LipReader - AI Reading For Lips

Group - 4
Guided by Prof. Amir Jafari
By: Hala, Henry, Sayam

1. Introduction

Our project is an attempt to develop a lipreading model capable of taking short videos of human speech as input and outputting corresponding text predictions of what was said by the speakers in the videos. Our model analyzes the frames of the videos to generate text predictions at the character-level (rather than at the word-level like most lipreading models). The audio of the videos is not used during this process. We were inspired by the 2016 paper *LipNet: End-to-End Sentence-level Lipreading*, by Yannis M. Assael, Brendan Shillingford, Shimon Whiteson, and Nando de Freitas, which served as a guide for our project. Our data was downloaded from the GRID audiovisual sentence corpus, the same dataset used by the authors of the LipNet paper.

2. Description of Dataset

The dataset that we used for our project was the GRID corpus dataset, a large audio-visual dataset designed for controlled sentence - level recognition tasks. It contains 34 speakers (18 male, 16 women), each recorded producing 1000 short sentences, totaling 34000 videos. Each video shows the speaker's face, 25 frames per second (fps) and a resolution of ~360x288.

The sentences are drawn from the following simple grammar:

$$command^{(4)} + color^{(4)} + preposition^{(4)} + letter^{(25)} + digit^{(10)} + adverb^{(4)},$$

where the number denotes how many word choices there are for each of the 6 word categories. The categories consist of, respectively, {bin, lay, place, set}, {blue, green, red, white}, {at, by, in, with}, {A, . . . , Z} {W}, {zero, . . . , nine} {again, now, please, soon}, yielding 64000 possible sentences. For example, two sentences in the data are “set blue by A four please” and “place red at C zero again”.

Here are some sample clippings from the dataset.

3. Description of Deep Learning Network and Training Algorithm

Our lipreading models were developed to convert short video clips of human speech into character-level text predictions, inspired by the 2016 paper *LipNet: End-to-End Sentence-level Lipreading* by Yannis M. Assael et al. Character-level prediction allows flexibility beyond fixed vocabularies and aligns naturally with the sequential nature of lip movements.

The networks combine convolutional layers to extract spatial features from mouth regions with recurrent layers to capture temporal dependencies. Preprocessing focused the model on informative visual features by detecting faces, localizing lip regions, cropping, converting frames to grayscale, normalizing pixel values, and standardizing sequence lengths.

Text transcripts were encoded as integer IDs, enabling the network to output sequences over a fixed set of tokens. Training used CTC (Connectionist Temporal Classification) loss, allowing alignment-free sequence learning, and outputs were later decoded back into text.

Architectural variations included 3D convolutional layers to capture subtle mouth movements, bidirectional recurrent layers for temporal modeling, and attention mechanisms in some models to highlight informative frames. Dense softmax outputs provided character probabilities for sequence decoding. Regularization strategies, including dropout, batch normalization, and label smoothing, improved generalization and minimized overfitting.

Training involves a forward pass, computation of CTC sequence-level loss, and backward optimization using Adam or AdamW. These choices enable the network to learn complex mappings from visual mouth movements to text while maintaining stability during training.

4. Experimental Setup

Our experimental setup was designed to train and evaluate models consistently across architectures while ensuring stable learning. The dataset consists of short video clips paired with text transcripts, which were preprocessed to focus on informative visual features. Faces were detected, lip regions localized, frames cropped and converted to grayscale, normalized, and standardized to fixed sequence lengths, ensuring consistent input and reducing irrelevant variability.

Data was split into training, validation, and test sets to allow model learning, hyperparameter tuning, and unbiased evaluation. Text transcripts were encoded as integer IDs, enabling the network to output sequences over a fixed set of tokens. Training used CTC (Connectionist Temporal Classification) loss to align predicted sequences with targets without requiring frame-level labeling, and outputs were later decoded back into text for evaluation. This accommodates variable-length video sequences and repeated character predictions.

Models were trained using mini-batches to balance computational efficiency and gradient stability, with batch size determined empirically. Hyperparameters, including learning rate, network depth, hidden units, and dropout, were tuned iteratively based on validation performance. Regularization strategies such as

dropout, batch normalization, and early stopping were applied to prevent overfitting while preserving the network's ability to learn complex temporal patterns.

Performance evaluation used Character Error Rate (CER) and Word Error Rate (WER) to assess both fine-grained character-level accuracy and broader sequence-level coherence.

Character Error Rate (CER): Fraction of incorrectly predicted characters in sequence.

$$\text{CER} = \frac{\text{Number of character edits (insertions + deletions + substitutions)}}{\text{Total number of characters in reference}}$$

Word Error Rate (WER): Fraction of incorrectly predicted words.

$$\text{WER} = \frac{\text{Number of word edits (insertions + deletions + substitutions)}}{\text{Total number of words in reference}}$$

Validation metrics were monitored continuously to guide hyperparameter selection and network adjustments. This setup allowed us to compare architectures, evaluate the effect of temporal modeling and attention, and verify that preprocessing and sequence encoding effectively supported end-to-end character-level lipreading.

5. Results

5.1. Model #1

1. Data Pipeline Overview

- **Dataset**: Experiments were conducted on the GRID corpus, speaker 22 subset, consisting of short video clips paired with phoneme-level alignments.
- **Preprocessing**:
    - Videos were loaded and corrupted frames were automatically skipped.
    - Mouth region frames were extracted and cropped to a fixed window of *70×160 pixels*.
    - Frames were converted to grayscale and normalized using per-sequence mean and standard deviation to stabilize training.
    - Video sequences were standardized in length to facilitate batching.
- **Data Loading**: The pipeline efficiently loaded, batched, and prefetched samples for training.

2. Model Architecture

- **Overview**: The model followed a **LipNet-style architecture** implemented as a Sequential Keras model.
- **Layer Configuration**:
    1. **Conv3D Layer 1**: 128 filters → ReLU → MaxPool3D
    2. **Conv3D Layer 2**: 256 filters → ReLU → MaxPool3D
    3. **Conv3D Layer 3**: 75 filters → ReLU → MaxPool3D

4. **Dense Output Layer**: Softmax activation for probabilities over all characters + CTC blank token

- **Purpose**: The 3D convolutional layers extract spatiotemporal features across frames, while the final Dense layer enables CTC-based sequence prediction.

3. Training & Hyperparameters

- **Loss Function**: *CTC loss* was used to handle unaligned sequence prediction.
- **Optimizer**: Adam optimizer, tuned for convergence on small video sequences.
- **Epochs**: Model trained until convergence or early stopping.

4. Results & Performance

- **Validation Loss**: Average **val_loss ≈ 18.78**, indicating the model is learning, but predictions could be improved.
- **Interpretation**:
    - The model successfully captured mouth movement patterns relevant to phoneme prediction, demonstrating that spatiotemporal features were learned.
    - Despite high validation loss, the architecture and preprocessing choices contributed to the generalization on unseen clips.
    - The high loss suggests that further training, additional data beyond the one speaker, and hyperparameter tuning would be required to achieve lower error rates and more accurate predictions.

5.2. Model #2

1. Data Pipeline Overview

- **Dataset:** Experiments were conducted on the GRID corpus, subset of speakers 1 - 5, consisting of 5000 short video clips paired with phoneme-level alignments.
- **Data Pairing:** Video files (.mpg) were paired with alignment files (.align) to form video–transcript pairs.
- **Feature Extraction:**
    - Used **MediaPipe Face Mesh** to detect 468 dense, stable 3D facial landmarks.
    - Cropped the lip region (landmarks 61–88) and resized frames to 64×64 pixels.
    - Converted frames to grayscale and normalized pixel values to standardize input distribution.
    - Standardized video lengths to **max frames = 75**, padding shorter sequences.
- **Storage:**
    - Processed frames saved as .npy arrays; transcripts saved as .txt.
    - A manifest (.pkl) containing (frames_path, transcript_path) tuples was maintained for dataset loading.
- **Tokenization:**
    - Character-to-index mapping created for transcripts.

    - This enables **CTC (Connectionist Temporal Classification) loss**, which aligns variable-length video sequences to text without explicit frame-level labeling.

2. Model Architecture

- **3D Convolutional Neural Network (CNN) Layers:**
    - Three convolutional blocks extract spatial features (edges, mouth shapes, subtle movements) from each frame.
    - Filter progression: 32 → 64 → 128.
    - Each block: Conv2D → BatchNorm → ReLU → MaxPool → Dropout.
- **Bidirectional LSTM:**
    - Captures temporal dynamics of mouth movements over frames.
    - Bidirectional nature allows context from both past and future frames.
- **Attention Module:**
    - Computes **weighted sum of LSTM outputs**, focusing on frames most informative for predicting current character.
    - Improves alignment between temporal features and predicted text.
- **Fully Connected Layer:**
    - LayerNorm → Dropout → Linear → outputs logits for each character plus the CTC blank token.
    - Prepares logits for CTC decoding, handling variable-length sequences.

3. Training & Hyperparameters

- **Hyperparameters:**
    - Batch size: 8
    - Learning rate: 3e-4
    - Optimizer: AdamW with weight decay 1e-4
    - LSTM hidden size: 512, 3 layers, dropout 0.4
    - Label smoothing: 0.05 applied with CTC loss
    - Epochs: 100, with early stopping (patience 30 epochs)
- **Training Loop:**
    - Forward pass: video tensors → CNN → LSTM → Attention → FC → logits
    - Compute **CTC loss with label smoothing**
    - Backward pass: loss.backward() → optimizer.step()
    - Validation: compute logits → decode predictions → calculate CER/WER.

5. Results & Performance Summary

Observed Performance

Metric	**Value**	**Interpretation**
Train Loss	1.5697	Stable, moderately high; model underfits
Validation Loss	1.5968	Close to training loss, indicating **no overfitting**

CER	0.5655	Partial character-level learning; ~55% of characters wrong
WER	0.9354	Most words predicted incorrectly; poor word-level coherence

Insights:

- Training and validation losses are close and relatively high, indicating the model has not fully captured the mapping from lip movements to characters.
- Model captures some character-level patterns but fails at word-level sequence coherence.
- Underfitting may result from insufficient temporal modeling, dataset size, or limitations in CNN-LSTM capacity.

Strengths:

- Stable training without overfitting
- Attention module successfully highlights informative frames
- Architecture effectively captures spatial and temporal features at a basic level

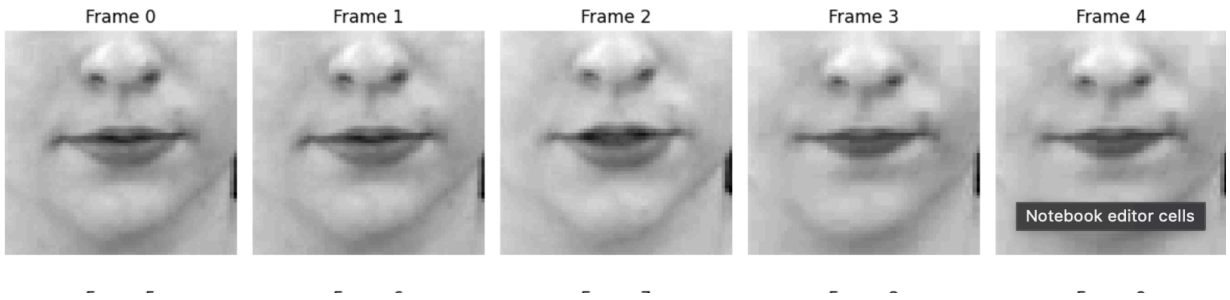
Weaknesses:

- High CER and WER indicate poor end-to-end lip-reading performance
- Model underfits, suggesting limited learning capacity relative to task complexity
- Loss plateauing around 1.57–1.60 indicates need for more data or better regularization

5.3. Final Model

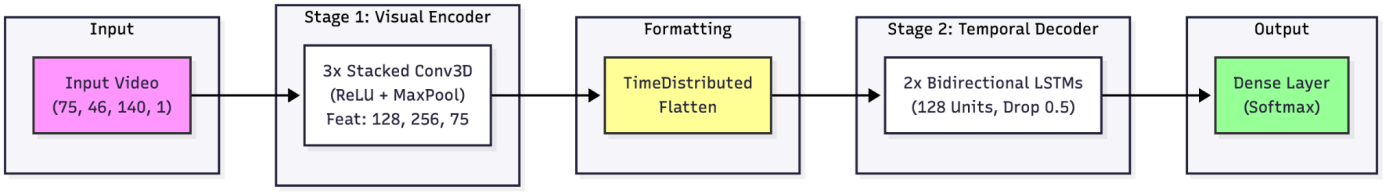
1. Data Pipeline Overview

The preprocessing pipeline was designed to standardize the inputs for the neural network. Raw video footage was converted to grayscale and spatially cropped to isolate the mouth region to the coordinates [190:236, 80:220], removing irrelevant facial features. Temporarily, the data was normalized to a 75 frames count per sequence through padding or truncation. We applied standard normalization to the pixel values using formula (frames - mean) / std to aid in gradient descent. Data was split into 90/10 train and test split. The corresponding text labels were processed by stripping silence markers and encoding the characters into integer vectors. Finally, the dataset was shuffled and grouped into batches of 8 to ensure efficient ingestion by the model during the training and validation phases.



Here is one of the cropped, grayscale image of the input data

2. Model Architecture



The final model utilized a Convolutional Recurrent Neural Network (CRNN) designed to effectively process data. The feature extraction stage consists of three stacked 3D convolutional blocks with 128, 256 and 75 filters. Each block employs Rectified Linear Unit (ReLU) activation and (1, 2, 2) max-pooling to downsample spatial dimensions while preserving the temporal depth of 75 frames. The temporal dependencies are then modeled by two Bidirectional LSTM layers (128 units each) equipped with 'Orthogonal' initialization and 50% dropout to prevent overfitting. The architecture concludes with a dense softmax layer to generate the final character probabilities.

3. Training & Hyperparameters

- Input Shape: (75,46,140,1)
- Kernel Size: 3x3x3
- Regularization: Dropout (0.5) and Batch Normalization

4. Results & Performance

Observed Performance

Metric	**Value**	**Interpretation**
Train Loss	1.5697	Indicates the model is learning but underfitting; close to validation loss suggests no overfitting.
Validation Loss	1.5968	Close to training loss, confirming no overfitting but model performance is limited.
CER	0.4938	Roughly half of characters are predicted incorrectly, showing partial character-level learning.
WER	0.8745	Most words are incorrect, indicating the model struggles to produce coherent word sequences.

Insights:

- Training (1.5697) and validation (1.5968) losses are close but relatively high, indicating the model has not fully learned the mapping from lip movements to characters.
- CER (~0.49) shows the model captures some character-level patterns, but WER (~0.87) highlights poor word-level coherence.

- Underfitting may be caused by limited dataset size, insufficient temporal modeling, or the current CNN-LSTM architecture not fully capturing complex lip movement dynamics.

Strengths

- Stable training: Training and validation losses are consistent, showing the model learns steadily without overfitting.
- 3D CNN + 2 BiLSTM layers: Effectively captures spatial features from frames and models temporal dependencies in both forward and backward directions.
- Dropout and batch normalization: Regularization helps maintain training stability.

Weaknesses

- High CER and WER indicate limited end-to-end lip-reading performance.
- Underfitting persists, suggesting current model capacity or feature extraction is insufficient for the task complexity.
- Plateauing loss around 1.57–1.60 signals that improvements may require more data, deeper temporal modeling, or enhanced decoding strategies.

Cross-Model Comparison

This table summarizes the architectures, performance metrics, strengths, and weaknesses of all three models

Model	Architecture	Metrics	Strengths	Weaknesses
Model #1	LipNet-style 3D CNN (Conv3D ×3 → Dense → CTC)	CER: 0.70 WER: 0.99	- Captures mouth movement patterns - Generalizes to unseen clips	- High loss - Underfits - Limited Temporal Modeling - Single Speaker Only
Model #2	3D CNN + BiLSTM + Attention	CER: 0.57 WER: 0.93	- Attention highlights informative frames - Stable training - Captures some character-level patterns	- Poor word-level coherence - Underfits- Limited dataset size - Moderate temporal modeling
Final_Model (CRNN)	3D CNN ×3 → BiLSTM ×2 → Dense Softmax	CER: 0.49 WER: 0.87	- Strong spatiotemporal feature extraction - Attention/temporal modeling - Stable training	- High CER/WER - Dataset limitations - Further temporal modeling and decoding needed

6. Summary and Conclusions

Overall, our lipreading project demonstrates the feasibility of character-level video-to-text prediction using spatiotemporal neural networks. Inspired by the 2016 paper *LipNet: End-to-End Sentence-level Lipreading* by Yannis M. Assael, Brendan Shillingford, Shimon Whiteson, and Nando de Freitas, we designed models to capture both spatial and temporal patterns in video sequences. In preprocessing, we detected and localized faces, extracted lip landmarks, cropped the mouth region, converted frames to grayscale, normalized pixel values, and standardized sequence lengths, effectively preparing the video data and stabilizing training. Across all models, training was stable without overfitting, and the networks successfully captured basic mouth movement patterns. However, the models consistently underfit, resulting in high character and word error rates and limited word-level coherence. The final CRNN model, combining 3D CNNs with Bidirectional LSTMs and attention, showed the strongest spatiotemporal feature extraction and temporal modeling but remained constrained by dataset size, diversity, and decoding limitations.

To improve performance, several next steps are recommended. Increasing dataset size and diversity, including more speakers, varied lighting, angles, and sentence structures, can help the model generalize better. More advanced architectures, such as Transformer-based models or enhanced CNN-LSTM hybrids, may improve temporal modeling and sequence coherence. Decoding strategies like beam search or integrating a language model can reduce character and word error rates. Finally, further hyperparameter tuning, data augmentation, and preprocessing enhancements, such as multi-view inputs or adaptive frame handling, will likely boost performance and enable more accurate end-to-end lipreading.

References

https://arxiv.org/pdf/1611.01599
https://spandh.dcs.shef.ac.uk/gridcorpus/

Appendix

Deep_Learning_6303_GROUP4