Henry Hirsch
Deep Learning
Group 4

Individual Final Report

Introduction

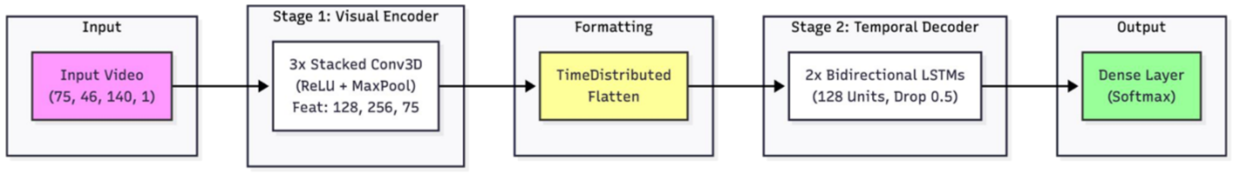
Our project was inspired by the 2016 paper *LipNet: End-to-End Sentence-level Lipreading* by Yannis M. Assael, Brendan Shillingford, Shimon Whiteson, Nando de Freitas. This groundbreaking paper describes the first time that convolutional neural networks (CNNS) and recurrent neural networks (RNNs) were jointly applied to the task of lipreading. The development of this architecture was a significant milestone in automated lipreading due to the fact that the CNN layers are able to learn local spatiotemporal patterns while the RNN layers are able to learn global temporal patterns. When combined, these layers are able to learn the appearance and sequence of elemental lip movements at the local (syllable) level and understand their place in the global (word or sentence-level) sequence. This approach yields unprecedented character-level prediction accuracy.

The original LipNet paper uses data from the GRID audiovisual sentence corpus, a large multi-speaker audiovisual sentence corpus intended to support computational-behavioral studies in speech perception. The corpus consists of high-quality audio and video (facial) recordings of 1000 sentences spoken by each of 34 talkers (18 male, 16 female). 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 six categories. The categories consist of, respectively, {bin, lay, place, set}, {blue, green, red, white}, {at, by, in, with}, {A – Z}, {zero – nine}, and {again, now, please, soon}, yielding 64,000 possible sentences. For example, two sentences in the data are “set blue by A four please” and “place red at C zero again”.

The corpus also provides the corresponding alignment files for each set of videos. These alignment files contain three columns: starting frame, ending frame, and word. The purpose of these files is to map constituent sequences of frames, from each of the 75-frame videos, onto textual output (the word being spoken between the starting and ending frames of the sequence). “Sil” is used to denote silence at the start and end of each video. These alignment files essentially enable researchers to set a target (the text of the word being spoken) and establish features for input into a model (the frames from the corresponding sequence).

Our group attempted to create an architecture similar to LipNet and train a model capable of taking videos of speakers as input and outputting predicted text of what the speaker said. Each group member attempted to develop their own model with the intent of using the best one for our project. I made two attempts, the first on a single speaker (s1) and the second on three speakers (s2, s3, and s4). My second model is the one that our group ended up using. I also contributed substantially to the coding of the app for our demo.

Individual Work
As previously stated, I developed two models. This first was trained on one speaker (s1). We found that this model didn't generalize well, partially due to the single speaker (lack of diverse speakers led to overfitting) and partially due to the fact that s1 was off-center in the videos (in contrast to the videos of the other speakers). Therefore, I developed my second model which was trained on s2, s3, and s4. To reduce the training time on this larger model, I increased the batch size (from two to eight), reduced the number of epochs (from 150 to 50), and incorporated mixed precision optimization (model uses float16 whenever possible during training and only uses float32 for critical values and calculations). I also added exponential learning rate decay after the 19th epoch to help the model escape any local minima that it might be stuck at. Both models used this architecture:



The model combines three 3D CNNs and two Bidirectional Long Short-Term Memory (LSTM) layers. The architecture consists of 8,471,924 trainable parameters and processes 75-frame video sequences of dimensions 46×140 pixels (cropped to the lip region using dimensions provided in the GRID corpus).

First, the three 3D CNNs capture both spatial features (lip shapes) and temporal features (lip movements over time). Unlike 2D CNNs which process each frame independently, 3D CNNs process multiple frames simultaneously, learning motion patterns crucial for lip reading. The subsequent ReLU activation functions then pass on any potential features (anything with a value > 0). Next, the pooling operations progressively reduce spatial dimensions (height and width) while maintaining the temporal sequence length (time). These layers identify features and learn local spatiotemporal patterns.

The time-distributed flatten then flattens the 3D tensors (height * width * time) fed into it into 1D vectors (time) with numeric values representing the flattened values for the height and width dimensions.

These 1D vectors can then be fed into the two bidirectional LSTM layers. These layers process the sequence in both forward and backward directions, allowing the model to use both past and future context when predicting each character. This is crucial because frames don't have much meaning in isolation. For example, imagine a frame of partially open lips. Are they opening or closing? It is impossible to know without the context provided by the adjacent frames in the sequence. Furthermore, the pronunciation of certain letters in words depends on the letters adjacent to them (e.g. the “c” in “call” vs. the “c” in “champion”). These LSTM layers are followed by 0.5 dropout to prevent overfitting to the training data (500 sampled videos per speaker is a lot of data so fine to have high dropout). These bidirectional LSTM layers learn global temporal patterns.

Finally, the dense output layer with the SoftMax activation function converts raw numeric values to a distribution of probabilities for every character in our vocabulary (A-Z + 0-9) so that we can get the most probable character output for every frame in our video. It uses a Connectionist Temporal Classification (CTC) loss function to bridge the divide between the 75 frames in each video and the <75 characters in the words that were spoken (no 1:1 relationship). It allows the model to learn which frames correspond to which characters without explicit frame-by-frame labels.

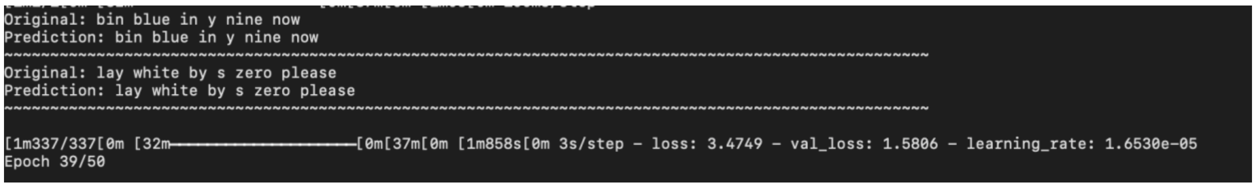
Results

We had trouble getting our models to train (corrupted videos, multiple crashes after 14+ hours of training, etc.) so we were short on time and didn’t do a great job of getting results/metrics. The LipNet evaluation uses two metrics to determine how accurate their predicted text outputs are: Character Error Rate (CER), the fraction of incorrectly predicted characters, and the Word Error Rate (WER), the fraction of incorrectly predicted words.

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

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

My model crashed during training and had to be restarted multiple times. Consequently, I had to cut its training short (39th epoch out of 50) due to a lack of time. However, I did get loss down to 3.4749 and validation loss down to 1.5806. So, the model was learning, it just hadn’t finished. As you can see below, it was starting to be able to predict some words in their entirety. However, it didn’t generalize well.



```
Original: bin blue in y nine now
Prediction: bin blue in y nine now
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Original: lay white by s zero please
Prediction: lay white by s zero please
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[1m337/337[0m [32m━━━━━━━━━━━━━━━━━━━━[0m[37m[0m [1m858s[0m 3s/step - loss: 3.4749 - val_loss: 1.5806 - learning_rate: 1.6530e-05
Epoch 39/50
```

I was getting a CER around 0.50 and a WER around 0.90. So, the model had made some progress on predicting characters but still struggled with complete words.

Summary and Conclusion

In short, my second (multi-speaker) model seemed to be learning despite its training being cut short. Ideally, all 34 speakers would be incorporated with additional data augmentation. This should improve generalization. It would also probably be beneficially to incorporate some advanced lip-tracking packages during the data preprocessing rather than crude frame cropping. Doing so would produce more tailored higher quality input images for the model, and consequently better training.

Code

Around 10% of the code was written by me. The other 90% was taken from other lipreading projects or AI-generated via my prompts.

References

LipNet: End-to-End Sentence-level Lipreading
GRID audiovisual sentence corpus