ReadME

Greg Bodik ‘23, Kaitlyn Chen ‘23, and Kevin Zhou ‘23
Cornell Data Science Insights: Fall 2020
Github: https://github.com/CornellDataScience/Insights-Lip-Reading

1. Overview

The current project, entitled Insights-ReadME was designed with the intention of producing a real-time end to end lip reading system, one that relies solely on visual information to produce a textual output representative of speech utterances. Although proving too large in scope to be implemented in a single semester, this first phase of the project sought to generate useful features from video data to enrich future lip reading models, while also setting the framework for a lip-reading architecture, and providing the visual interface that would be needed to use it. To accomplish the first part of this goal, we trained convolutional neural networks (CNNs) on datasets of faces with ground truth emotion labels to generate a model that upon video input would approximate a subject’s emotion and age. In the context of our eventual lip reading, this allows several benefits. Namely, in contrast to traditional systems that depend on continuous streams of data from which to draw text, an alternative could employ NLP (withemotion characteristics behind utterances as features) to not only output current words being said, but also predict ones that are going to be uttered. In addition to the generation of this model, the given project found and attained the usage rights to a dataset developed specifically for the purposes of lip-reading and developed most of the tools necessary to pre-process that data. Those tools as well as the model output are displayed in the final program. Consequently, this leaves Insights-ReadME poised to implement the lip-reading model itself in future semesters.

2. Background

Conventional human experience readily conveys that hearing is the dominant sense involved in communication. However, although speech dominates communication channels, vision is a secondary, yet essential component of these interactions. During communication, humans rely on visual signals to affirm those that they hear, and humans perceive different syllables when fed conflicting auditory and visual information (Mcgurk and Macdonald). Any speech-to-text service on a

phone or computer serves as proof that conventional systems of audio speech recognition (ASR) can achieve very high accuracy in converting audio input to text, but are limited in their ability by the quality of the acoustic input signal they receive. Contrastingly, literature describes automated lip reading (ALR) systems as those that rely purely on visual data of speech to produce corresponding text, exemplified in (Zhou et al., Fernandez-Lopez and Sukno). Here again, human experience dictates that accurately recognizing a word solely from its associated lip movements is a far more challenging task, as it is easy to conflate many similar *looking* but differently *sounding* phonemes; the building blocks of words. Nevertheless, the pursuit of ALR systems remains useful due to their application in many instances when ASR is not an option. The most obvious example is in the case of those who are hearing impaired, and are forced to rely solely on visual cues in many social contexts. In a broader sense, ALR applies to many situations in which audio signals are noisy or unreliable, and thus ASR systems rendered useless, for example, a noisy coffee shop. Consequently, our project aimed to investigate the ALR avenue of speech analysis.

Sentiment analysis has also become an increasingly essential field in our data-driven world (Jayalekshmi and Mathew). In order to personalize web resources, automated sentiment analysis has been crucial in analyzing a user's opinions about certain products and policies simply through the things that they type and click. Consequently, a subset of sentiment analysis—facial sentiment analysis—has become an extremely popular research field in computer vision as noted by Torres et al. Generally, facial sentiment analysis is conducted by capturing an individual's facial expression in real-time through a video stream, classifying it through a machine learning-based image classifier, and determining which emotional category of angry, surprised, neutral, happy, sad, or fearful the facial expression most resembles.

3. Methods and Results

Crucial to the eventual creation of a lip reading is the data acquired to train it. Lopez and Sukno describe the databases available in terms of the task performed by their participants, or in other words, the type of speech being uttered. Mainly, tasks fall into categories based on the complexity and duration of the associated

phonemes. In order of increasing complexity, these tasks include: reading single alphabet characters, reading single digits (the ALR equivalent of MNIST), reading single words, and reading full sentences. In the current project, the LRS3 dataset (Afouras et al. 3) was selected due to its standing as one of the largest compiled datasets in the ALR field, and the complexity of the task performed within (reading full sentences). The dataset contains thousands of spoken sentences from TED and TEDx videos with no overlap between the pre-training, training, and testing sets. The pre-training set consists of 5,090 videos, 118,516 utterances, and 3.9 million word instances. The training set consists of 4,004 videos, 31,982 utterances, and 358,000 word instances. The testing set consists of 412 videos, 1,321 utterances, and 10,000 word instances. Accompanying each pre-training video file is a text file containing the face detection bounding box for each frame, the ASD score for each word, and the word boundary timestamps. In total, the dataset contains over 400 hours of video, cropped to a resolution of 224 x 224 pixels for face frames, and at 25 frames per second. In comparison with the group's first dataset compiled for such purposes, which used data from the British Broadcast Corporation (BBC), we felt the LRS3, which instead uses data from TED talks featuring a variety of speakers and English dialects would produce a more robust model that was less susceptible to bias toward a particular accent or speech pattern. Notably, the Oxford Research Group who compiled the data achieved an accuracy score of close to 60% for sentence decoding on the smaller LRS dataset (Chung and Zisserman).

Concerning the videos in the pre-training set, we initially planned to select several random frames of each word uttered, train the model using those still frames and the phonemes from the text file that the speaker was uttering at that timestamp, and use natural language processing to predict or complete the rest of the phrase. As a result, we had to preprocess our videos and convert them into still images centered on the speaker's mouth. To do so, we wrote a script that converted each video into a directory of its frames. Then, we utilized the `face_recognition` Python library—which utilizes a convolutional neural network to identify distinct facial features—to crop each frame to an image containing only the speaker's lips and a 10 pixel border. Given the volume of the dataset, we discarded any video that could not have the speaker's mouth "extracted" at every frame. To demonstrate the success of our preprocessing scripts, we created the following powerpoint.

To train our facial sentiment analysis model, we used the Kaggle dataset, fer2013, which contains a training set and a testing set that have each been further categorized into seven emotions: angry, fearful, surprised, disgusted, happy, neutral, and sad. Each emotion contained a folder of stock images of a diverse range of individuals (in terms of age, gender, and ethnicity) expressing that emotion. We noted that some emotions were underrepresented. For example, there were only 437 pictures of "disgusted" in the training dataset compared to 7,216 pictures of "happy." We took that into account when assigning weights to our model.

3.1 Lip Extraction

Lopez and Sukno's paper, "Survey on automatic lip-reading in the era of deep learning," compares the prevalence of existing models with DNN and CNN architectures. They highlight that the recent shift from traditional technological architectures "typically consisting of image features in combination with HMMs, toward end-to-end DNN architectures, currently dominated by CNN features in combination with LSTMs." This progression has allowed for more diverse input types to the model, such as RGB images, gray-scale images, 3D and 2D structures as well as network specifications such as the number of convolutional and fully connected layers. LSTM networks also differ from traditional automatic lip reading systems in how their outputs are decoded, the network's direction, and the number of layers. As a result, there has been a significant improvement in performance over the last few years. Consequently, we decided to build our lip reading model using a convolutional neural network.

We decided that the input to the convolutional neural network would be an RGB image, specifically a frame derived from a video of the LRS3 dataset. After saving each set of videos into different folders, we passed the directory of each video of the pre-training and training sets into a script that read in the frame at the current index and saved it as a separate image with the video name as well as the frame number in a separate folder. This directory of frames would then be passed into our "extract lips" script. For

Figure 1

each individual image, we used the Python library, `face_recognition`, to first save the location of each major "landmark" facial feature in an array. For example, the location and outline of the speaker's bottom lip would be saved as `face_landmarks_list[0]['bottom_lip']`. We compiled the (x, y) coordinates of the top and bottom lip, added it to a margin of 5% of the video height and width respectively (to account for variation), and saved those as the new borders for the cropped image of the speaker's mouth. The frame was then cropped according to those new boundaries and saved in a separate directory. An example of this is shown in **Figure 1**.

After attempting to crop each frame to just the mouth, we checked whether the number of cropped mouth pictures was equal to the number of frames of the video that it was taken from. If they were not equal, we were able to dispose of that video's cropped images because we had enough remaining data in the pre-training and training sets.

Once we compiled the mouths of each frame of each pre-training and training video, we encountered some challenges. In the case of our preprocessing, we did not take into account which phoneme each frame corresponded to, nor what to do with duplicate frames that corresponded to the same phoneme. This would require us to develop an algorithm that would determine the timestamp boundaries of each phoneme rather than each word, choose one frame within those boundaries that best visualized the shape of the speaker's mouth, and save that frame in a directory associated with the phoneme.

Another complication that would arise when passing the testing dataset through our model would be the lack of word boundary timestamps. Since the TED and TEDx speakers tended to speak quickly and their thoughts flowed together, it would be difficult to differentiate between words, let alone the individual phonemes. As a result, we would struggle with selecting a frame that would clearly express the distinct phoneme the speaker was uttering to a high level of accuracy. We realized that we needed to build additional infrastructure to support our model. In fact, we decided that NLP could be extremely useful in predicting which phoneme or word

the speaker was uttering given other environmental factors such as the speaker's age or emotions. Thus, we redefined the scope of our project.

3.2 Modeling: Facial Sentiment Analysis

To further our goal of being able to convert a stream of lip frames into text as well as predicting future words in the utterance, we wanted to consider additional factors that could help decipher the speaker's intentions. One factor that came to mind was the speaker's emotions, as knowing how someone is feeling could help in guiding the models. For example, we wanted to see if certain words in an eventual lip-reading model would correspond to their speakers' emotions during utterance, and thus help to generate more accurate predictions.

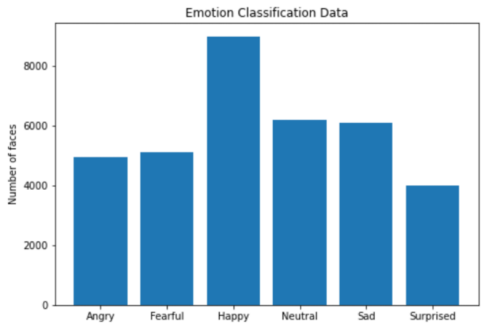

Figure 2

Images from the fer2013 dataset originally spanned seven emotions: angry, disgusted, fearful, happy, neutral, sad, and surprised. Images from the disgusted class were removed because they spanned only 1.5% of the total and would consequently be neglected by most training algorithms. After data was randomly split into training and testing sets (80% and 20% respectively), we had 28,273 unique faces in the training set, and 7067 in the test set. The distribution of the data across the emotion categories is shown in **Figure 2**.

Each image was converted to grayscale and a standard resolution of 48 x 48 pixels, for an input size of (48 x 48 x 1). A convolutional neural network, built sequentially using the keras API for tensorflow, was used to detect emotions from these faces. The model included 5 convolutional layers, with max pooling performed after convolution to both reduce the computational complexity needed to train the model and to abstract the results of convolution in an effort to mitigate overfitting. The full architecture is shown in figure **Figure 3**. In order to avoid overfitting, and duly to augment the size of the data, images in the training dataset were randomly rotated by up to 20 degrees, zoomed by up to 25%, or flipped horizontally during processing. The final layer of the network was a softmax layer

that assigned each image a label from one of the six possible output classes of emotion. Overall accuracy of the model reached 60% after 40 epochs, beating the baseline accuracy for a 6-category classification of 16%, and other efforts on the same data that reached ~55% accuracy. These results, as well as a confusion matrix for predictions on the test set are shown in **Figure 4**. The accuracy of this model was greatly limited by the relatively low resolution of the input images, meaning that most deep CNN architectures (such as ResNet50 or VGG19) that are often used in image recognition, overfit tremendously on the given data, or are inapplicable due to negative dimensionality. These trends are seen in the interactive D3 accuracy visualization made for the purposes of this project. Although higher resolution image data does exist for the given task, due to our ultimate goal to use emotions as prediction insights during lip reading, we felt it was important for any model component to robustly work on the kind of low-resolution data that results from face cropping during continuous video as is the case for the intended LRS3 dataset.

Layer (type)	Output Shape	Param #
conv2d_10 (Conv2D)	(None, 46, 46, 32)	320
conv2d_11 (Conv2D)	(None, 44, 44, 64)	18496
max_pooling2d_8 (MaxPooling2	(None, 22, 22, 64)	0
conv2d_12 (Conv2D)	(None, 20, 20, 128)	73856
max_pooling2d_9 (MaxPooling2	(None, 10, 10, 128)	0
conv2d_13 (Conv2D)	(None, 8, 8, 128)	147584
max_pooling2d_10 (MaxPooling	(None, 4, 4, 128)	0
conv2d_14 (Conv2D)	(None, 2, 2, 256)	295168
max_pooling2d_11 (MaxPooling	(None, 1, 1, 256)	0
flatten_2 (Flatten)	(None, 256)	0
dense_4 (Dense)	(None, 1024)	263168
dense_5 (Dense)	(None, 6)	6150

Total params: 804,742
Trainable params: 804,742
Non-trainable params: 0

Figure 3

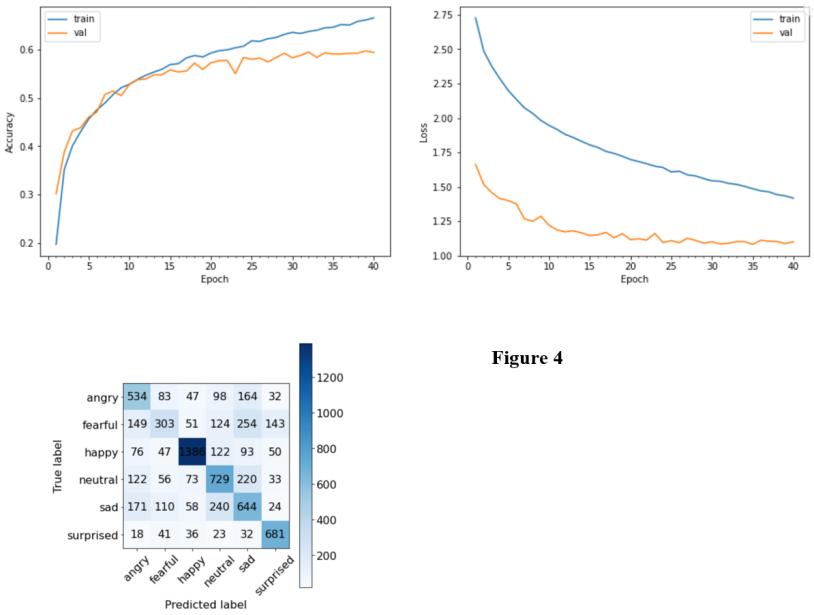

Figure 4

4. Design Decisions

In an effort to have an interactive way for the user to be able to access and view the effects of our models, we created a graphical user interface (GUI) using the tkinter package in Python. We originally used Java's Swing toolkit to build the GUI, but ended up switching to Python's tkinter to make the integration with the modeling scripts easier, as those were written in Python.

The functions of the GUI are accessed through buttons on the main screen, with buttons for an introduction to the project, cropping lips, opening a live video stream that will read facial sentiment analysis in real time, and an explanation of next steps and future features. In order to be able to resize and place the buttons more effectively, we used pixelVirtual to place an invisible pixel in each button and used that to set configurations for the buttons. Each time a button is pressed, a corresponding function in Gui.py is called by setting the command option equal to that function.

Clicking the "Introduction" and "Future Features" buttons both open up a new text box that describes an overview of the project and future work, respectively. We thought that having these would be helpful in giving the user context for the function buttons and making it a more streamlined and interactive experience. Since there are different models and sections of the project, with the intent of eventually resulting in a synthesis into an ALR system, it's important that the user is provided with the project's goals. Along those lines, we used the Canvas widget to put pictures above each button that demonstrate the effects of each button.

Clicking the "Crop Lips" button uses tkinter's filedialog module to open up an input file request that allows the user to select a video from their computer. After selecting the file, this window closes and frames.py is then called to parse the video into still frames. Then, extract_lips.py is called, going through the folder that contains the still frames and creating a new folder with frames of just the cropped lips portion, as detailed in section 3.1.

Clicking the "Capture" button calls emotions_disp.py, which opens up the computer's camera, and if it detects a face, will create a box around it with a

detected emotion written above the box. This process happens in real time, so the written emotion, as well as the box, can change if the subject's facial expression changes or if the subject moves around.

5. Conclusions and Future Work

As stated above, the models we built this semester serve as infrastructure to better implement our Automatic Lip Reader. As such, the combination of these models and other natural language processing features would allow us to not only process individual, random frames taken of a video but also consider the linguistic, situational, and social environments the speaker is speaking in. The emotions an individual is feeling greatly influences the words they choose to say, and this information would consequently improve the accuracy of our Automatic Lip Reader in predicting the speaker's next words and phrases.

Furthermore, we were considering constructing other computer vision models as well, such as an age prediction model. While this feature is also rather subjective, dependent only on visual appearance, the general ability to judge one's age can provide valuable information on the vernacular they regularly use.

Works Cited

Afouras, Triantafyllos, et al. “LRS3-TED: A Large-Scale Dataset for Visual Speech Recognition.” *ArXiv:1809.00496 [Cs]*, Oct. 2018. *arXiv.org*, http://arxiv.org/abs/1809.00496.

Chung, Joon Son, and Andrew Zisserman. “Lip Reading in the Wild.” *Computer Vision – ACCV 2016*, edited by Shang-Hong Lai et al., vol. 10112, Springer International Publishing, 2017, pp. 87–103. *DOI.org (Crossref)*, doi:10.1007/978-3-319-54184-6_6.

Fernandez-Lopez, Adriana, and Federico M. Sukno. “Survey on Automatic Lip-Reading in the Era of Deep Learning.” *Image and Vision Computing*, vol. 78, Oct. 2018, pp. 53–72. *ScienceDirect*, doi:10.1016/j.imavis.2018.07.002.

Jayalekshmi, J., and T. Mathew. “Facial Expression Recognition and Emotion Classification System for Sentiment Analysis.” *2017 International Conference on Networks Advances in Computational Technologies (NetACT)*, 2017, pp. 1–8. *IEEE Xplore*, doi:10.1109/NETACT.2017.8076732.

Mcgurk, Harry, and John Macdonald. “Hearing Lips and Seeing Voices.” *Nature*, vol. 264, no. 5588, 5588, Nature Publishing Group, Dec. 1976, pp. 746–48. *www.nature.com*, doi:10.1038/264746a0.

Torres, Alex D., et al. “Chapter 3 - Patient Facial Emotion Recognition and Sentiment Analysis Using Secure Cloud With Hardware Acceleration.” *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications*, edited by Arun Kumar

Sangaiah et al., Academic Press, 2018, pp. 61–89. *ScienceDirect*, doi:10.1016/B978-0-12-813314-9.00003-7.

Zhou, Ziheng, et al. "A Review of Recent Advances in Visual Speech Decoding." *Image and Vision Computing*, vol. 32, no. 9, Sept. 2014, pp. 590–605. *ScienceDirect*, doi:10.1016/j.imavis.2014.06.004.