Research Journal of Cell Sciences

PRIME OPEN ACCESS

Volume 1, Issue 1

Research Article

Date of Submission: 04 May, 2025
Date of Acceptance: 25 May, 2025
Date of Publication: 07 June, 2025

Lip Reading with Deep Learning: A Comprehensive Analysis of Model Architectures

Ahmed Cherif*

Orange Innovation Department, Tunisia

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

Citation: Cherif, A. (2025). Lip Reading with Deep Learning: A Comprehensive Analysis of Model Architectures. *Res J Cell Sci*, 1(1), 01-08.

Abstract
Lip reading, a pivotal skill in augmenting communication for the hearing impaired, has seen significant advancements with deep learning techniques. This study presents a comprehensive analysis of various deep learning model architectures for lip reading using a newly constructed dataset, DATAV1. Our investigation explores and evaluates multiple architectures, including ResBlock3D, Conv3D, Conv2D, TimeDistributed, attention mechanism and LSTM. Through extensive experimentation and rigorous evaluation metrics, we identify and discuss one of the optimal architectures for accurate lip-reading performance, achieving a peak validation accuracy of 98.18%. This research contributes insights into effective model selection and lays groundwork for further advancements in enhancing human-machine communication through lip reading systems.

Keywords: Lip Reading, Deep Learning, Conv3D, Time Distributed Layers, Attention Mechanisms, LSTM Networks, ResBlock3D, Batch Normalization, Model selection, Video Sequences, Validation Accuracy, Model architectures

Introduction
Lip reading, the art of deciphering spoken language from visual cues of lip movements, has long been a challenge for both human perception and automated systems. In recent years, the advent of deep learning has revolutionized the field, offering promising avenues for accurate and efficient lip-reading systems. These systems not only hold immense potential for aiding the hearing impaired but also find applications in noisy environments where audio-based communication is compromised. This paper presents a comprehensive analysis of various deep learning architectures tailored specifically for lip reading tasks. Our focus extends beyond mere model comparison; we delve into understanding the nuances of each architecture's performance. Central to our investigation is the training and the evaluation on a novel dataset, DATAV1, meticulously curated to reflect real-world challenges in lip reading. Through systematic experimentation and evaluation, we aim to provide insights into the effectiveness of different model configurations. The goal is to identify optimal architectures that not only achieve high accuracy in transcription but also exhibit scalability and practical feasibility in deployment scenarios.

Related Work
The field of lip reading has undergone significant evolution with the integration of deep learning techniques. Early approaches predominantly relied on handcrafted features and traditional machine learning algorithms, often encountering challenges such as variability in lighting conditions, speaker pose, and speech speed. The introduction of deep neural networks (DNNs), particularly convolutional neural networks (CNNs) and recurrent neural networks (RNNs), marked a transformative shift in the field. Early works by Wand et al. introduced CNNs for visual speech recognition, demonstrating their efficacy in capturing spatial dependencies within lip regions [1].

This seminal work laid the groundwork for subsequent innovations, including the pioneering LipNet by Chung and Zisserman which integrated CNNs with long short-term memory networks (LSTMs) for end-to-end sentence-level lip reading [2]. LipNet achieved stateof-the-art performance on standard benchmarks, underscoring the potential of deep learning in decoding visual speech cues.

Rekik et al. pioneered the use of Hidden Markov Models (HMMs) for lip reading, integrating both image and depth information [3].

Their Approach Involved a two-Step Process: First, estimating a 3D model of the speaker's face, followed by segmenting the speech video to identify meaningful utterances using the Viterbi algorithm. Subsequently, an HMM classifier was trained on these segmented features, achieving an overall accuracy of 65.9. In a subsequent work, Rekik et al. proposed a comprehensive four-step method [4]. Initially, they tracked the pose of the speaker's face, then extracted the mouth region and computed relevant features. Following this, a Support Vector Machine (SVM) classifier was employed, which first performed speaker recognition to tailor feature learning for individual speakers. Their method achieved notable success, reaching an overall accuracy of 71.15% on the MIRACL-VC1 Dataset. Attention mechanisms have further propelled the field by enabling models to selectively focus on pertinent frames and features during decoding [2,3]. This selective attention improves robustness against noise and enhances accuracy in challenging scenarios.

Recent advancements include the integration of 3D convolutional networks with attention mechanisms, facilitating both spatial and temporal modeling for enhanced lip-reading accuracy [4]. Furthermore, efforts in unsupervised and semi-supervised learning approaches have addressed the challenge of data scarcity by leveraging large-scale unlabeled datasets to improve model generalization [5,6]. These approaches have shown promise in learning discriminative features directly from raw video frames.

Methodologies Used

This section elaborates on the methodology employed in our lip-reading system, detailing each step in the workflow.

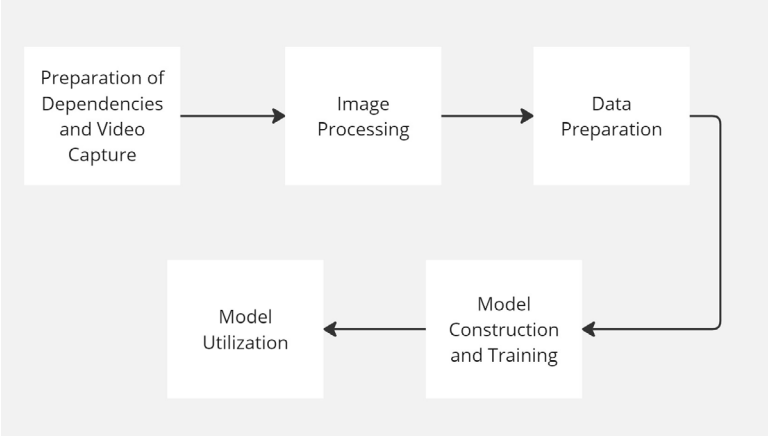

Figure 1: Workflow of the Lip-Reading Models

Preparation of Dependencies and Video Capture

In this subsection, we outline the initial setup required for our lip-reading system, focusing on the preparation of dependencies and the video capture process. Firstly, all necessary dependencies are imported to ensure that the system has access to the libraries and tools needed for video processing and model training. This includes importing deep learning frameworks, image processing libraries, and other essential packages. Once the dependencies are in place, we initialize the necessary objects for video capture and processing. This involves setting up the video capture device. The video capture process is then initiated, where the system begins recording the video frames that will be used for training and testing the lip-reading models. Proper initialization and setup of these components are crucial for maintaining the integrity and consistency of the data used in subsequent stages of the workflow.

Image Processing

In this subsection, we detail the comprehensive steps involved in processing the captured video frames, which are crucial for preparing the data for model training. The process begins with converting each frame to the RGB format using the OpenCV library, ensuring a standard color space for further processing. Subsequently, facial landmarks are detected using the MediaPipe library, specifically focusing on the upper and lower lip landmarks to identify open mouths. The detection function calculates the vertical distance between the upper and lower lips, determining if the mouth is open if this distance exceeds a predefined threshold (T).

$T = 0.03$

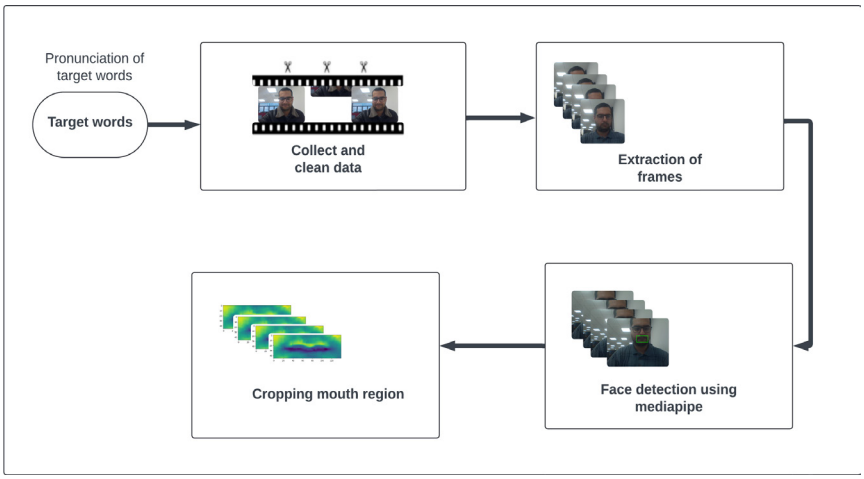

Figure 2: Data Preparation and Preprocessing Pipeline for Lip Reading

Mathematically, The Mouth is Considered OPEN If

$$\text{Mouth\_Open} = \max_{i \in \text{LowerLip}} (y_i) - \min_{i \in \text{UpperLip}} (y_i) > T \quad (1)$$

- y_i represents the vertical coordinates of the lip landmarks.
- LowerLip and UpperLip refer to the sets of indices for the lower and upper lip landmarks, respectively.
- T is the predefined threshold.

Upon identifying an open mouth, the region of interest (ROI) around the mouth is extracted from the frame. This involves calculating the bounding box coordinates for the mouth landmarks and cropping the mouth region from the frame. The extracted mouth region is then resized to a fixed dimension of 140×46 pixels and converted to grayscale. This conversion simplifies the data and reduces the computational load. The resized images are normalized to ensure consistent pixel value distribution across the dataset. The normalization process involves calculating the mean (μ) and standard deviation (σ) of the pixel values and adjusting each pixel value x using the formula.

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i \quad (2)$$

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)^2} \quad (3)$$

$$x' = \frac{x - \mu}{\sigma} \quad (4)$$

Where:

- x_i are the pixel values.
- N is the number of pixels.
- μ is the mean pixel value.
- σ is the standard deviation of the pixel values.

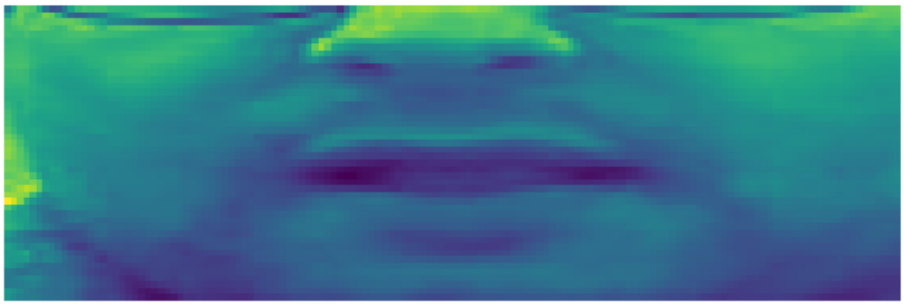

Figure 3: Normalized Frame

The normalized images, along with their corresponding labels, are added to lists for subsequent conversion into arrays. These arrays form the dataset required for training the lip-reading model. The 21 collected frames and labels are saved into .npy files, providing persistent storage for easy loading and manipulation during the training phase. Finally, the video capture is terminated, and the resources are released, ensuring no memory leaks occur. The dataset used for this project contains a total of 546 video clips, each labeled with one of ten target words.

Label Number	Word
0	bye
1	can you
2	demo
3	go
4	hello
5	no
6	read
7	stop
8	welcome
9	yes

Table 1: Label Mapping

These words represent common commands and phrases that are typically used in lip reading systems. The distribution of the labels is fairly balanced, as illustrated by the bar chart below

Dataset Preparation

To prepare the dataset for training the lip-reading model, we begin by loading and encoding the collected data. The normalized video frames and their corresponding labels are loaded from saved .npy files. We use a dictionary to map these numeric labels to their respective word representations, as shown in Table I. Additionally, a reverse dictionary is created to facilitate encoding and decoding operations. The labels are then encoded into numerical format suitable for model training using a reverse mapping dictionary. Next, the dataset is split into training and testing sets using the train_test_split function from scikit-learn, with a test size of 20% and a fixed random seed for reproducibility.

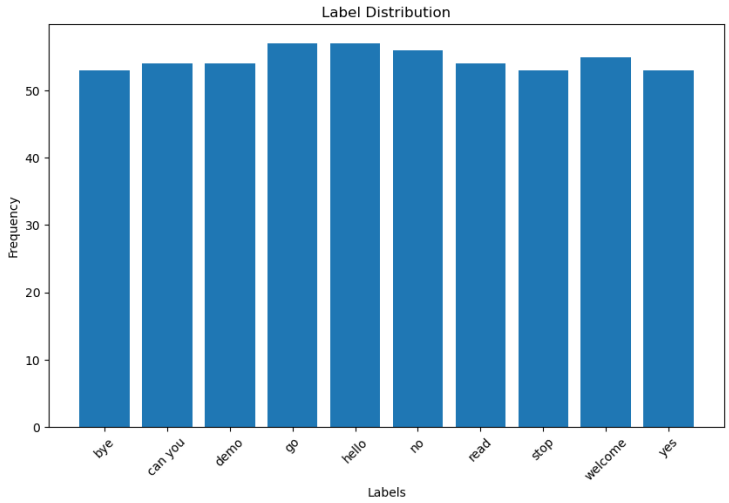

Figure 4: Label Distribution

Dataset	Percentage
Training Dataset	80%
Testing Dataset	20%

Table 2: Dataset Splits

To prepare the labels for model input, they are converted into one-hot encoding format using the two categorical function from Keras. This transformation ensures that the labels are represented as binary vectors, where each vector has a length equal to the number of unique labels (9 in this case), with a value of 1 indicating the presence of that label and 0 otherwise.

Encoded Labels: $y_{encoded}$ = {0,5,9,3,0,4,7,8,1, 2...}

$$\text{One-Hot Encoding: } y_{\text{onehot}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

In the table and equations above, $y_{encoded}$ represents the encoded labels mapped from the original word labels using the reverse dictionary, and y_{onehot} denotes the resulting onehot encoded labels used for model training. This structured approach ensures that the dataset is appropriately prepared and formatted for effective training and evaluation of the lip-reading model.

Model Construction and Training

In this section, we outline the construction and training of various models for lip reading using different architectures ResBlock3D + Conv3D, TimeDistributed + LSTM, TimeDistributed + Conv3D + Attention + LSTM, Conv3D + TimeDistributed + LSTM, and TimeDistributed + LSTM + Conv2D. Each model was compiled using the Adam optimizer and a softmax activation function for the output layer.

The Categorical Cross-Entropy Loss Function is Defined As

$$\text{Loss} = -\sum_{c=1}^{C} y_c \log(\hat{y}_c) \qquad (5)$$

- C is the number of classes.
- y_c is the true label (one-hot encoded).
- $\hat{y}_c$ is the predicted probability for class c.

The Adam optimizer updates the network weights θ iteratively based on gradients gt of the loss function L(θ)

$$\theta_{t+1} = \theta_t - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \qquad (6)$$

- η is the learning rate.
- $\hat{m}_t$ is the bias-corrected estimate of the first moment (mean) of the gradients.
- $\hat{v}_t$ is the bias-corrected estimate of the second moment (uncentered variance) of the gradients.
- ϵ is a small constant to prevent division by zero.

The First and Second Moment Estimates are Computed as Follows

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \qquad (7)$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \qquad (8)$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \qquad (9)$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \qquad (10)$$

- β_1 is the exponential decay rate for the first moment estimate.
- β_2 is the exponential decay rate for the second moment estimate.
- m_t is the first moment estimate at time t.
- v_t is the second moment estimate at time t.

- $\hat{m}_t$ is the bias-corrected first moment estimate.
- $\hat{v}_t$ is the bias-corrected second moment estimate.
- g_t is the gradient at time t.

The Softmax Function Computes the Probability Distribution Using the Formula

$$\hat{y}_c = \frac{e^{z_c}}{\sum_{k=1}^{C} e^{z_k}} \quad (11)$$

- $\hat{y}_c$ is the predicted probability for class c.
- z_c represents the logits (raw scores) for class c.
- C is the total number of classes.
- The denominator $\sum_{k=1}^{C} e^{z_k}$ normalizes the exponentiated logits to ensure the probabilities sum to 1.

Model Architecture	Epochs Trained
ResBlock3D + Conv3D	50
TimeDistributed + LSTM + Conv2D	50
TimeDistributed + Conv3D + Attention + LSTM + BatchNormalization	50
Conv3D + TimeDistributed + LSTM	100

Table 3: Model Training Details

For each model, the training process aimed to minimize the categorical cross-entropy loss function over 50 epochs, except for one model specifically trained for 100 epochs, with adjustments made using only the ReduceLROnPlateau without Early Stopping callback to dynamically adjust the learning rate based on validation loss. This training methodology was employed to optimize the models for accurate classification of lip-reading sequences.

Experimental Results

Model Performance Metrics

The experimental results evaluating various model architectures for lip reading are summarized in Table IV. Each model was trained and evaluated based on validation accuracy and loss metrics.

The results exhibit substantial variation in validation accuracy and loss across the evaluated architectures. The ResBlock3D + Conv3D architecture demonstrates the poorest performance, achieving a validation accuracy of only 13.64% with a high validation loss of 18.3506. These findings indicate that this configuration is less suitable for the lip-reading task.

Model Architecture	Validation Accuracy (%)	Validation Loss
ResBlock3D + Conv3D	13.64	18.3506
Conv3D + TimeDistributed + LSTM	83.64	0.4423
TimeDistributed + LSTM + Conv2D	95.45	0.4749
TimeDistributed + Conv3D + Attention + LSTM + BatchNormalization	98.18	0.0823

Table 4: Validation Metrics for Different Model Architectures

In contrast, the Conv3D + TimeDistributed + LSTM architecture achieves significantly improved results with a validation accuracy of 83.64% and a validation loss of 0.4423. This enhancement underscores the effectiveness of temporal layers in capturing critical temporal dynamics for lip reading. Further improving upon this, the Time Distributed +

LSTM + Conv2D model achieves a validation accuracy of 95.45% with a marginally higher validation loss of 0.4749. This architecture highlights the benefit of combining 2D convolutions with temporal processing to achieve competitive performance. The Time Distributed + Conv3D + Attention + LSTM + Batch Normalization architecture achieves the highest performance among the tested models, with a validation accuracy of 98.18% and a minimal validation loss of 0.0823. Attention mechanisms enable the model to focus on salient features within video sequences, while batch normalization aids in stabilizing and accelerating the learning process.

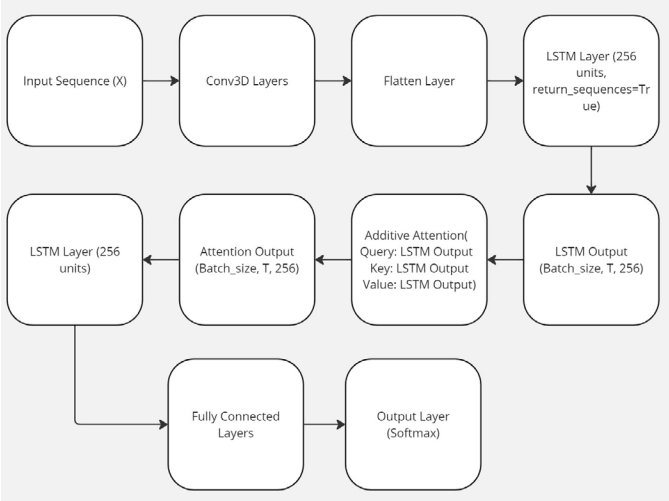

Figure 5: Neural Network Architecture with Additive Attention Mechanism

Graphical Representations

The figure 6 visualize the Training Accuracy Evolution. Additionally, Figure 7 presents the confusion matrix illustrating model performance.

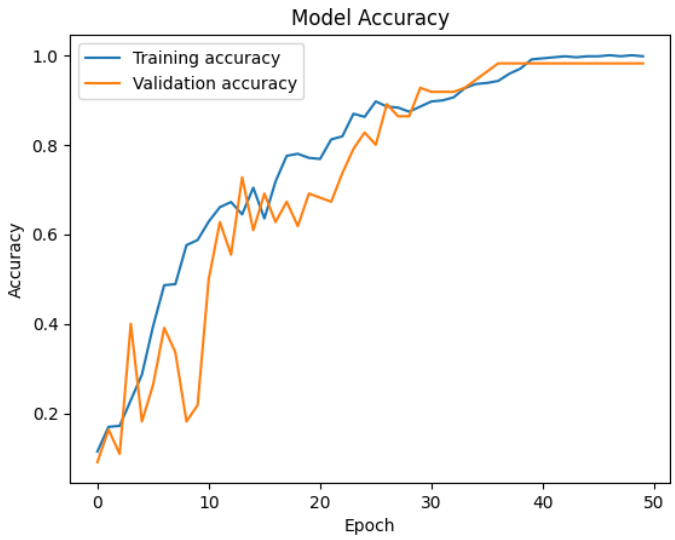

Figure 6: Training Accuracy Evolution

This paper presented an in-depth analysis of various deep learning architectures for lip reading using the newly constructed DATAV1 dataset. We evaluated models including ResBlock3D, Conv3D, Conv2D, TimeDistributed layers, attention mechanisms, and LSTM networks. Our experiments identified the TimeDistributed + Conv3D + Attention + LSTM + Batch Normalization architecture as the optimal model, achieving the highest validation accuracy of 98.18%. The success of this model theoretically highlights the importance of attention mechanisms and batch normalization in enhancing performance. These components allowed the model to focus on relevant features and stabilize the learning process, respectively. This research provides valuable insights for model selection in lip reading tasks and supports the development of advanced communication aids for the hearing impaired.

Future Works

Future work will focus on developing real-time lip-reading solutions to enable immediate communication for the hearing impaired. Optimizing models for faster inference without compromising accuracy is a key goal. Expanding the dataset and investigating different preprocessing methods will also be prioritized to improve model robustness and generalization.

These efforts aim to advance lip reading technology, making it more effective and accessible.

Acknowledgements
I am deeply thankful to Sofrecom Tunisia and the Orange Innovation Department for their support and assistance. [7,8].

References
1. M. Wand, J. Koutn´ık, and A. Schmidhuber," Lip reading with CNNs," in European Conference on Computer Vision (ECCV), 2016, pp. 472-488.
2. 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).
3. Rekik, A., Ben-Hamadou, A., & Mahdi, W. (2016). An adaptive approach for lip-reading using image and depth data. Multimedia Tools and Applications, 75, 8609-8636.
4. A. Rekik, A. M. Alimi, C. Ben Amar, and A. Ben Hamadou," A fourstep method for lip reading: tracking, mouth region extraction, feature extraction, and SVM classification," Pattern Recognition Letters, vol. 88, pp. 23-30, 2017.
5. Ephrat, A., Mosseri, I., Lang, O., Dekel, T., Wilson, K., Hassidim, A., ... & Rubinstein, M. (2018). Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation. arXiv preprint arXiv:1804.03619.
6. Chung, J. S., Nagrani, A., & Zisserman, A. (2018). Voxceleb2: Deep speaker recognition. arXiv preprint arXiv:1806.05622.
7. Afouras, T., Chung, J. S., Senior, A., Vinyals, O., & Zisserman, A. (2018). Deep audio-visual speech recognition. IEEE transactions on pattern analysis and machine intelligence, 44(12), 8717-8727.
8. T. Afouras, J. S. Chung, and A. Zisserman," Lip reading in the wild using unsupervised learning," in International Conference on Computer Vision (ICCV), 2019, pp. 5207-5216.