CSY4010 (Computing Dissertations) Interim Report on

**Sign Language Interpreter**

**BY**

Nima Dorje Lama

UN ID: 21422145

BSc Computing (Software Engineering)

Supervisor: Roshan Koju

NAMI College

University of Northampton

Jorpati, Kathamandu, Nepal

16th June 2023

**Table of Contents**

* 1. Project Introduction
     1. Project background
     2. Project Aims and Objectives
  2. Literature review
  3. Requirement Analysis
     1. Comparable System
     2. Elicitation Activities
        1. Interview plans
        2. Interview findings
        3. Interview 1– client name
        4. Interview 2-client name
  4. Requirement Specification
     1. Problem Domain Research
     2. Information Gathering
     3. Identification of alternative solution from existing systems
     4. Proposed solution strategy
     5. Functional Requirements
  5. Research Methodology and implementation
     1. Data collection
     2. Data processing
     3. Model Training
     4. Model Evaluation
     5. Integration
     6. Deployment
     7. Maintenance
  6. Testing and Evaluations
  7. System Design and Analysis
     1. Entity Relationship diagram
     2. Database schema and its Relationship.
     3. Use Case Diagram
  8. Resources Requirements
  9. References
  10. Appendix
      1. Project Gantt Chart

## Project Introduction

According to recent records there are around 700,000 to 900,000 deaf people in the world. Sign language is the gestural language that is used by deaf people or people with hearing disability. Sign language uses hand gestures and expressions to communicate with each other. The sign language interpreter system aims to help users to close communication barriers faced by deaf and other people. This project uses deep learning methods, specifically Python’s Keras library to convert sign language hand gestures into textual form in real-time so that others can know what they are saying. It closes the communication barriers between deaf and non-deaf people.

In recent years technology has been so huge in industry and day to day work. There is also advancement in deep learning and computer vision which helps for the automation of sing language interpreter systems. Although it needs some improvement in handling various sign languages. To improve this problem the project uses Django framework as it helps to create a beautiful design and easy to understand system and its interface which can be used by all.

This system’s main objective is to include give accurate sign language interpreter system which is capable of converting sign language to textual form. It breaks down the hand gesture and helps to recognize the gesture and gives text as an output which helps in breaking barrier in communication. By using the power of deep neural networks, data processing techniques and the web base platform this system will help everyone try to learn or convert sign language to text form. This system also aims to help sign language users to participate in various domains such as education, healthcare, employment, and social interactions.

The main goal of this system is to enable effective communication between both deaf and non-deaf people. Exploiting the capabilities of Python, Keras, Deep learning, Open CV and Django, this system provides seamless communication. It will help to provide equal opportunities for all.

### Project background

The development of a sign language interpreter system arises from the recognition of communication challenges faced by the deaf and hard-of-hearing community. Sign language, a complex visual-gestural language, presents barriers for non-sign language users. This project utilizes deep learning techniques, specifically Python's Keras library, to convert sign language gestures into real-time textual representations. By doing so, it aims to foster effective communication between deaf and non-deaf individuals.

While previous research has paved the way for sign language interpreter systems, there is a need for robust solutions that handle various sign languages, adapt to different signing styles, and operate in real-time. The project addresses these needs by leveraging deep neural networks, efficient data processing techniques, and integrating the Django framework for a user-friendly web-based interface.

The project's goals include developing a reliable and accurate sign language interpreter system, capable of real-time interpretation, to promote inclusivity and overcome communication barriers. By harnessing the power of deep learning, the system empowers the deaf community to participate fully in education, employment, healthcare, and social interactions.

Through this project, the aim is to contribute to the advancement of sign language interpreter systems, bridging the gap between sign language users and non-sign language users. By providing an accessible means of communication, the system promotes understanding, inclusivity, and equal opportunities for all individuals, irrespective of hearing abilities.

### Project Aims and Objectives

#### Aims:

The main aims of the project are mentioned below:

* Develop a sign language interpreter system for effective communication between sign language users and non-sign language users.
* Promote inclusivity by breaking down communication barriers faced by the deaf and hard-of-hearing community.
* Improve accessibility and participation in various domains, including education, employment, healthcare, and social interactions.
* Leverage deep learning techniques and Python's Keras library to achieve accurate conversion of sign language gestures into textual representations.
* Enhance usability and user experience through the integration of a user-friendly web-based interface using the Django framework.

#### Objectives:

* The main objective of the project is mentioned below:
* Train deep learning models to recognize and interpret sign language gestures accurately.
* Develop algorithms and techniques to handle various sign languages and adapt to different signing styles.
* Implement real-time processing to enable seamless and immediate translation of sign language gestures.
* Integrate Python's Keras library to facilitate efficient deep learning model training and inference.
* Create a web-based interface using the Django framework to provide easy access and usability for users.
* Evaluate and refine the system to improve accuracy, reliability, and performance.
* Validate the system's effectiveness through user feedback and testing in real-world scenarios.
* Contribute to the advancement of sign language interpreter systems by addressing limitations and exploring further enhancements.
* Raise awareness and advocate for the importance of inclusivity and accessibility for individuals with hearing impairments.

## Literature review

According to recent records there are around 700,000 to 900,000 deaf people in the world. Sign language is a visual and gestural language used by the deaf and people with hard of hearing to communicate with the world. To communicate they use hand movements, facial expression, and body language to make others understand what they are saying. Sign language interpreters are basically translators for both who know sign language and who don’t. The main motive of this system is the same as them as it helps a person to change sign language to text and text to sign language.

Sign language is very important specially to communicate with deaf people as it is one and only way to communicate with them. It helps to break down the barriers of language. It helps deaf people to participate in various social, educational, and professional settings. It has its own grammar, syntax, and vocabulary. Sign language goes beyond acquiring a means of communication as it is also encouraging cultural awareness and appreciation. It is deeply connected to the history of deaf communities.

This system uses a webcam to recognize gesture for American Sign Language (ASL) and convert them to text. It uses Convolutional Neural Network (CNN) algorithm, Keras, Open CV and Django as a frontend framework. The main aim of this project is to recognize the sign language gesture to capture in video and use images or videos to give an accurate output. It will help or be a bridge between both sign language users and others.

### Problem Domain Research:

In this phase, extensive research is conducted to understand the challenges and requirements of sign language interpretation systems. Existing literature, studies, and technologies related to sign language interpretation are explored to gain insights into the domain. The goal is to identify the limitations and shortcomings of current systems, which will inform the proposed solution and ensure that it addresses the specific needs of the users.

#### Information Gathering:

Information gathering involves actively engaging with domain experts, sign language users, and stakeholders to gather insights into their needs and expectations. Various methods such as interviews, surveys, and consultations are employed to understand the requirements of the sign language interpreter system. By directly involving the target users and stakeholders, the system can be designed to meet their specific needs and enhance usability.

#### Identification of Alternative Solutions from Existing Systems:

This phase involves investigating and analyzing existing sign language interpretation systems. Different approaches, technologies, and methodologies employed in those systems are identified. By evaluating the strengths and weaknesses of these alternatives, the project team can make informed decisions and leverage the best-suited techniques for the proposed system. This analysis ensures that the system is built upon existing knowledge and takes advantage of proven solutions.

#### Proposed Solution Strategy:

The proposed solution strategy outlines the overall approach, architecture, and design principles of the sign language interpreter system. It encompasses the methodologies, algorithms, and technologies that will be used to address the identified challenges. Factors such as accuracy, real-time performance, adaptability to different sign languages, and user-friendliness are considered to ensure an effective and efficient solution.

#### Functional Requirements:

Functional requirements define the specific functionalities and features that the sign language interpreter system should possess. This includes capturing and processing video input, recognizing sign language gestures accurately, and generating textual representations in real-time. Additionally, user interaction requirements, such as video uploading, playback, and text output display, are determined. Any additional modules or features needed to enhance the system's functionality are also specified, such as user authentication, error handling, or language support.

1.5. Research Methodology and Implementation

1.5.1. Data Collection:

The research methodology for the sign language interpreter system involves data collection to build and train the model. Various sign language datasets need to be collected, including images or videos of individuals performing sign language gestures. These datasets can be obtained from existing publicly available sources or through data collection from volunteers or sign language experts.

1.5.2. Data Processing:

Once the data is collected, it needs to be processed to prepare it for training the model. This involves preprocessing steps such as resizing images, normalizing pixel values, and potentially augmenting the dataset to increase its diversity and generalization capabilities. Additionally, any noise or background interference in the data may need to be removed or minimized through filtering techniques.

1.5.3. Model Training:

The processed data is then used to train the sign language interpreter model. Deep learning techniques, such as convolutional neural networks (CNNs), can be employed to train the model to recognize and interpret hand gestures. The training process involves feeding the model with labeled data and iteratively adjusting the model's parameters to minimize the loss and optimize performance.

1.5.4. Model Evaluation:

After training the model, it needs to be evaluated to assess its performance and accuracy. This involves using a separate set of data, called the validation or test set, to assess how well the model generalizes to unseen examples. Various evaluation metrics, such as accuracy, precision, recall, and F1 score, can be used to measure the model's performance.

1.5.5. Integration:

Once the model is trained and evaluated, it needs to be integrated into the sign language interpreter system. This involves connecting the model with the user interface, which can be developed using Django or other web development frameworks. The system should allow users to input sign language gestures through a webcam or other input devices, process the input, and display the corresponding text or audio output.

1.5.6. Deployment:

After integration, the sign language interpreter system is ready for deployment. It can be deployed on a local machine or hosted on a server to make it accessible to users. The deployment process may involve configuring the necessary infrastructure, ensuring compatibility with different operating systems and devices, and optimizing the system's performance for real-time usage.

1.5.7. Maintenance:

Once the system is deployed, ongoing maintenance is crucial to ensure its smooth operation and address any potential issues that arise. This may involve regular updates and enhancements to improve the model's accuracy and efficiency, addressing any user feedback or reported bugs, and staying up-to-date with advancements in deep learning, computer vision, and web technologies.

The research methodology outlined above provides a systematic approach to developing and implementing a sign language interpreter system. By following these steps, researchers can effectively collect and process data, train and evaluate the model, integrate it into a user-friendly interface, deploy the system, and maintain its functionality for long-term use.

* 1. Resources Requirements

Developing a sign language interpreter system requires various resources to ensure smooth implementation and operation. The following are the key resource requirements for building and maintaining the system:

1.8.1. Hardware:

- Computer: A powerful computer or server is necessary for training deep learning models, processing large datasets, and running the interpreter system.

- Webcam: A high-quality webcam is essential for capturing hand gestures in real-time for interpretation.

- Processing Power: Sufficient CPU and GPU capabilities are needed to handle the computational demands of deep learning algorithms and image processing tasks.

1.8.2. Software:

- Python: The system is built using Python programming language, so having Python installed is a prerequisite.

- Development Libraries: Essential libraries like Keras, TensorFlow, OpenCV, and Django need to be installed to facilitate model training, image processing, and web development.

- Deep Learning Framework: Installing a deep learning framework, such as Keras or PyTorch, is necessary for implementing and training deep learning models.

- Integrated Development Environment (IDE): An IDE like PyCharm or Jupyter Notebook provides a user-friendly environment for coding, debugging, and running the system.

1.8.3. Dataset:

- Sign Language Dataset: A comprehensive dataset containing labeled images or videos of sign language gestures is required for training and evaluating the model. This dataset can be obtained from publicly available sources or through data collection efforts.

1.8.4. Training and Testing Resources:

- Compute Resources: Sufficient computational resources, such as CPUs and GPUs, are necessary for training deep learning models on the sign language dataset. This can be achieved using local hardware or cloud-based services like Amazon Web Services (AWS) or Google Cloud Platform (GCP).

- Training Time: Training deep learning models can be computationally intensive and time-consuming. Sufficient time should be allocated for model training and optimization.

- Testing Set: A separate set of data should be reserved for testing and evaluating the trained model's performance. This set should contain unseen examples to assess the model's generalization capabilities accurately.

1.8.5. User Interface Development:

- Web Development Framework: Django or similar web development frameworks are required for creating the user interface of the sign language interpreter system.

- Front-end Technologies: HTML, CSS, and JavaScript are essential for designing and implementing the user interface components.

- Deployment Platform: A hosting platform or server is needed to deploy the web application, making it accessible to users.

1.8.6. Maintenance and Updates:

- Regular Updates: Resources should be allocated to maintain the system, keep it up-to-date with the latest versions of libraries and frameworks, and incorporate any bug fixes or performance improvements.

- User Feedback Mechanism: Establishing a mechanism to collect and address user feedback is crucial for continuously improving the system's functionality and user experience.

By ensuring the availability of these resources, researchers and developers can effectively build, deploy, and maintain the sign language interpreter system, enabling efficient communication for speech-impaired individuals.