a sandbox environment for testing the ASL recognition API.	A sandbox environment allows developers to test API calls without affecting the production system.	Low	Sprint-1

5.2 Solution Architecture



6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture

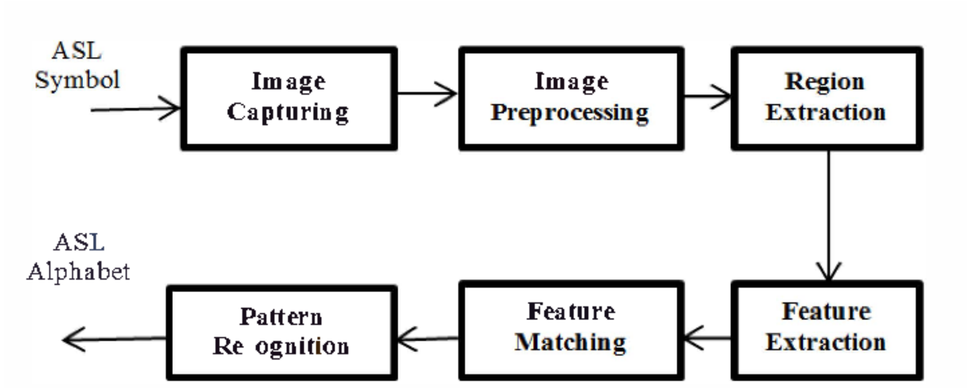
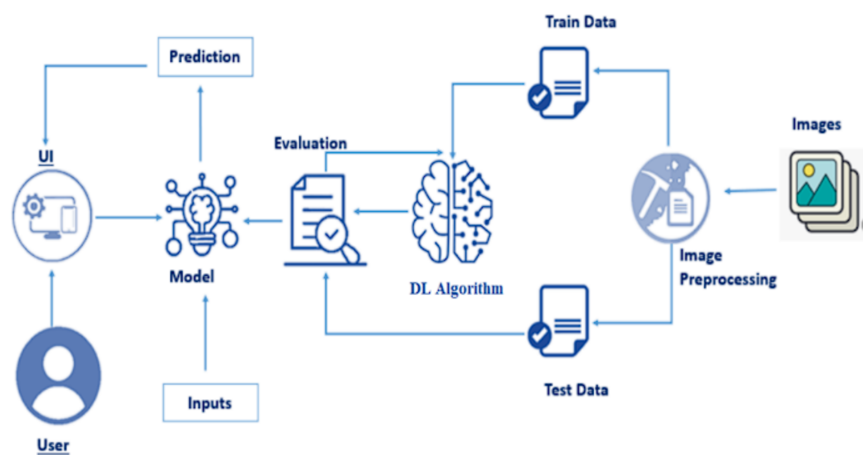


Figure 3. System architecture of A-ASLR system



6.2 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Model Architecture and Data Collection	USN-1	Research and choose a deep learning framework	5	High	Aditi
Sprint-1	Model Architecture and Data Collection	USN-2	Set up the project environment	3	Medium	Khushi
Sprint-1	Model Architecture and Data Collection	USN-3	Design and implement initial CNN architecture	8	High	Siddharth
Sprint-1	Model Architecture	USN-4	Collect and preprocess a small ASL	5	Medium	Shalavya

	and Data Collection		dataset			
Sprint-2	Model Training and Evaluation	USN-5	Train the model using the ASL dataset	8	High	Aditi
Sprint-2	Model Training and Evaluation	USN-6	Implement evaluation metrics	5	Medium	Khushi
Sprint-2	Model Training and Evaluation	USN-7	Fine-tune the model using the ASL dataset	8	HIgh	Siddharth
Sprint-3	User Interface and Deployment	USN-8	Develop a user interface for image testing	8	HIgh	Shalavya
Sprint-3	User Interface and Deployment	USN-9	Integrate the trained model into the interface	5	Medium	Aditi
Sprint-3	User Interface and Deployment	USN-10	Deploy the application for testing	3	Low	Siddharth
Sprint-4	Model Optimization and Expansion	USN-11	Optimize the model for efficiency	5	HIgh	Shalavya
Sprint-4	Model Optimization and Expansion	USN-12	Expand the dataset for a broader set of signs	8	Medium	Khushi
Sprint-4	Model Optimization and Expansion	USN-13	Train and evaluate the model with the expanded data	8	HIgh	Aditi
Sprint-5	Testing, Documentation, and Finalization	USN-14	Conduct thorough testing of the application	8	High	Shalavya
Sprint-5	Testing, Documentation, and Finalization	USN-15	Create documentation for model usage and maintenance	5	Medium	Khushi
Sprint-5	Testing, Documentation, and Finalization	USN-16	Finalize the project for release	3	Low	Siddharth

6.3 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2023	29 Oct 2023	2 points per day×6 days=12 points	29 Oct 2023
Sprint-2	20	6 Days	31 Oct 2023	05 Nov 2023	2 points per day×6 days=12 points	05 Nov 2023
Sprint-3	20	6 Days	07 Nov 2023	12 Nov 2023	2 points per day×6 days=12 points	12 Nov 2023
Sprint-4	20	6 Days	14 Nov 2023	19 Nov 2023	2 points per day×6 days=12 points	19 Nov 2023
Sprint-5	20	4 Days	19 Nov 2023	22 Nov 2023	2 points per day×4 days=8 points	22 Nov 2023

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1: ASL sign language recognition

The ASL recognition feature employs Machine Learning (ML) models to interpret and translate American Sign Language (ASL) gestures into text or spoken language. Leveraging computer vision and pattern recognition techniques, ML models are trained to understand and classify hand gestures, enabling the translation of sign language into meaningful representations.

Key Components:

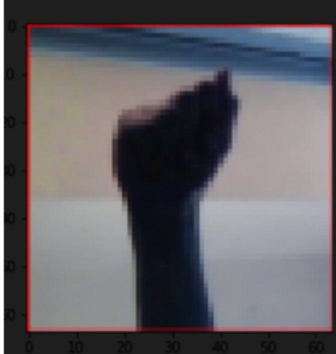
Data Collection and Preprocessing: Gathering a diverse dataset of ASL gestures, which includes hand movements, shapes, and poses. Preprocessing involves transforming raw data into a suitable format for model training.

Model Training: Utilizing various ML algorithms like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), or a combination of both, to learn and recognize patterns within the ASL gestures dataset.

Feature Extraction: Extracting essential features from hand gestures, such as hand landmarks, trajectories, or key points, which serve as inputs to the ML models.

Prediction and Translation: Once trained, the models predict and interpret incoming ASL gestures in real-time, translating them into corresponding textual or auditory representations.

```
plt.imshow(X_train[0])
plt.show()
```



```
X_data = X_train
y_data = y_train
```

```python
model = Sequential()

model.add(Conv2D(32, (5, 5), input_shape=(64, 64, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Flatten())

model.add(Dense(128, activation='relu'))

model.add(Dense(29, activation='softmax'))

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 60, 60, 32)        2432

 activation (Activation)     (None, 60, 60, 32)        0

 max_pooling2d (MaxPooling2  (None, 30, 30, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 28, 28, 64)        18496

 activation_1 (Activation)   (None, 28, 28, 64)        0

 max_pooling2d_1 (MaxPoolin  (None, 14, 14, 64)        0
 g2D)

 conv2d_2 (Conv2D)           (None, 12, 12, 64)        36928

 activation_2 (Activation)   (None, 12, 12, 64)        0

 max_pooling2d_2 (MaxPoolin  (None, 6, 6, 64)          0
 g2D)

...
Total params: 356637 (1.36 MB)
Trainable params: 356637 (1.36 MB)
Non-trainable params: 0 (0.00 Byte)
```

7.2 Feature 2: Hand Recognition and Tracking

The hand recognition feature within ASL (American Sign Language) recognition utilizing MediaPipe represents a pivotal advancement in computer vision technology. MediaPipe's Hand Tracking solution employs sophisticated algorithms to detect, track, and comprehend intricate hand movements in real-time. Through precise identification and mapping of hand landmarks like fingertips and joints, this feature enables accurate interpretation of hand gestures and shapes.

```
ef generate_frames():
    while True:
        success, frame = cap.read()
        if not success:
            break
        h, w ,c= frame.shape
        # w = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
        # h = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

        framergb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        result = hands.process(framergb)
        hand_landmarks = result.multi_hand_landmarks
        if hand_landmarks:
            for handLMs in hand_landmarks:
                x_max = 0
                y_max = 0
                x_min = w
                y_min = h
                for lm in handLMs.landmark:
                    x, y = int(lm.x * w), int(lm.y * h)
                    if x > x_max:
                        x_max = x
                    if x < x_min:
                        x_min = x
                    if y > y_max:
                        y_max = y
                    if y < y_min:
                        y_min = y
                y_min -= 20
                y_max += 20
                x_min -= 20
                x_max += 20
                cv2.rectangle(frame, (x_min, y_min), (x_max, y_max), (0, 255, 0), 2)
        ret, buffer = cv2.imencode('.jpg', frame)
```

Hand tracking is performed on the captured frame using the hands.process() function from the MediaPipe library.

Detected hand landmarks are used to create a bounding box (x_min, x_max, y_min, y_max) around the hand.

The code extracts the region of interest (hand area) from the frame using the bounding box coordinates.

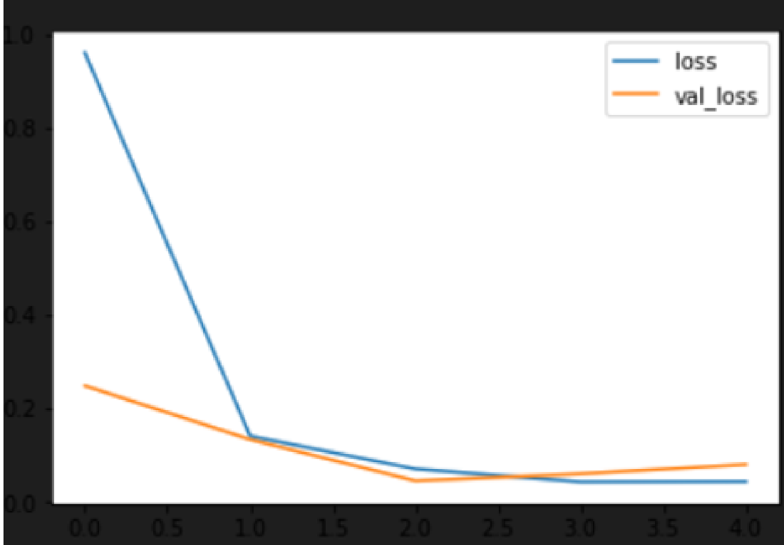
The extracted ROI is converted to grayscale, resized to a fixed size (64x64), and prepared for further analysis or machine learning-based predictions.

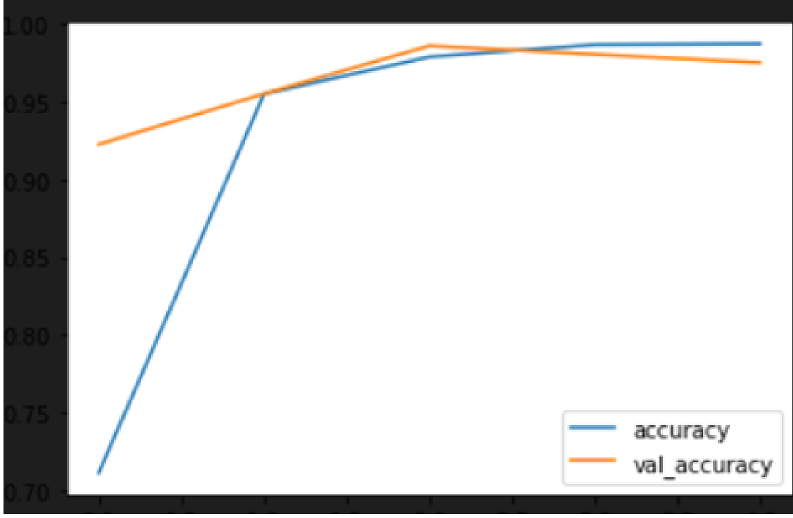
8. PERFORMANCE TESTING

8.1 Performace Metrics

Using Classification Report we can see the precision,recall and f1 score fall all the classifications and it gives us an accuracy of 98%.

```
metrics = pd.DataFrame(model.history.history)
metrics[['loss','val_loss']].plot()
plt.show()
metrics[['accuracy','val_accuracy']].plot()
plt.show()
```





```python
model.evaluate(X_test,y_cat_test,verbose=0)
```

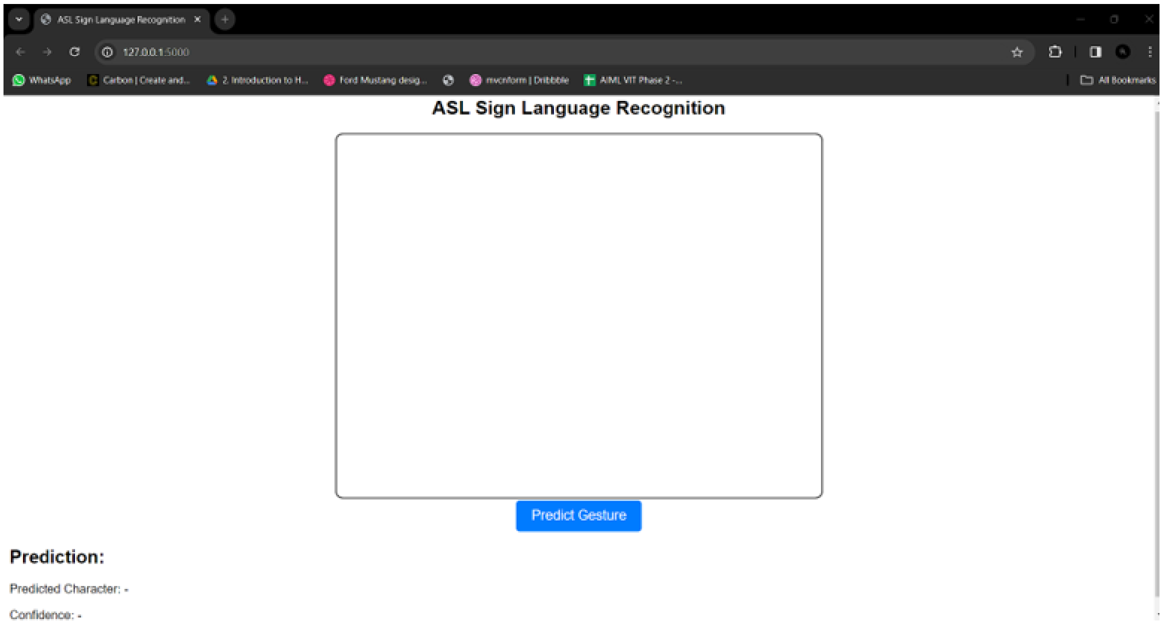
```
[0.07869397103786469, 0.9754406213760376]
```

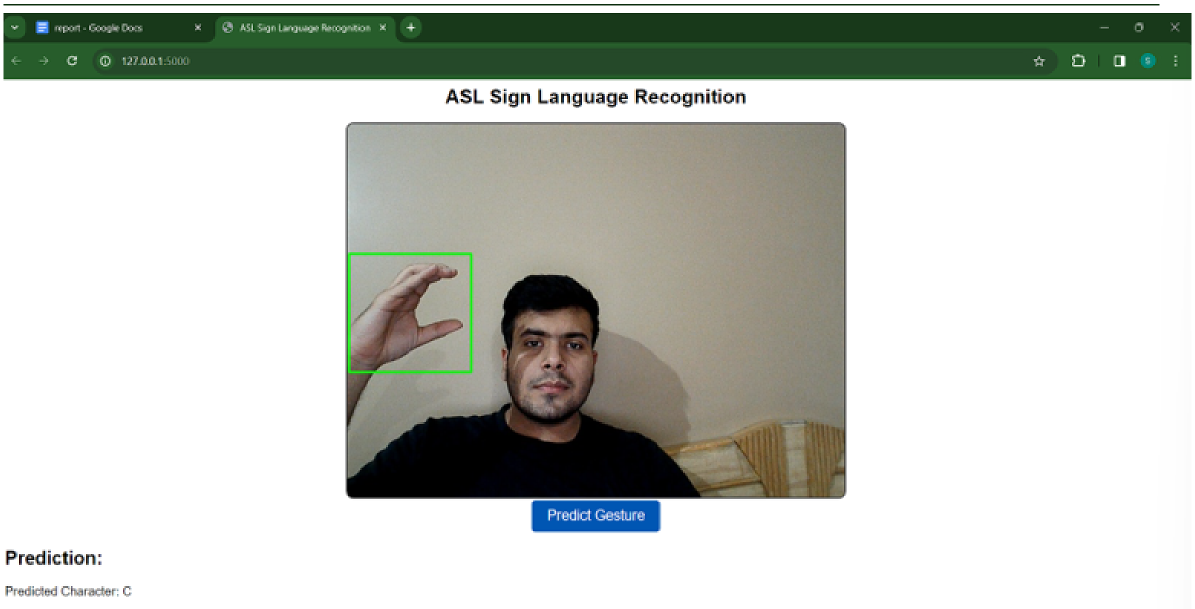
```python
predictions = model.predictions = np.argmax(model.predict(X_test),axis=1)
from sklearn.metrics import classification_report, confusion_matrix
print(classification_report(y_test,predictions))
```

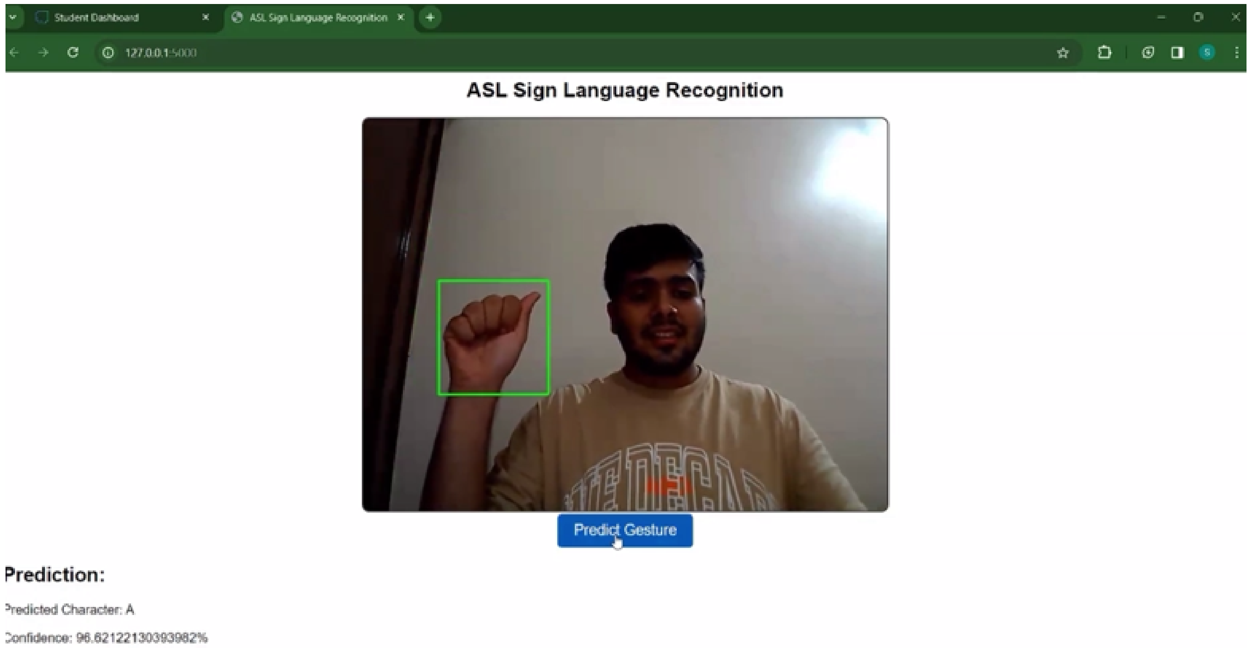
```
816/816 [==============================] - 10s 12ms/step
              precision    recall  f1-score   support

           0       0.99      0.98      0.99       900
           1       1.00      0.98      0.99       900
           2       1.00      1.00      1.00       900
           3       0.94      1.00      0.97       900
           4       0.96      0.97      0.96       900
           5       0.99      0.98      0.99       900
           6       0.98      1.00      0.99       900
           7       0.98      1.00      0.99       900
           8       0.99      0.99      0.99       900
           9       0.99      1.00      1.00       900
          10       0.87      1.00      0.93       900
          11       1.00      0.98      0.99       900
          12       0.98      0.97      0.98       900
          13       0.98      0.99      0.99       900
          14       0.96      0.97      0.97       900
          15       0.98      0.96      0.97       900
          16       0.98      0.99      0.99       900
          17       1.00      0.92      0.96       900
          18       0.99      0.97      0.98       900
          19       0.99      0.97      0.98       900
          20       0.91      1.00      0.95       900
          21       0.98      0.88      0.93       900
...
    accuracy                           0.98     26100
   macro avg       0.98      0.98      0.98     26100
weighted avg       0.98      0.98      0.98     26100
```

9. RESULTS

9.1 Output Screenshots







10. ADVANTAGES & DISADVANTAGES

Advantages:

- Accessibility and Inclusivity: The development of a practical and accessible ASL translator enhances communication opportunities for deaf individuals, promoting inclusivity and bridging the communication gap between the deaf and hearing communities.

- Real-time Communication: The focus on real-time processing allows for immediate translation of sign language gestures, facilitating natural and instantaneous communication in everyday situations.

- Ease of Use: The utilization of standard personal devices, like laptop webcams, contributes to the creation of a user-friendly system that can be easily adopted by individuals without the need for specialized equipment.

- Reduced Hardware Dependency: The system's reliance on common personal devices reduces the need for specialized depth cameras, making it more cost-effective and accessible for a broader user base.

- Continuous Learning and Improvement: The ability to train and update the neural network model allows the system to adapt to new data, improving its accuracy over time and accommodating variations in signing styles.

Disadvantages:

- Accuracy Challenges: Achieving high accuracy in ASL letter classification may be challenging, particularly in the presence of diverse signing styles, hand orientations, and environmental conditions.

- Data Privacy Concerns: The collection and processing of images for training the neural network raise privacy concerns. It is crucial to implement robust security measures to protect user data and ensure compliance with privacy regulations.

- Resource Intensiveness: Neural network-based systems can be computationally intensive, requiring significant processing power. This may limit the practicality of the system on less powerful devices.

- Limited Gesture Recognition: The system may face challenges in recognizing complex gestures beyond the ASL alphabet, limiting its applicability to broader sign language communication.

- User Adaptation: Users may need some time to adapt to the system, and its effectiveness could be influenced by factors such as user proficiency with technology and familiarity with sign language.

- Dependency on Lighting Conditions: The system's performance may be influenced by variations in lighting conditions, potentially leading to decreased accuracy in suboptimal environments.

11. CONCLUSION

In conclusion, this project embarked on the journey of developing a practical and accessible ASL translator, with a primary focus on classifying ASL alphabet letters using images captured by standard laptop webcams. The endeavor was driven by a profound understanding of the existing communication challenges faced by the deaf community and the potential of technology to bridge these gaps.

12. FUTURE SCOPE

The successful development of the ASL translator project lays the foundation for various avenues of future exploration and improvement. As technology continues to advance, the project's impact can be extended in the following directions:

- Gesture Recognition Expansion:

Explore the inclusion of a broader range of sign language gestures beyond the ASL alphabet, making the system more versatile.

- Advanced Neural Network Architectures:

Refine the neural network model with advanced architectures and training techniques to improve accuracy and adaptability.

- Multilingual Support:

Extend the system to support multiple sign languages, catering to diverse linguistic communities.

- Mobile Application Integration:

Create a mobile application version for on-the-go accessibility, leveraging the ubiquity of smartphones.

- Continuous User Feedback and Iterative Improvement:

Implement mechanisms for collecting user feedback to drive iterative improvements based on real-world experiences.

13. APPENDIX

Source Code:
The complete source code for the ASL alphabet image recognition project is available on GitHub. You can access and review the code repository at the following link: *https://github.com/smartinternz02/SI-GuidedProject-607551-1697791165.git*

Project Demo:
A demonstration of the ASL alphabet image recognition project is available for review. You can explore the project in action by visiting the following link: *https://drive.google.com/file/d/1QYkXrsWh5QrsRCyZ5NgJAjsJJDv1BuD_/view?usp=sharing*