SilentSpeak: A LipNet-style Visual Speech Recognition

Lipreading model, data preparation and training process (GRID corpus)

Garvit (235890406) **Pranav (225890248)** **Aadhisha Vaibhav**

October 28, 2025

Abstract

SilentSpeak is a lipreading (visual speech recognition) project inspired by the LipNet architecture. The model combines 3D convolutional layers to capture spatiotemporal features from mouth-region video frames with stacked bidirectional LSTM layers and a CTC-based decoder to obtain character-level transcripts without frame-level alignment. This report details dataset preparation (GRID corpus), preprocessing, model architecture, training configuration, evaluation metrics and strategies used to mitigate speaker overfitting.

Index Terms— Visual Speech Recognition, Lipreading, 3D Convolutional Neural Networks, Bidirectional LSTM, Connectionist Temporal Classification, GRID corpus

1 Introduction

Visual speech recognition (lipreading) aims to translate visual mouth movements into textual speech. SilentSpeak implements a LipNet-style architecture that captures short-term spatiotemporal patterns using 3D convolutions and models longer-range temporal structure using bidirectional LSTMs. Training uses Connectionist Temporal Classification (CTC) to avoid explicit frame-level labeling. This report documents the dataset, preprocessing pipeline, model, training strategy and measures to reduce speaker overfitting.

2 Objectives

The main objectives of SilentSpeak are:

- Develop a lipreading pipeline that maps short video clips of the mouth region to character-level transcripts.
- Use a LipNet-style architecture (Conv3D encoder + BiLSTM decoder) and CTC loss for alignment-free training.
- Prepare a speaker-disjoint training/validation/test split on the GRID corpus to evaluate generalization.
- Document preprocessing, training hyperparameters and overfitting mitigation strategies.

3 System Overview

SilentSpeak comprises three logical modules:

1. **Preprocessing module**: extracts fixed-length frame sequences, crops to mouth region, converts to grayscale and normalizes.
2. **Model module**: 3D Conv encoder followed by BiLSTM layers and a time-distributed dense output producing per-frame logits.
3. **Training & evaluation**: CTC loss for training, greedy/beam CTC decoding for inference; metrics include WER and CER.

4 Dataset and Preprocessing

4.1 GRID corpus (reference)

The GRID corpus serves as the reference dataset for this project:

- **Speakers:** 34
- **Utterances per speaker:** ≈1000
- **Total duration:** ≈34 hours

4.2 Preprocessing steps

Each video sample is processed as follows:

1. Extract **75 frames** per utterance (3 seconds at 25 fps).
2. Crop to the mouth region (target size: **140** × **46** pixels, width × height).
3. Convert frames to grayscale (single channel).
4. Normalize pixel values to the range $[0, 1]$.
5. If a clip has fewer than 75 frames, pad by repeating the final frame; if longer, temporally sample or center-crop to 75 frames.

4.3 Annotations and splits

Each sample record contains `video_path`, `transcript` (character-level) and `speaker_id`. To prevent speaker memorization, create **speaker-disjoint** splits (e.g., 70% train, 15% validation, 15% test by speaker).

5 Model Architecture

SilentSpeak is built around three blocks: Conv3D encoder, Bidirectional LSTM stack, and time-distributed dense output for CTC.

5.1 Conv3D encoder

- Three Conv3D layers, each with **128 filters**.
- Kernel size: $3 \times 3 \times 3$ (time, height, width).

- MaxPooling3D after each conv block (recommended pool size: $(1, 2, 2)$ to preserve temporal resolution while reducing spatial dims).
- Batch normalization and SpatialDropout3D used for stability and regularization.

5.2 Temporal modeling (BiLSTM)

- Two stacked Bidirectional LSTM layers, **128 units** per direction.
- Return sequences = True (required for CTC).
- Dropout and recurrent dropout (typical 0.2–0.3) to reduce overfitting.

5.3 Output and decoder

- TimeDistributed Dense layer projects recurrent outputs to **vocab-size** logits per time-step (characters + blank symbol).
- Training uses **CTC loss**; inference can use greedy or beam-search CTC decoding.

5.4 Textual architecture diagram

```
Input (75 x 46 x 140 x 1) ->
[Conv3D(128) + MaxPool(1,2,2)] x3 ->
TimeDistributed(Flatten/Dense) ->
BiLSTM(128) -> BiLSTM(128) ->
TimeDistributed(Dense(vocab_size)) -> logits -> CTC
```

6 Training Configuration

6.1 Loss: CTC

Connectionist Temporal Classification (CTC) is used because it allows training without frame-to-label alignment. It sums probabilities over possible alignments and computes a negative log-likelihood.

6.2 Optimizer and hyperparameters

Hyperparameter	Typical value
Optimizer	Adam
Learning rate	1×10^{-4}
Batch size	2–4 (GPU memory constrained)
Epochs	Up to 100 (with EarlyStopping)
Callbacks	ModelCheckpoint, ReduceLROnPlateau, EarlyStopping
Framework	TensorFlow / Keras
Hardware	NVIDIA GPU (e.g., GTX 1080 or better)
Estimated training time	$\sim$2–3 days (GPU, dataset dependent)

7 Evaluation Metrics

To quantify performance:

- **Character Error Rate (CER)**: edit distance at character level normalized by reference length.
- **Word Error Rate (WER)**: edit distance at word level normalized by reference word count.

8 Overfitting: Problem and Mitigation

8.1 Observed problem

A pre-trained LipNet-style model trained on some GRID speakers may overfit speaker-specific features (lip shape, lighting, viewpoint) and perform poorly on unseen speakers. This is the main practical challenge.

8.2 Mitigation strategies

- **Speaker-disjoint splits:** Ensure no speakers overlap between train/val/test.
- **Data augmentation:** temporal jittering, brightness/contrast variation, small spatial shifts, frame dropout/duplication.
- **Regularization:** SpatialDropout3D, L2 weight decay, dropout in LSTM layers.
- **Early stopping and LR scheduling:** Monitor WER/CER on validation speakers and use ReduceLROnPlateau/EarlyStopping.
- **Domain adaptation / adversarial training:** learn speaker-invariant features (advanced).
- **Speaker adaptation:** freeze encoder and fine-tune decoder on small speaker-specific data if available.

9 Extensions and Alternatives

- **Transformer encoder:** Replace BiLSTM with a Transformer to capture long-range dependencies using self-attention (requires more data/compute).
- **Attention-based seq2seq:** Use an encoder-decoder with attention rather than CTC (more flexible but more data hungry).
- **Multi-modal fusion:** Combine audio and visual streams for robustness when audio is available.

10 Practical Implementation Notes

- **CTC input lengths:** Ensure the `input_lengths` passed to CTC match the model's time-step outputs (adjust if time is downsampled).
- **Batch size:** Use small batches; use gradient accumulation if you need a larger effective batch size.

- **Checkpointing:** Save both training model (with CTC loss) and an inference model (produces logits).
- **Speaker evaluation:** Always evaluate on speaker-disjoint test set for realistic generalization numbers.

11 Conclusion

This report described the SilentSpeak LipNet-style visual speech recognition system: preprocessing for fixed-length mouth-region clips, a 3D convolutional encoder to capture spatiotemporal cues, stacked bidirectional LSTMs to model sequence dynamics, and a CTC-based training/decoding pipeline. Speaker overfitting is a primary concern when using GRID-like datasets; the recommended countermeasures include speaker-disjoint splits, augmentation, regularization, and careful validation using unseen speakers.