What Was That?: Lip-Reading Silent Videos

Akayla Hackson *
Department of Electrical Engineering
Stanford University
akayla@stanford.edu

Miguel Gerena *
Department of Computer Science
Stanford University
miguelg2@stanford.edu

Linyin Lyu *
Department of Computer Science
Stanford University
llyu@stanford.edu

Abstract

Lip reading, also known as visual speech recognition, involves transcribing text from videos. This is an ability to recognize what is being said from visual information alone. Lip reading is inherently ambiguous as different characters could produce exactly the same lip sequence (e.g. 'p' and 'b') and different people could say the same word but have different lip movements.

In this study, we developed three models aimed at advancing lip reading technology. The first model focused on being able to detect if the subject was speaking using a 3d-CNN without any addition for handling long-range temporal dependencies. This first model successfully predicts whether a person is speaking with an accuracy of 77.63%, serving as a strong foundation for our project. The second model used this CNN network to produce characters and forming the words being spoken. It handled short and long term dependencies utilizing an LSTM. It is focused on the LRW dataset and achieves a promising accuracy of 52.19%, indicating the potential to accurately predict spoken words without audio input. Our third model used the previous architectures with an Encoder-Decoder Transformer layer on top in order to handle the word dependencies and able to establish word and sentence structure. This third model, designed to predict entire spoken sentences, requires further improvement as it currently struggles with local minima issues. Although the first two models showed commendable performance, the complexities of sentence prediction necessitate continued experimentation. Due to time constraints, we were unable to fully address these challenges within this project, but we are committed to refining our approach and enhancing the model's performance in future work.

1. Introduction

Lip reading is a task to generate text transcriptions from videos. It can facilitate various applications, such as understanding surveillance videos when only visual signals are available, enabling video conferences in silent environments and transcribing archival silent films. Lip reading is challenging as it's influenced by angle, light, and background [5].

The team decided to pursue an end-to-end model to predict text from silent videos. This end-to-end model would ingest video frames and output the sentences being spoken in the video. Designing a lip reading model requires both a visual component to identify mouth movements and a temporal sequence modeling component, which typically involves learning a language model[22]. Most modern deep lip reading architecture can be divided into two parts: 1) the frontend part to capture spatiotemporal features and short-term dynamics of the mouth region [27]; 2) the backend part to emphasize on sequence-level patterns, learning the temporal dynamics of the sequence based on the features extracted by the frontend module. For our lip reading model, we choose spatiotemporal CNNs to process video frame and extract features. Spatiotemporal CNNs consisting of multiple 3D convolutional layers[2], where both spatial (height and width) and temporal (time) dimensions are considered. Spatiotemporal convolutional layers have been proven to effectively capture the short-term dynamics of the mouth region [27]. In the backend, we leverage the Bidirectional Long Short-Term Memory (Bi-LSTM) to capture the temporal dynamics and encoder-decoder transformer layers to produce the resulting sentence.

In this work, we performed three stages of studies: 1) predict if person in the video is speaking; 2) predict 1 word from the video; 3) predict the entire sentence. All the mod-

els use CNNs as their backbone with different heads on top. The 1 word (LRW) and speaking models proved the most successful as they did not have to handle the word dependency complexity.

2. Related Work

The lip reading system was first proposed in [20] by extracting visual features of the lip region and using Hidden Markov Models (HMMs) for classification. The strength of this approach lied in its simplicity and the interpretability of HMMs. However, it was limited by its reliance on handcrafted features and the inability to capture complex visual patterns. With the development of deep learning, instead of hand-crafted features, CNNs was commonly used to extract features from the lip region. LipNet[2] was the first end-to-end sentence-level model to predict character sequences for lip reading task. This model combined spatiotemporal convolutions with Long Short-Term Memory (LSTMs) and was trained using the CTC loss function. Our model leverage a similar concept. The difference is that we use a layer of 3D CNN instead of multiple 2D CNNs (STCNN used by LipNet). [27] prove that 3D achieve better performance than 2D in terms of capturing the short-term dynamics of the mouth region. One trend in recent work is moving to Transformer-based architectures [1], leveraging the self-attention mechanism to model long-range dependencies in video sequences.

2.1. Transformer

The Vision Transformer (ViT) [7] integrates a transformer architecture into visual recognition, leveraging large datasets. It divides each input image into non-overlapping patches, creating fixed-length tokens. These tokens are then processed by several standard transformer layers. The ViT model uses only transformers and achieves good performance, but it requires large datasets. The state-of-art model use AV-HuBERT, which is a self-supervised learning approach that integrates both audio and visual data for robust speech recognition. It builds on the HuBERT (Hidden-Unit BERT) model [24][25]. The strength of this model could be that it does not rely on CNNs for their backbone and the information is passed directly to the transformer as opposed to the VTP [21] model where CNN logits are passed to the transformer layers.

2.2. Temporal modeling

A temporal model is the next essential component after vision encoder in the lip reading pipeline. The two popular choices are Bi-GRU (or Bi-LSTM) or Temporal Convolutional Networks (TCN) [18]. TCN is a type of convolutional neural network designed for sequential data. TCN use 1D convolutions along the temporal dimension to capture dependencies over time. It's able to capture long-range dependencies efficiently. GRU or LSTM is more interpretable due to its hidden states. There are conflicting conclusion about their performance, which could be due to the limitations of 1D CNNs. MS-TCN perform better than GRU in [18] but not in [8].

3. Data

We utilized two distinct datasets and conducted three separate tasks. The first task involved the LRS2 dataset, where we aimed to predict whether a person was speaking or not. The second task employed the LRW dataset to predict the specific word a person was saying. The final task again utilized the LRS2 dataset to predict the entire sentence spoken by a person in the input video.

Unfortunately, due to the licensing of the data, we cannot include any examples. The clips were snippets of British television shows that focused and centered the speaker on the frame. Examples can be found in the Lip Reading in the Wild [4] and Lip Reading Sentences 2 [1] papers. For both datasets, we discretized the temporal dimension by the number of frames per second.

3.1. LRW

The Lip Reading in the Wild (LRW)[4] datasets consist of short 112x112 videos (1.16 seconds) from BBC program, mainly news and talk show. There are more than 1,000 speakers and a large variation in head pose and illumination and as a consequence is a challenging dataset. The number of words, 500. All videos are 29 frames in length, and the word occurs in the middle of the video. The training set consists of up to 1,000 occurrences per target word, while the validation and test sets both consist of 50 occurrences per word. It is worth mentioning that the target word is part of whole utterances rather than isolated.

3.2. LSR2

The Lip Reading Sentences 2 (LRS2-BBC) audio-visual datasets [1] consists of 224.1 hours with 144,482 video clips from BBC programs. Each video has 160×160 pixel resolution and 25 fps. The dataset is divided into development (train/val) and test sets according to broadcast date. The dataset also has a "pretrain" partition that includes extensive head tracks including word boundaries that have been produced by force-aligning subtitles to the audio. There are 96,318 utterances for pre-training (195 hours), 45,839 for training (28 hours), 1,082 for validation (0.6 hours), and 1,243 for testing (0.5 hours).

3.3. Data Preprocessing

The LRS2 dataset used for this research consists of short video clips, each of which paired with a corresponding transcript. Each video was processed to extract frames at a rate

of 25 frames per second. The extraction was performed on all mp4 files, ensuring a uniform frame rate across all samples, being that this consistent framing is crucial for the temporal alignment necessary in the subsequent stages of the model. The RGB frames were normalized to have a mean of 0 and a standard deviation of 1 per channel.

4. Methods

We performed three distinct tasks in our research. The first task focused on predicting whether a person was speaking. The second task aimed to predict the specific word a person was saying. The final task involved predicting the entire sentence spoken by a person in the video. All the models developed for these tasks utilized Pytorch [19] as their deep learning framework.

4.1. Data Loading

To accommodate the needs of batch processing during training, we developed a custom data loader. This loader efficiently retrieves data and handles the necessary padding and encoding for both video and text data. It can re-scale the images and add random horizontal flips for data augmentation. For video frames, padding ensures that all input sequences (frames) match the longest sequence in a batch, maintaining consistent tensor dimensions across batches. For the textual data, we utilized a pretrained tokenizer based on 'distilbert-base-uncased' to pad and encode the transcripts, which has a vocabulary of 30,522 tokens. This encoding converts text into a numerical format that is optimized for natural language understanding, utilizing the pre-existing knowledge embedded within the BERT architecture, which ultimately enhances the model's ability to process and interpret the text associated with the video frames [6].

4.2. Speaking model

The speak model architecture is composed of 11 3D CNN layers with residuals and Relu Activation in between the blocks. This architecture was inspired by a portion of the VTP model architecture [21] and it served as a baseline for building a new model layer by layer. The model ingests a video of dimension [1, 3, 125, 160, 160], where each dimension is the batch size, channels, number of frames, height, and width. The first layer of the model utilizes a kernel size of 5 with a stride of (1, 2, 2). Each subsequent filter is of size (1,3,3). Each layer utilizes the same stride as we want to keep the temporal dimension the same. For each layer where we want to use residuals, we keep the same number of input and output filters and change the stride to 1. The resulting logits after passing through all the 3D CNN layers are [1, 512, 125, 3, 3]. This serves as the backbone of the model and different permutations of linear layers on top were studied.

4.3. LRW model

See Figure 1 for a depiction of the full architecture.

4.3.1 Architecture

The LRW model, is designed for predicting a single word based on video input. The inputs to the network are tensors of shape [16, 3, 29, 96, 96], where each dimension corresponds to the batch size, channels, number of frames, height, and width, respectively. The video frames are then processed through a 3D convolutional neural network with 64 kernels of 5×7×7 size (frames/height/width), followed by Batch Normalization [12] and Rectified Linear Units (ReLU). After the 3D CNN, the shape becomes [16, 64, 29, 48, 48]. The extracted feature maps are then passed through a max pooling layer, which is used to downsample the feature maps, reducing their spatial dimensions while preserving the temporal aspect. The output from the max pooling layer is then fed into a Long Short-Term Memory (LSTM) sequence-to-sequence [3], which is adept at capturing temporal dependencies and maintaining a memory of past frame information. Finally, the sequential output from the LSTM is passed through a fully connected (FC) layer and softmax to produce the final output.

We also experiment using the model to predict single word as a classification task with the Cross-Entropy loss. In this case, the sequence output from LSTM is labeled based on the average feature representation of the entire sequence.

We begin with the simple model described previously. We then added ResNet after 3D CNN inspired by [27], which verified the effectiveness of ResNet.

4.3.2 CTC loss

$$p(Y|X) = \sum_{A \in A_{(X,Y)}} \prod_{t=1}^{T} p_t(a_t|X) \tag{1}$$

In our lip-reading project, implementing Connectionist Temporal Classification (CTC) loss offers great advantages. The CTC objective for a single X, Y pair is shown in Equation 1, where A is given by the Alignment vector and a_t is given by the alignment step. CTC loss provides alignment flexibility by considering all possible alignments between input sequences (video frames) and output sequences (text), making it particularly suitable for tasks with uncertain temporal correspondence. It handles variable-length output sequences effectively, which is crucial given the varying lengths of spoken sentences. Additionally, CTC loss reduces the effort needed for detailed frame-level labeling, as it allows training with only sequence-level labels.

4.3.3 Beam search

$$y_t^k = argsort P(y|y_1^k, \ldots, y_{t-1}^k) \tag{2}$$

To enhance the accuracy and coherence of our lip-reading model's predictions, we employed beam search decoding. Beam search, given by Equation 2, explores multiple candidate sequences simultaneously, maintaining a top-k set of the most likely hypotheses at each step. This approach improves accuracy by considering several possible outputs, helps handle ambiguities in lip movements, and balances exploration and exploitation. During inference, beam search generates a probability distribution over possible words and keeps the top-k sequences with the highest probabilities at each time step. When we reach time step T, we select the most likely sequence as the final prediction.

4.4. LRS2 model

See Figure 2 for a depiction of the full architecture.

4.4.1 Convolutional Neural Network (CNN)

Initially, the video frames are processed in batches through a CNN to extract features from each frame. We employed a ResNet-18 architecture, for its efficacy in feature representation [10]. However, we modified this architecture by removing the final classification layer, allowing us to use the network as a feature extractor rather than for classification. ResNet-18 is designed for the ImageNet Dataset with 1000 categories. We use the pretrained weights from ImageNet in our model.

In addition to the 2D CNN, we also experimented with the implementation of a 3D CNN architecture, instead of a 2D CNN, to capture spatiotemporal features from the video frames. This architecture included three main components: the first 3D convolutional layer with ReLU activation and max pooling, a second 3D convolutional layer also followed by ReLU activation and max pooling, and finally a global average pooling layer that reduces the spatial dimensions.

4.4.2 Long Short-Term Memory (LSTM)

The sequential output from the CNN then feeds into an LSTM network. The LSTM used is crucial for capturing temporal relationships between frames, an important concept with lip reading across the entirety of a video [11]. It processes the sequence of feature vectors, maintaining a memory of past frame information. Bidirectionality was added to the LSTM to be able to capture long-range dependencies before and after the current word in order to help with sentence structure.

4.4.3 Transformer Encoder/Decoder

The combined CNN and LSTM outputs provide a detailed representation of both the individual frames and their temporal dynamics. To enhance these representations, we added positional encodings to the LSTM output. This enriched output was then passed through an encoder. The encoder's output, along with the target sentence, was fed into a Transformer decoder. By using the target sentence as input to the decoder, we employed a popular method known as teacher forcing. The decoder's role is to generate the text corresponding to the lip movements depicted in the video frames. The Transformer architecture is particularly suited for this task due to its ability to handle sequences of data and its effectiveness in translation tasks [28].

4.4.4 GPT2 Decoder

We also experimented with the GPT2LMHeadModel as a decoder to generate text based on the input embeddings from LSTM. We experimented with this decoder to assess its potential to enhance our model's performance, leveraging the benefits of a pretrained language model. The GPT-2 decoder, with its robust language generation capabilities, provided an alternative approach to improve the accuracy and fluency of the predicted text.

4.4.5 Final Model Architecture

Our approach to solving the problem of predicting spoken text from video input is based on a comprehensive model architecture inspired by [21]. Our model begins by processing video frames through a 3D CNN to capture spatiotemporal features. These features are then passed to a bidirectional LSTM network to capture temporal dependencies and maintain memory of past frame information. We enhanced the LSTM output with positional encodings and passed it through a Transformer encoder.

The Transformer's output, along with the target sentence, is fed into a Transformer decoder, using teacher forcing to improve performance. This multi-layered approach allows us to effectively handle the complexity of lip reading, combining the strengths of convolutional, recurrent, and transformer networks.

We considered alternative approaches, such as using purely convolutional or recurrent networks, but found that the hybrid model incorporating transformers provided superior performance in handling the sequence-to-sequence nature of the task. The decision to use a 3D CNN, LSTM, and Transformer combination is justified by their respective strengths in feature extraction, temporal modeling, and sequence translation. Notably, we chose opt out of using the GPT2 decoder because it did not seem to improve performance, however, we may revisit the implementation in future work.

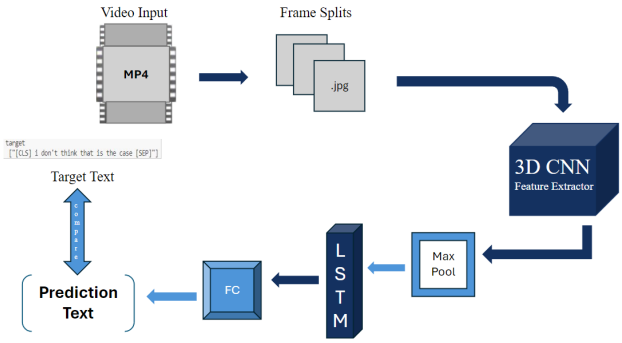

Figure 1. LRW model architecture. *Note: Light blue sections indicate the difference between the LRS2 model.*

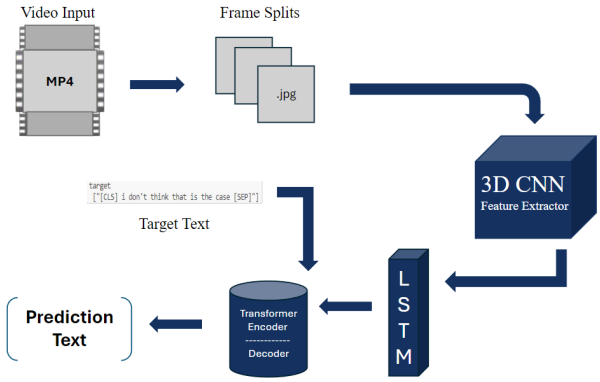

Figure 2. LRS2 model architecture.

5. Experiments

5.1. Speaking Model

Given the difficulties with word prediction, the team developed a model to predict if the person in the video frame was speaking with the idea that if the model can detect if they are speaking, it serves as a good backbone for predicting words. This model was also used as a heuristic to determine what changes to the CNN backbone could improve general performance of the other models. In order to reduce memory problems, the videos were truncated to 5 seconds. The videos are 25 frames per second, therefore each sample contained 125 frames. The model objective is to classify each frame correctly as speaking or not speaking.

The train data was composed of 77% of the frames where the person was speaking and 23% where they were not. A weighted cross entropy loss function was used to predict if they were speaking or not. Due to the data skew, the team weighted the cross entropy to match the distribution of the data 77/23. AdamW was used as the optimizer with a learning rate (lr) of 5e-5, weight decay of 1%, beta1 of 0.99 and beta2 of 0.95 with a batch size of 48. Originally, a batch size of 16 and 32 were used for the first few experiments, but the model exhibited better performance with a batch size of 48. This unfortunately was the biggest batch size that our hardware allowed. A larger batch size might be ideal. The 5e-5 lr was determined the best, after performing a lr sensitivity study for convergence within one epoch on a small dataset. A cosine scheduler was used with 5% of the steps of the first epoch used as warm-up steps. The train data was split with a fixed seed into a 90/10 split for validation. This model is able to obtain an 80% accuracy with the validation split containing 4,602 video samples. The model performed better than predicting only one case (speaking / not speaking) or random.

This model was then evaluated against the pretrain data. The pretrain data was sampled with a fixed seed and 10% of the data was used as the test set. This data contained 9,632 samples and 1,204,000 words. It obtained a 77.63% accuracy. From this test split, 75% of the data were frames where the person was speaking and 25% where they were not.

5.1.1 Error Analysis

The top 10 videos with the lowest accuracy's were studied to get an understanding of what the limitations of the model are. Most of the videos where the model performed poorly, only 16% - 27.2% of the frames were classified correctly, contained a person that had asymmetric lips or they were tilting their head while talking. Including -15 to 15 degree random rotation transformations could help the model be able to overcome the need for the subject to have symmetric lips and be more robust when the subject is talking while tilting their head.

5.1.2 Removing Layers of the Trained Model

In order to get insights into what matters most in the model, three of the 3D CNN layers were removed. The model was not retrained after removing the layers as the initial motivation was to see the performance decrement by removing intermediate 3d convolution layers. The video resolution remained the same at 160x160. Experiments were conducted in down-sampling the last layer with 3d average pooling and 3d max pooling in order to make the dimensions match.

These experiments were able to provide qualitative information about the importance between the temporal and video dimensions. The base model present in Table 1 is the base performance, 77.64% without removing any layers. After removing three layers, various pooling strategies were used. The base comparison for these pool layers would be pooling all dimensions equally. By average pooling equally, the model performance was 51.58% a 26.05 point decrement. The initial hypothesis was that the temporal dimension would be more important than resolution, but in reality the video resolution is the most important. Maintaining the most information possible for the resolution and reducing

Model	Pooling	Final Dimension	Accuracy
Base	None	[512, 125, 3, 3]	77.63%
All dimensions Avg Pool	AvgPool3d(2)	[512, 45, 5, 5]	51.58%
Temporal fixed	AvgPool3d((1,2,2))	[512, 125, 3, 3]	48.09%
Resolution Fixed	AvgPool3d((2,1,1))	[512, 45, 5, 5]	57.90%
All dimensions Max Pool	MaxPool3d(2)	[512, 45, 5, 5]	47.94%
Resolution Fixed Max Pool	AvgPool3d((2,1,1))	[512, 45, 5, 5]	57.90%

Table 1. Speak Model Removed Layers Study

the temporal dimension by 64% resulted in an accuracy of 57.90% only a 19.73 point decrement, while maintaining the temporal dimension as high as possible and matching the resolution of the base model with all the layer, results in the worse performance with an accuracy of 48.09% or a 29.54 point decrement. Table 1 shows there was no major difference between using max and average pooling. Note that if the final dimensions didn't match, the closest highest dimension for the variable being explored was used and the others were truncated or padded.

5.2. Speaking Words Model

The team then utilized the pretrained 3d convolution layers of the speak model to be able to predict words. Two linear layers were added on top and a Gelu activation layer in between the linear layers. The team explored both freezing the base speak model and training only the linear layers and training all the layers at once. Training all the layers at once resulted in faster convergence.

5.3. LRW Model

We first try to overfit the model with 102 samples from the training. The model is able to get loss = 0.0092 (see Figure 3) and accuracy = 92%.

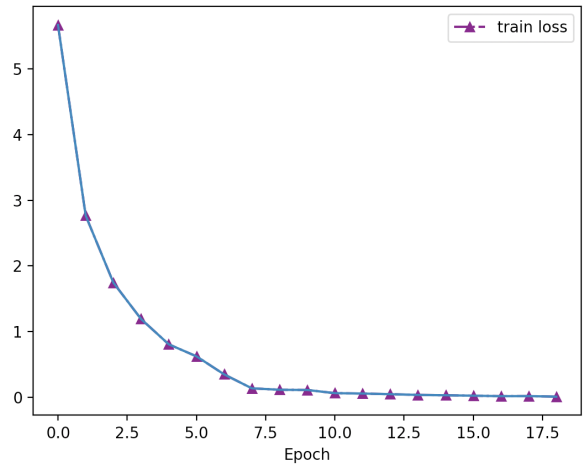

Figure 3. LRW model overfitting with small data size.

We trained the LRW model end-to-end for 20 epochs with batch size 16 (we also tried batch size 32, but it resulted in out of memory errors). We use the AdamW optimizer [15] with initial learning rate of 3e-4, and cosine scheduler[14].

5.4. LRS2 Model

5.4.1 Training and Optimization

To optimize the training process, we used the Adam optimizer due to its adaptive learning rate capabilities, which helps in converging faster and more effectively than traditional stochastic gradient descent [13]. The model was trained using cross-entropy loss as the loss function, which is particularly effective for classification tasks with multiple classes, such as our case where each word in the output sequence can be considered a separate class [23]. This setup ensures that the model's predictions are tightly aligned with the actual sequence of spoken words, ultimately improving the accuracy of the lip-read text.

During training, we encountered issues with exploding gradients. To mitigate this, we implemented gradient clipping, limiting the gradients to a range, ultimately preventing the gradients from growing too large. Due to memory constraints we also employed loss accumulation, which involve accumulating the gradients over mini-batches and updating the model weights less frequently. This has the downside that when performing backpropagation, additional compute and memory are required, but it is a net positive. We were having an issue where the computation graph kept growing leading to out-of-memory errors, which was solved by deleting the accumulated loss when it was no longer needed. Simply replacing it with a primitive, i.e., 0, was not enough. Additionally, to enhance the adaptability of the learning process, we incorporated a learning rate scheduler.

$$\text{Accuracy} = \frac{\text{Total Correct words}}{\text{Total Words on the Evaluation Dataset}} \quad (3)$$

All models and results contained herein utilize accuracy as the main metric. The metric is used on a per-frame basis, where on each frame we determine if the predicted word is correct, We defined our accuracy as shown in Equation 3. We do not take credit for words that are off by one character.

6. Results

6.1. Speaking Words Model

Figure 4 shows the model loss stagnating even with the cosine scheduler after 11 epochs, purple line, at a lr of 5e-5. The model was stopped at epoch 11 since the final iteration of the cosine scheduler lr was close to zero and was subsequently restarted with a reduced max lr of 1e-5 and 6 more epochs were conducted. Figure 5 shows performance improved slowly but again stagnated with a final validation accuracy of about 1% for sentence prediction. The total training time for this model was around 30 hours on an A100 GPU. Special care was taken to keep everything in the 40gb HBM memory so that the model did not suffer from the Ram to HBM I/O compute performance decrements.

Figure 4. Speak Word Model Trained for 17 Epochs. Note: The lower loss line is for the higher lr stagnated model. The upper loss line is interrupted due to tensorboard logging issues, but it is the same training run with an adjusted lr after epoch 11.

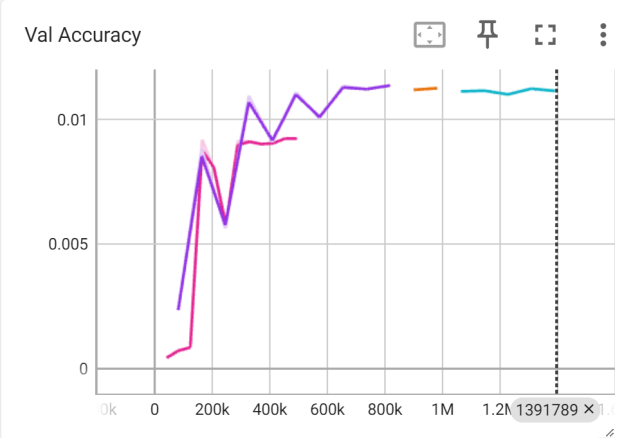

Figure 5. Validation Accuracy Plot of the Speak Model for the Aforementioned Runs.

Figure 4 also illustrates a run with a higher lr of 5e-4, shown as pink, where the model learned quickly but also stagnated earlier on. This model also obtained lower accuracy after being trained for a longer amount of time as shown in Figure 5. The smaller lr 5e-5 model, purple line, continued to learn, but also stagnated. It was later restarted with a lower lr of 1e-5. When using a small dataset of 10 videos the model was able to overfit and predict all the words correctly. Due to the compressed timeline, the team did not perform an error analysis on the model as the accuracy was low and instead allocated additional resources to the LRW model.

6.2. LRW Model

Table 2 shows the result of using Connectionist Temporal Classification (CTC) loss [9] or Cross-Entropy loss.

By comparing CTC and Cross-Entropy, we observe

Method	Accuracy
LRW + CTC	50.04%
LRW + Cross-Entropy	52.19%

Table 2. Comparison of performance based on word accuracy.

Cross-Entropy yields improved performance. Since CTC performs on a character-level, we think the model needs to train longer to achieve better performance (e.g. LipNet takes 5 - 7 days).

The team explored mismatched word pair errors, for example, THERE and THEIR, POWER and POWERS, WANTED and WANTS. Most of the word pairs are phonetically and ”visemically” similar to each other. The team then investigated the corresponding video clips and found that facial movements or gestures can interfere with the clarity of the lip movements and could be the cause of these incorrect predictions.

6.3. LRS2 Model

Our model shows signs of underperformance despite successfully converging during training (see Figure 6). Initially, it produces reasonable predictions, but after a few iterations, it tends to predict only the SEP token. After removing special tokens, the model shifted to repetitively predicting a neutral token such as ”the,” indicating it was still unable to generalize effectively. This is further illustrated in Figure 7 and Figure 8, where the frequency of the SEP token and ”the” is prominently high. Figure 7 shows the top 20 words, including special tokens, with SEP and ”the” being among the most frequent, suggesting the model’s bias towards these tokens. Figure 8 extends this analysis to the top 100 words, revealing a similar pattern of over-reliance on these common tokens. Notably, these issues were observed during training, and the model was never validated. This further suggests that the model has not seen enough varied training examples to generalize well.

We hypothesize that the model might be getting stuck in a local minima, causing it to settle on repetitive token outputs such as SEP or a common word like ”the.” This indicates a lack of generalization, finding the most common word to be the most important. Despite experimenting with several adjustments, including reducing the learning rate, using different schedulers and regularization techniques, experimenting with loss functions, and trying various optimizers, the issue persisted.

It is important to note that the dataset had a pretrain section, which may indicate that the model has not seen enough varied training examples to generalize well. That paired with the complexity of a task like lip-reading, may be the main reason our model in underperforming. Increasing the amount of training data or extending the training period could provide the model with more exposure to different examples, potentially improving its performance.

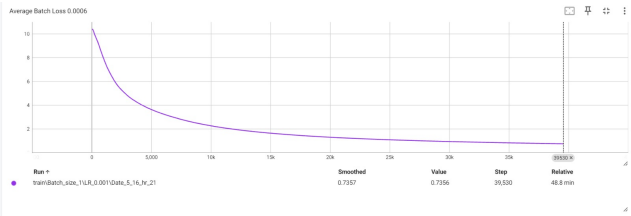

Figure 6. LRS2 model converging during training.

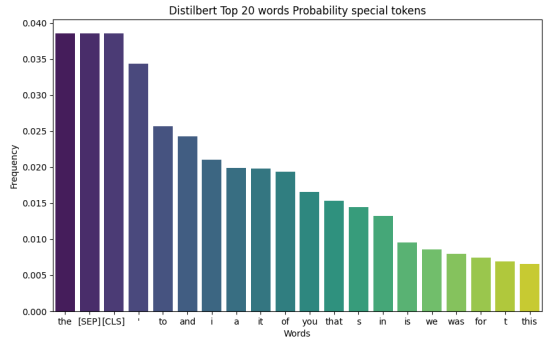

Figure 7. Probability of the Top 20 Words Including Special Tokens.

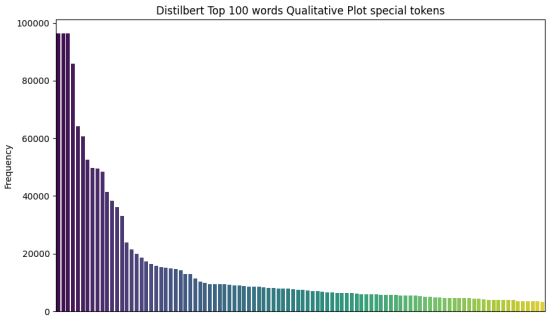

Figure 8. Qualitative Plot of the Top 100 Words Including Special Tokens. Note: X label was omitted as it is meant to be qualitative and show the drastic difference of frequency even on the top 100 words.

7. Conclusion

In summary, we developed two commendable models and one model that has significant room for improvement. Our model that predicts whether a person is speaking or not achieved an accuracy of 77.63%, successfully kickstarting our project. The LRW model also yielded promising results with an accuracy of 52.19%, demonstrating the feasibility of predicting spoken words without any audio input. However, our more challenging model, which attempts to predict spoken sentences, still requires substantial improvement as it appears to be getting stuck in a local minima.

While we are proud of our initial two models, we remain determined to enhance the performance of the sentence prediction model. Given the limited time allocated for this project, we were unable to fully resolve its issues within the current timeframe. We plan to continue experimenting and refining our approach in future work, incorporating the insights and enhancements we have identified.

8. Future Work

After reviewing some of the benchmark models on the LRS2 dataset, we identified several enhancements for future implementation.

First, regarding preprocessing, we observed that top-performing models often crop and scale input images to focus more on the mouth region of the person in the video and apply various transformations to the images. We intend to adopt these preprocessing techniques to improve our model's focus and accuracy. Some other data augmentation techniques also prove to be effective, including mixup (linearly interpolating between pairs of samples and their corresponding labels [16]), time masking (mask N consecutive frames with the mean frame of the sequence). [17] conclude Time Masking is the most effective augmentation method followed by mixup and Densely-Connected Temporal Convolutional Networks (DC-TCN) are the best temporal model for lip-reading of isolated words.

Secondly, for training, we plan to utilize both the train and pretrain data splits for training. The pretrain data split contains over 2 million word instances compared to the 300,000 in the train split and more than twice the vocabulary size. This provides a much richer dataset, allowing the model to be exposed to a greater variety of words and contexts, which is crucial for improving its generalization capabilities. Word boundary indicators are also useful in training. [26] show that concatenating word boundaries with the extracted features yields substantial improvement. Given our error analysis of the speak model, including a -15 to 15 degree rotation might help create a more robust model where it is not affected by the speaker tilting their head. We would limit the rotation to this bound since beyond that would be an unnatural pose for a speaker.

We also plan to extend the training duration. Many top-performing models are trained for 1 - 3 weeks, which allows the model to learn the necessary information to handle the complexity of the lip reading task. Using pretrained model or pretraining on different objective such as self-supervised learning could also help with performance. [18] pretrain on a subset of the 10% hardest words, which are 50 classes for LRW2 and prove such initialization allows for faster training, and performance improvement.

By incorporating these enhancements, we aim to greatly improve our model's performance and accuracy in predicting spoken text from lip movements.

9. Contributions & Acknowledgements

Akayla Hackson

- Preprocessing of the Data
- Architecture of LRSW2
- Dataloader

Miguel Gerena

- Architecture of Speak
- Helped debug LRSW2 and resolve memory issues
- Dataloader
- Word Distribution Analysis

Linyin Lyu

- Architecture of LRW
- Optimization of Models
- CTC loss and beam decode

References

[1] T. Afouras, J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman. Deep audio-visual speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):8717–8727, Dec. 2022.

[2] Y. M. Assael, B. Shillingford, S. Whiteson, and N. de Freitas. Lipnet: End-to-end sentence-level lipreading, 2016.

[3] J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman. Lip reading sentences in the wild. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, July 2017.

[4] J. S. Chung and A. Zisserman. Lip reading in the wild. In *Asian Conference on Computer Vision*, 2016.

[5] J. S. Chung and A. Zisserman. Learning to lip read words by watching videos. *Computer Vision and Image Understanding*, 173, 02 2018.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[8] D. Feng, S. Yang, S. Shan, and X. Chen. Learn an effective lip reading model without pains, 2020.

[9] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks. volume 2006, pages 369–376, 01 2006.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[12] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017.

[15] I. Loshchilov and F. Hutter. Decoupled weight decay regularization, 2019.

[16] P. Ma, B. Martinez, S. Petridis, and M. Pantic. Towards practical lipreading with distilled and efficient models, 2021.

[17] P. Ma, Y. Wang, S. Petridis, J. Shen, and M. Pantic. Training strategies for improved lip-reading. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, May 2022.

[18] B. Martinez, P. Ma, S. Petridis, and M. Pantic. Lipreading using temporal convolutional networks, 2020.

[19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[20] E. Petajan. Automatic lipreading to enhance speech recognition (speech reading). 1984.

[21] K. R. Prajwal, T. Afouras, and A. Zisserman. Sub-word level lip reading with visual attention. 2021.

[22] K. R. Prajwal, R. Mukhopadhyay, V. Namboodiri, and C. V. Jawahar. Learning individual speaking styles for accurate lip to speech synthesis, 2020.

[23] C. E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[24] B. Shi, W.-N. Hsu, K. Lakhotia, and A. Mohamed. Learning audio-visual speech representation by masked multimodal cluster prediction. *arXiv preprint arXiv:2201.02184*.

[25] B. Shi, W.-N. Hsu, and A. Mohamed. Robust self-supervised audio-visual speech recognition. *arXiv preprint arXiv:2201.01763*.

[26] T. Stafylakis, M. H. Khan, and G. Tzimiropoulos. Pushing the boundaries of audiovisual word recognition using residual networks and lstms, 2018.

[27] T. Stafylakis and G. Tzimiropoulos. Combining residual networks with lstms for lipreading, 2017.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.