Dermatology Research: SkinInsights ISSN: 3068-9384

PRIME OPEN ACCESS

**Volume 1, Issue 1**

**Research Article**

**Date of Submission:** 02 April, 2025
**Date of Acceptance:** 14 October, 2025
**Date of Publication:** 23 October, 2025

# Real-Time Lip Reading and Speech Synthesis Using CTC-CNN-BILSTM Networks with Flask Deployment

## Ahmed Cherif*

Orange Innovation Department Sofrecom, Tunisia

***Corresponding Author:**
Ahmed Cherif, Orange Innovation Department Sofrecom, Tunisia.

**Citation:** Cherif, A. (2025). Real-Time Lip Reading and Speech Synthesis Using CTC-CNN-BILSTM Networks with Flask Deployment. *Dermatol Res SkinInsights*, 1(1), 01-07.

## Abstract
This paper presents a novel approach to real-time lip reading to speech using a combination of Connectionist Temporal Classification (CTC), Convolutional Neural Networks (CNN), media pipe and Bidirectional Long Short-Term Memory (Bi- LSTM) networks, followed by a deployment strategy using Flask. The proposed system aims to transcribe spoken language from silent video sequences by leveraging the spatial and temporal features of lip movements. Extensive experiments on the GRID dataset demonstrate the effectiveness of the model, achieving 8.15% as Character Error Rate (CER) and 91.85% as Character Accuracy (CA). Additionally, we outline the deployment process, which enables real-time lip reading through a web application.procedures during medical education, simulation-based routines and periodic NR training are required to improve neonatal outcome.

**Keywords:** Long Short-Term Memory, Bidirectional, Con-Volutional, Temporal, Connectionist, Real-Time Deployment, CTC Loss, Camera Activation, Media Pipe, Facial landmarks, Region of Interest (ROI), Open Mouth Detection, Lip Movements, Image Resizing, Grayscale Conversion, Image Normalization, CTC Decoding, Word Segmentation, Word Ninja, Text Blob, Text Correction, GTTS (Google Text-to-Speech), Speech Synthesis, Real-Time Inter- Action

## Introduction
Lip reading, the process of interpreting speech by visually observing the movements of the lips, face, and tongue, has sig-nificant applications in various fields. Traditional approaches often fail to capture the complex spatiotemporal dynamics of lip movements. Recent advancements in deep learning, particularly CNNs and LSTMs, have shown great promise in modeling these dynamics. This paper proposes a CTC- CNN-BI-LSTM-based approach to lip reading, focusing on real-time. Additionally, we demonstrate the deployment of the model using Flask, allowing users to interact with the lip-reading system through a web application.

Our project focuses on developing an application leveraging advancements in deep learning to improve lip reading. It identifies and extracts lip movements in videos to optimize recognition. A key element is an intuitive and accessible user interface, facilitating the integration of our technology for a broad audience. Our goal is to enhance the accuracy and adaptability of lip reading, securing communication between people and enriching human-machine interactions. Some widely used datasets for lip reading include LRW1000, OuluVS2, AV Digits, Reading in the Wild, and GRID COR- PUS. These datasets consist of video recordings where lip movements are synchronized with audio speech, providing detailed transcriptions at the character level.

## Related Work
The field of automated lip reading has seen significant advancements with the integration of deep learning techniques. Early contributions include the development of the GRID dataset, which is extensively used for audio-visual speech recognition research and comprises video recordings of speakers reciting predefined sentences, making it an essential

resource for training and evaluating lip reading systems [1].The introduction of the Connectionist Temporal Classification (CTC) loss function marked a significant milestone, as it is designed for sequence prediction tasks where the alignment between input sequences and target labels is unknown, a scenario common in lip reading [2]. A deep learning approach for lip reading utilizing a combination of convolutional and recurrent neural networks demonstrated significant improvements over traditional methods when trained and evaluated on a large-scale dataset of sentence-level lip reading [3].

End-to end models like Lip Net, employing a sequence-to-sequence architecture with CTC loss, have set new benchmarks in the field, processing entire sentences in video sequences with state-of-the-art performance on the GRID dataset [4]. The combination of Residual Networks (Res Nets) and Long Short-Term Memory (LSTM) networks has also been explored, where Res Nets extract spatial features from video frames while LSTMs capture temporal dependencies, resulting in improved performance over standalone CNN or LSTM models [5]. Further comparisons of various deep learning models for lip reading have been conducted, providing insights into their strengths and limitations in real-world scenarios and introducing an online lip-reading application [6].

A multi-view lip reading system using a hybrid neural network architecturethat combines CNNs for feature extraction and LSTMs for sequence modeling demonstrated robust performance across different viewing angles and is designed for real-time applications [7]. For deployment, developing web applications using Flask, a lightweight web framework for Python, is valuable for deploying deep learning models as web services [8]. The combination of Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and Connectionist Temporal Classification (CTC) loss function has been instrumental in advancing lip reading technologies. For instance, the integration of 3D CNN with Bidirectional LSTM achieved a word error rate of 15.8% and a character error rate of 6.2% after 92 epochs [9]. Despite these advancements, challenges such as multi-language support, real-time performance, and robustness to varying video qualities remain areas for further research.

## Methodologies Used for The Model

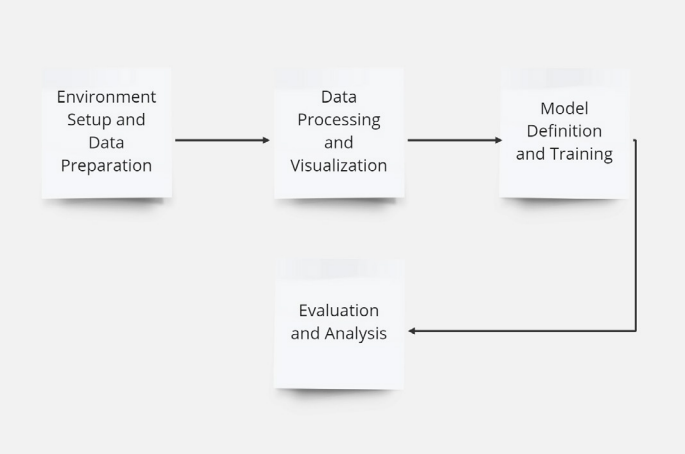


**Figure 1: Workflow of The Lip-Reading Model**

Environment Setup and Data Preparation To develop a robust lip-reading application, we start by setting up the necessary dependencies and configuring the video capture system. Essential Python libraries such as `OpenCV python`, `matplotlib`, `imageio` and `TensorFlow` are installed. `OpenCV-python` is utilized for video capture and processing, enabling us to handle video streams efficiently. `matplotlib` aids in data visualization, while `image` is employed for reading and writing image data and `TensorFlow` serves as the backbone for building and training deep learning models. Furthermore, we configure TensorFlow to utilize GPU acceleration, significantly enhancing computational performance by leveraging available GPU resources. This setup ensuresthat our system is well-prepared for handling real-time video input and processing, forming the foundation for effective lipreading analysis. The dataset used in this study consists of 1000 video sequences paired with their corresponding alignment files. Each video captures a speaker's facial movements, specifically focusing on the mouth area, crucial for lip-reading applications. The alignment files provide textual annotations synchronized with the video frames, mapping spoken words to their temporal segments. This dataset, termed the Grid dataset, combines visual and textual information, facilitating the training and evaluation of a deep learning model for accurate and robust lip-reading.

## Frame and Data Processing

Video data is processed frame-by-frame, where each frame is converted to grayscale and cropped into (140 x 46) pixels to focus on the region of interest, specifically the mouth area. The frames are then normalized by computing the mean and standard deviation, which aids in consistent and effective training of the neural network.

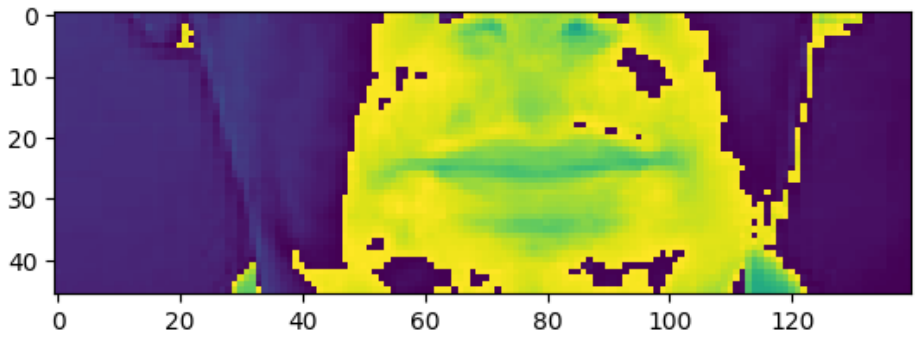


**Figure 2: Frame Preprocessing**

Frames in the video sequences are normalized to zero mean and unit variance before feeding into the neural network model. This normalization process helps in stabilizing the training process and improving convergence. Given a set of frames {Fi} from a video sequence:

- **μ is the Mean of the Frames:**

$$\mu = \frac{1}{N}\sum_{i=1}^{N} F_i$$

- **σ is the Standard Deviation:**

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(F_i - \mu)^2}$$

- **ˆ Fi Represents the Normalized Frame:**

$$\hat{F}_i = \frac{F_i - \mu}{\sigma}$$

This preprocessing step ensures that the input data has a consistent scale, which is crucial for effective training and inference in the deep learning model. To handle the accompanying text data, a vocabulary is constructed from the alignment files, and characters are mapped to numerical values. This transformation facilitates the training process by converting textual annotations into a format that the model can process. The text data is parsed to exclude non-essential elements like silence, and characters are encoded using the established vocabulary. For each video, frames and their corresponding alignments are loaded and paired. The data is then organized into a TensorFlow dataset, which is shuffled and batched for efficient processing. This dataset is split into training and testing sets to evaluate the model's performance. Throughout the process,

| Dataset | Percentage |
|---|---|
| Training | 90% |
| Testing | 10% |

**Table 1: Dataset Distribution**

TensorFlow's capabilities are leveraged to handle batching and prefetching, ensuring the pipeline is optimized for speed and efficiency.

## Model Training and Storage

We developed and trained the model over 50 epochs using TensorFlow and Keras, employing a robust architecture that combines :

- **Conv3D:** Performs 3D convolution operations to extract features from volumetric input data.
- **Activation:** Typically applies a non-linear activation function, such as RELU, to the outputs of convolutional layers.
- **MaxPooling3D:** Conducts spatial down sampling of extracted features to reduce dimensions and capture essential information.
- **Bidirectional LSTM:** Layers of Long Short-Term Memory (LSTM) recurrent neural networks capable of capturing temporal dependencies in data from both directions.
- **Dropout:** A layer that randomly deactivates neurons during training to mitigate overfitting.

- **Dense:** Fully-connected layers that combine extracted features for the classification task.
- **CTC Loss:** CTC loss computes the discrepancy between predicted sequences and ground truth labels by considering all possible alignments, enabling the model to learn effectively even when the lengths of sequences differ.

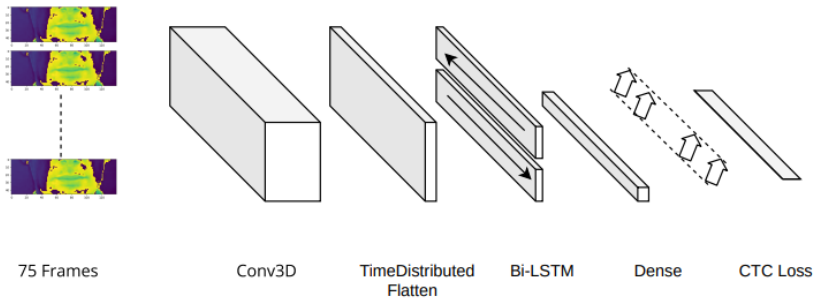


**Figure 3: Model Architecture**

The Connectionist Temporal Classification (CTC) loss function is a pivotal component in sequence prediction tasks, particularly in fields like speech and lip reading. It addresses the challenge where the alignment between input and output sequences may be variable or unknown. This approach is crucial in scenarios where precise temporal alignment is challenging to determine directly. By integrating CTC loss into the training process, models can effectively handle sequence data without relying on explicit alignment annotations, making it highly suitable for applications requiring robust handling of temporal dependencies and variable-length outputs. is defined as:

$$1 = -\frac{1}{B}\sum_{b=1}^{B}\log\left(\frac{\sum_{\pi\in\mathcal{B}^{-1}(y_{\text{tr}}^{(b)})}\prod_{t=1}^{T}P(y_{\text{pr}}^{(b)}(t,\pi(t))\mid x^{(b)})}{|\mathcal{B}^{-1}(y_{\text{tr}}^{(b)})|}\right)$$

Where:
- $B$: Batch size.
- $T$: Maximum sequence length.
- $x^{(b)}$: Input sequence for batch element b.
- $y_{\text{tr}}^{(b)}$ True label sequence for batch element b.
- $y_{\text{pr}}^{(b)}(t,\pi(t))$: Predicted probability or logits for the output at time t and alignment $\pi(t)$
- $\mathcal{B}^{-1}(y_{\text{tr}}^{(b)})$: Set off all possible alignments (path) of $y_{\text{tr}}^{(b)}$.

The CTC loss computes the negative log probability of thetrue labels given the predicted outputs, considering all possible alignments of the input and output sequences. This formulation allows the model to optimize for sequence prediction tasks where the alignment between input and output sequences is not explicitly provided, making it suitable for tasks like speech recognition and lip reading. The dimensions of input and output tensors are provided at each step, enabling tracking of data flow through the network. For example, input data has a shape of (None, 75, 46, 140, 1), where None represents the batch size, and the other dimensions represent 3D spatial dimensions and the number of channels.

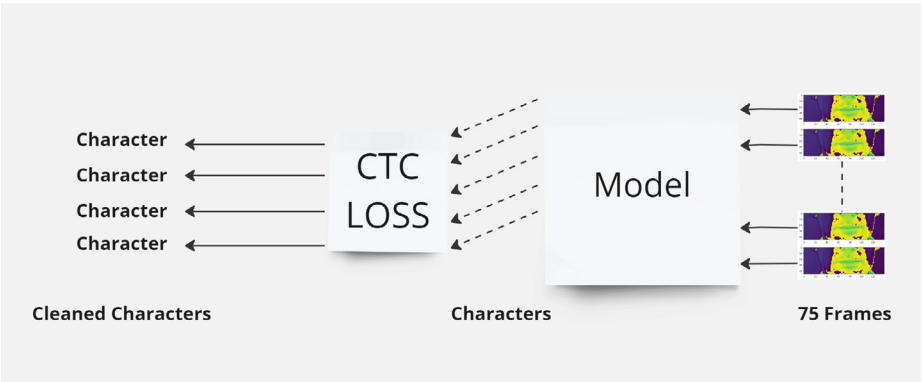


**Figure 4: Lip Reading CTC Loss Flow**

This type of 3D CNN architecture is commonly used to analyze volumetric data such as video sequences or other complex 3D data. This architecture was chosen for its ability to capture spatiotemporal features from lip movements, crucial for accurate speech decoding from video inputs. To optimize training efficiency, especially with large datasets, we implemented TensorFlow's MirroredStrategy, enabling distributed training across multiple GPUs in a synchronized manner. The model was fine-tuned using an Adam optimizer with a learning rate schedule managed through callbacks,

ensuring adaptive optimization throughout training epochs. Additionally, we employed a custom CTC (Connectionist Temporal Classification) loss function, tailored for sequence prediction tasks, which effectively measures the dissimilarity between predicted and ground truth sequences. Post-training, the model was saved inHDF5 format, facilitating seamless deployment and evaluation across different environments.

## Text Prediction from Video

In the realm of text prediction from video, the Character Error Rate (CER) serves as a fundamental measure for evaluating the accuracy and effectiveness of automated transcription systems. It quantifies discrepancies between predicted and actual text outputs, offering crucial insights into the reliability and performance of algorithms designed to convert spoken or visual information into text. Our investigation into Text Prediction from Video achieved a CER of 0.0815, derived from analysis across 14 test examples. This score reflects an accuracy rate of approximately 91.85%, highlighting areas where enhancements can be implemented to improve system precision. The Character Accuracy (CA), calculated as 1 − CER, stands at approximately 0.9185.

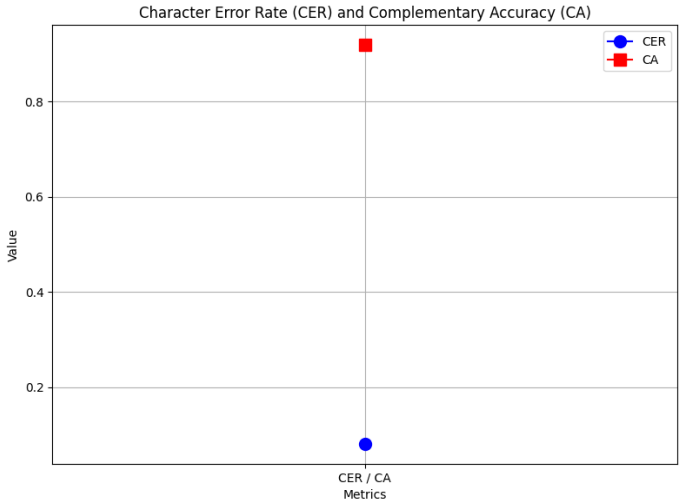


**Figure 5: Graphical Representation of Results**

## Real-Time Deployment

To deploy the lip-reading model in real-time, the following steps are executed: First, the pretrained prediction model and necessary configurations including vocabularies and character mappings are loaded. The CTC loss is then initialized. Next, the camera is activated to capture real-time images , configured with specific dimensions to optimize detection and processing. Media Pipe is employed to detect facial landmarks, focusing on the lips to determine the region of interest (ROI) [10].

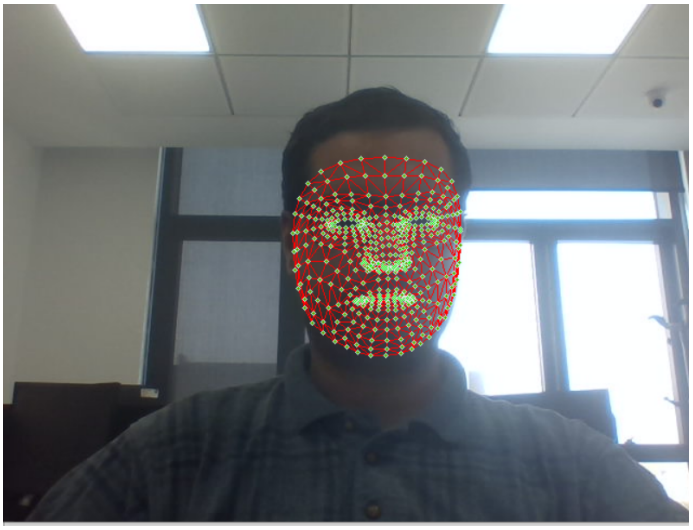

**Figure 6: The Face Landmarks**

A criterion detects an open mouth by evaluating the vertical distance between the upper and lower lips. The mouth is deemed open if this distance exceeds a predefined threshold H (where H = 0.05). Mathematically, this condition can be formulated as:

$$\text{Mouth Open} = \max_{i \in \text{LowerLip}} (L_i) - \min_{i \in \text{UpperLip}} (L_i) > H \quad (1)$$

- Li denotes the vertical coordinates of the lip landmarks.
- Lower Lip and Upper Lip refer to the sets of indices for the lower and upper lip landmarks, respectively.
- H is the predefined threshold.

Upon detection of an open mouth, the region containing the mouth is isolated in each image to focus exclusively on lip movements. Extracted mouth images are resized, converted to grayscale, and normalized to ensure consistent input to the prediction model, including mean subtraction and standard deviation division. A sequence of 75 images is collected over time to capture sufficient data for accurate lip movement analysis. These images are then fed into the prediction model, which utilizes them to predict text corresponding to lip movements using CTC decoding. The predicted text is segmented intodistinct words using algorithms like word ninja for meaningfulsegmentation. TextBlob corrects predicted words to enhanceaccuracy before displaying them in the interface. Finally, thecorrected text is synthesized into speech using GTTS (Google Text-to-Speech), enabling real-time interaction as long as the camera remains active [11].

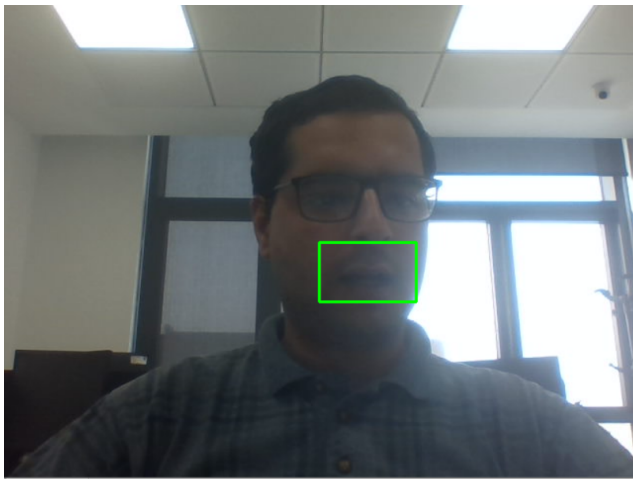

**Figure 7: The Mouth Region**

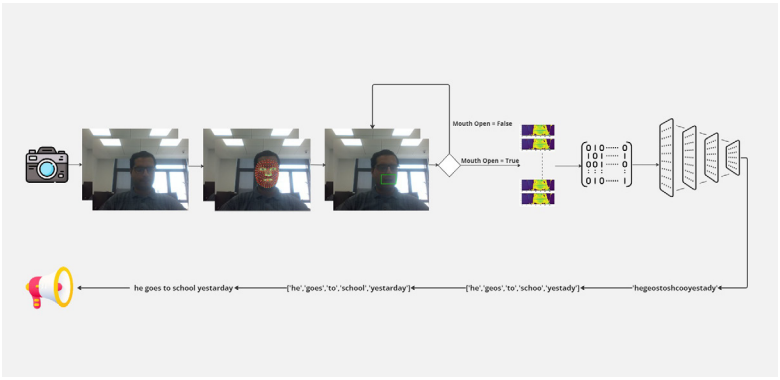


**Figure 8: Real-Time Lip Reading to Speech**

Figures 9 and 10 illustrate the interface of the application.

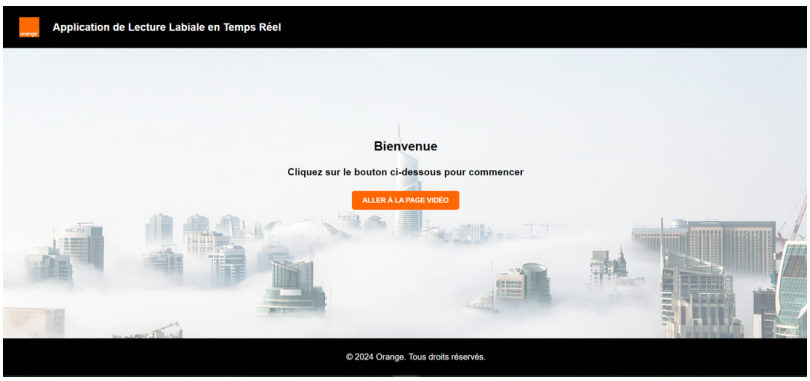


**Figure 9: The Welcome Page**

## Conclusion

In this paper, we introduced a robust framework for Realtime lip reading, integrating Connectionist Temporal Classification (CTC), Convolutional Neural Networks (CNN), Media Pipe, and Bidirectional Long Short-Term Memory (Bi-LSTM) networks. Our approach effectively captures and leverages the spatial and temporal features of lip movements to transcribe spoken language from silent video sequences. The performance of our model was thoroughly evaluated using the GRID dataset, yielding a Character Error Rate (CER) of 8.15% and a Character Accuracy (CA) of 91.85%, which underscore the system's accuracy and reliability.

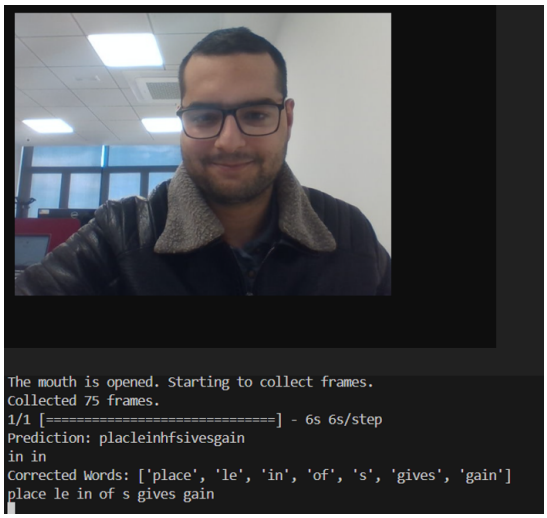


**Figure 10: The Lip-Reading Page**

Furthermore, we detailed the deployment strategy using Flask to facilitate real-time lip reading to speech through a web application, highlighting the system's practical applicability in dynamic environments. Our results demonstrate significant advancements in the field of automated lip reading, offering promising prospects for various applications including accessibility support, silent speech interfaces, and improved human computer interaction.

## Future Work

Future work will focus on enhancing model accuracy, expanding the dataset to include more diverse speech patterns, and refining the real-time deployment framework to accommodate broader usage scenarios. We believe that the proposed system marks a substantial step forward in real-time lip-reading technology, paving the way for further innovations in the domain.


## Acknowledgement

I would like to express my gratitude to Sofrecom Tunisia and the Orange Innovation Department for their invaluable support and assistance.


## References


1. Cooke, M., Barker, J., Cunningham, S., & Shao, X. (2006). An audio-visual corpus for speech perception and automatic speech recognition. The Journal of the Acoustical Society of America, 120(5), 2421-2424.
2. Graves, A., Fernández, S., Gomez, F., & Schmidhuber, J. (2006, June). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In Proceedings of the 23rd international conference on Machine learning (pp. 369-376).
3. Son Chung, J., Senior, A., Vinyals, O., & Zisserman, A. (2017). Lip reading sentences in the wild. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 6447-6456).
4. Assael, Y. M., Shillingford, B., Whiteson, S., & De Freitas, N. (2016). Lipnet: End-to-end sentence-level lipreading. arXiv preprint arXiv:1611.01599.
5. Stafylakis, T., & Tzimiropoulos, G. (2017). Combining residual networks with LSTMs for lipreading. arXiv preprint arXiv:1703.04105.
6. Afouras, T., Chung, J. S., & Zisserman, A. (2018). Deep lip reading: a comparison of models and an online application. arXiv preprint arXiv:1806.06053.
7. Petridis, S., Pantic, M., & Martinez, B. (2018). End-to-End Multi-View Lipreading. British Machine Vision Conference (BMVC), 1-12.
8. Grinberg, M. (2018). Flask web development. " O'Reilly Media, Inc.".
9. Shilaskar, S., & Iramani, H. (2024, March). CTC-CNN-Bidirectional LSTM based Lip Reading System. In 2024 International Conference on Emerging Smart Computing and Informatics (ESCI) (pp. 1-6). IEEE.
10. MediaPipe Cross-Platform Framework for ML-Based Multi-modal (Vision, Audio) Applied Research," Google, 2021.
11. Durette, P. N. (2022). gTTS (Google Text-to-Speech), a Python library and CLI tool to interface with Google Translate text-to-speech API.