Lip-reading from images

Massimiliano Baldo
massimiliano.baldo@studenti.unipd.it

Cristiano Panighel
cristiano.panighel@studenti.unipd.it

Davide Bassan
davide.bassan.1@studenti.unipd.it
https://github.com/davidebassan/project-vcs

Abstract

Lip reading is the method of visualizing the lip movements of a speaker and identifying the spoken speech. Recent advancements in the field of computer science have helped automate this challenging task. The system consists of three main stages: face and lip detection, feature extraction, and text recognition. The accuracy of the system largely depends on the lip localization and the robustness of the extracted features. The system has various practical applications. In this project, we present various methods for predicting phrases and words from images without audio signals using the MIRACL-VC1 dataset. We employ an architecture that uses a CNN pre-trained model and then explore different ways of integrating it with a custom RNN. We also try to improve the results by keeping the same CNN as a basis and changing the initial RNN with LSTM and then with GRU. Experimental results show that these changes improve the accuracy of word recognition and various quantitative metrics.

1. Introduction

One of the most critical aspects of human life is communication. It can be done via speaking, writing and many other ways using different mediums. Lip-reading is an important aspect of human-computer interaction in noisy environments where there may be a lack of acoustic information or where audio speech recognition may be difficult. It can also be useful as a hearing aid for the hearing impaired. However, lip-reading systems face several challenges due to variations in input, such as facial features, skin colors, speaking speed and intensity. This leads to a naturally ambiguous result at a viseme level, where different characters can create the same lip sequence. The context information can alleviate such ambiguity to some extent, but many systems are still limited to a small number of phrases and speakers. To aid in lip-reading, more visual input data can be collected in addition to color image sequences, such as depth image sequences.

In this study, we used a dataset of image sequences that each show a person speaking a word or phrase. The goal was to classify these sequences, taking into account their varying lengths and number of features. To capture the temporal information, various methods were used, including locating the talking mouth region that comprises the motion associated with speech.

The first method used the VGGNet pre-trained on ImageNet to extract a set of features from each individual image, and then passed each sequence of features through a custom RNN with several recurrent layers to retrieve the classification label from the final output. The second and third methods used the same CNN as the backbone and changed only the RNN to perform the classification. For the second method, each sequence of features was passed through several LSTM layers, obtaining the classification label from the final output. For the third method, the LSTM layers were substituted with GRU layers and the classification was still retrieved from the final output. In addition, we also tried to change the VGGNet with other two CNN pre-trained again on ImageNet which are MobileNet and finally with ResNet. The latter turned out to be the best in our particular case and allowed us to obtain much better results.

In order to make the problem tractable, we formulate it as a classification problem of detecting what words or phrases are being spoken out of a fixed set of known words and phrases. Each method received a single image sequence as input, and produced a single word or phrase classification label as output. Code available at https://github.com/davidebassan/project-vcs.

2. Related works

There have been several publications and studies on the lip reading problem, with most of the work utilizing non-neural network approaches. These approaches extract various features from the images and then use machine learning

algorithms to classify what was spoken.

In the paper [3] Hidden Markov Models (HMM) were proposed as a way to perform lip reading using only image and depth information. The system consisted of two main parts: feature extraction and speech recognition. The first step estimated the speaker's face pose and then detected the mouth, followed by the use of feature descriptors to extract interesting data for the model, such as Histogram of Oriented Gradients (HOG). The second step segmented the speech video to look for frames corresponding to speech expressions. The features from these frames were then fed into the HMM for classification. The dataset is the same as what we use, i.e MIRACL-VC1 [6]. The final results for speaker-independent testing, where the data from one speaker was never used during training, showed an accuracy of 66.7% for phrases and 59.8% for words.

In another paper [5] a four-step method was proposed for attempting the task of lip-reading. The process is composed of face tracking, mouth region extraction, feature computation, and classification using Support Vector Machines (SVM). Different datasets were used, including MIRACL-VC1, with both 3D images and depth information. A unique aspect of their implementation was a speaker identification system, allowing the algorithm to learn different models based on different speakers. They achieved an accuracy of 79.2% for phrases and 63.1% for words on the MIRACL-VC1 dataset.

A different approach was studied in [2] where a non-deep learning method was used for lip reading. The authors used Random Forest Manifold Alignment for training and tested their model on various lip reading datasets, comparing their results to other approaches.

Differently from the previous approaches, in the papers [7], [4] it was tackled the issue of variance in sequence length by using Long Short-Term Memory (LSTM) layers on top of a neural network. Their results were similar to those of Convolutional Neural Networks (CNNs) and showed improvement over an SVM with HOG or Eigenlips features [1]. Although they used a relatively large dataset and all of their tests were speaker-dependent, with train and test data taken from the same speaker, this suggests that LSTMs could indeed show improvement over dedicated sequence classifiers.

3. Dataset

In our project, we have decided to use the MIRACL-VC1 dataset. The dataset was created from 15 people who spoke each of them ten words and ten phrases ten times leading to a total of 15 × 20 × 10 = 3000 instances. Each instance is a sequence of color and depth images of 640 × 480 pixels (Fig. 1). In our project, we have decided to use only the color image and discard the depth information to be consistent with the pre-trained CNN model. The words and phrases in the dataset are listed in Table 4. The length of sequences varies, with the minimum length being 4 images and the maximum being 22 images for words, and 6 to 27 images for phrases.

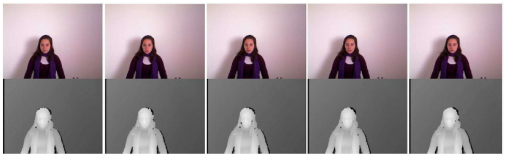

Figure 1. Instance of MIRACL-VC1.

Words	Phrases
Begin	Stop navigation
Choose	Excuse me
Connection	I am sorry
Navigation	Thank you
Next	Good bye
Previous	I love this game
Start	Nice to meet you
Stop	You are welcome
Hello	How are you?
Web	Have a good time

Table 1. Words and phrases in MIRACL-VC1 dataset.

The data was split into 9 people for training, 3 people for validation, and 3 people for testing, resulting in 1800 instances in the training set, 600 instances in the validation set, and 600 instances in the test set. Our objective is to achieve the highest possible classification accuracy in the test set, differentiating between words and phrases.

3.1. Data Pre-processing

Each figure in the dataset contains a lot of background information which is not relevant to the lip reading task. To address this, we tested multiple face detection modules, including those in OpenCV such as *haarcascade_frontalface_deafult* and *haarcascade_mcs_mouth*, as well as the face detector offered by dlib which is *shape_predictor_68_face_landmarks.dat*. Although the dlib face detector was slower than the other models, it was more accurate in identifying the lips, which is crucial for solving the lip reading problem. Given the small size of our dataset, it was important to avoid wasting computation on irrelevant parts of the image. After this step, the size of each image was reduced to 100 × 100 x 3. This size may be further reduced by cropping as required, depending on the method used for training. During this data pre-processing phase, we also tried to normalize the inputs, i.e. we tried to bring all the values present in the images within a fixed range so as to be able to compare them with greater precision.

3.2. Data Augmentation

Our dataset contains a total of 3000 instances, which means that our dataset is relatively small compared to other datasets used for deep learning tasks. To overcome this limitation, we employ data augmentation techniques to artificially increase the size of our dataset. These techniques include random rotations, zooming in and out, random shifting both horizontally and vertically from the original images. We also apply shear transformations, which involve fixing one axis and stretching the image at a certain angle, creating a kind of stretching of the image.

Additionally, we also flip some of the images horizontally. For the new points created by the transformations that have no value, we fill them with the nearest pixel value, which is determined by selecting the closest pixel value and repeating it for all empty values.

4. Method

We describe the various methods used to address the lip reading problem using Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). Our model primarily relies on using CNNs as they are effective in extracting features from images. However, one of the challenges faced is that the dataset contains multiple images for each data instance, which means that for each data pair (x, y), where y is the label to be predicted, x is not a single image but a sequence of images with variable length. To tackle this problem, one of our initial approaches was to use LSTM layers to process the data sequences without the need to create a single large image by concatenating the individual images, which is more complex and less intuitive. This approach was expected to perform better as the CNN layers would not have to infer the temporal information.

4.1. Transfer Learning

It is a way to speed up the learning process and improve accuracy in creating new models by leveraging existing models. In practice, there is often not much information to learn from the beginning of the Convolutional Network. Most problems can be solved using models that have already been trained, such as the number of layers, activation functions, hyperparameters, etc. Transfer learning reduces the amount of time and effort required to train a new model as the initial performance of the new model is higher than if it were trained from scratch.

There are several benefits of using transfer learning:

- Higher starting point: The initial performance of the new model is higher than it would be if it were trained from scratch.
- Faster learning rate: The rate of improvement during training is faster than if the model was trained from scratch.
- Better overall performance: The final performance of the trained model is better than it would be if it were trained from scratch.

In our particular problem, where we have not very much data, transfer learning can enable us to develop skillful models that we simply could not develop in the absence of transfer learning. In our case, where the dataset is small, transfer learning can help us develop models with high accuracy that would otherwise not be possible to achieve.

4.2. Convolutional Neural Network

Looking at the architectures used mostly in trying to solve lip reading problem, we noticed that the most common architecture used in solving this task mainly consists of a CNN followed by some RNN layers.

4.2.1 VGG16

Our first approach involved utilizing a pre-trained VGG16 deep convolutional neural network as a feature extractor on the MIRACL-VC1 dataset images. The VGG architecture consists of blocks, with each block being composed of 2D Convolution and Max Pooling layers. The weights in the neural network are updated using the backpropagation algorithm, which makes small changes to each weight in a way that the model's loss decreases. As the gradient keeps flowing backward to the initial layers, the gradient becomes smaller, causing the updates to the initial layers to be small and leading to an increase in training time.

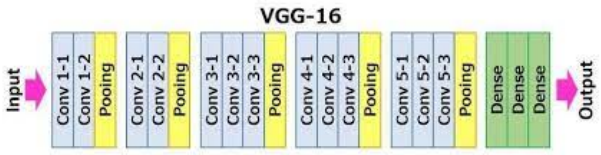

Figure 2. Representation of VGG-16.

For these reasons, our approach was to keep the ImageNet learned weights frozen and use them as a feature extractor for images. We also removed the layers related to the classification, which we performed later.

4.2.2 ResNet50

After trying to use VGGNet, we decide to switch to ResNet which is more faster in extracting the features than VGGNet, this thanks to the two types of shortcut connections introduced mainly to solve the vanishing gradient problem.

As we have done for the VGGNet, we decide to maintain unaffected the ImageNet learned weights and subsequently use them as a feature extractor for images.

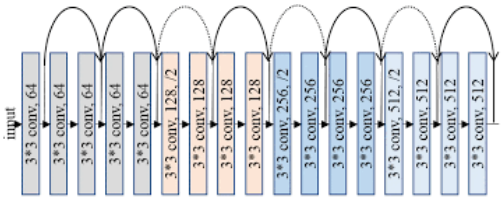

Figure 3. Representation of ResNet.

4.3. Recurrent Neural Network

Our next model used Recurrent Neural Networks (RNNs) to capture the temporal information from the data. RNNs have the ability to maintain state information that gets updated with every input in a sequence, making them a suitable choice for working with sequential data.

4.3.1 RNN

Initially, we attempted to add and then train some RNN layers on top of the pre-trained CNN, but the results were not as expected, so we decided to quickly abandon this idea and explore other, more complex architectures such as LSTMs and GRUs.

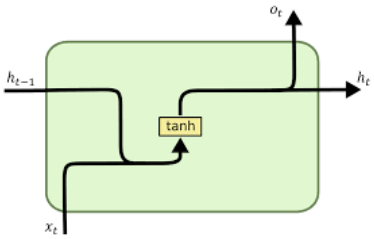

Figure 4. Reperesentation of a RNN cell.

4.3.2 LSTM

We implemented some Long-Short Term Memory (LSTM) layers to address the vanishing gradient problem that we encountered in our simple RNN, which occurs when the gradients of the weights in the network become very small and the network has difficulty learning. LSTM networks are a type of RNN that use a special kind of memory cell to store and output information. These memory cells are designed to remember information for long periods of time, and they do this by using a set of gates that control the flow of information into and out of the cell. The gates in an LSTM network are controlled by sigmoid activation functions, which output values between 0 and 1. The gates allow the network to selectively store or forget information, depending on the values of the inputs and the previous state of the cell. In this model, we fed the images through the VGGNet in our first approach and ResNet in the second one, both of which pre-trained on ImageNet dataset. The features extracted by the final CNN layer were used as input by the first LSTM layer to capture the temporal information from the sequences. The final LSTM layer utilized a softmax classifier to generate the label which was subsequently compared with the ground truth.

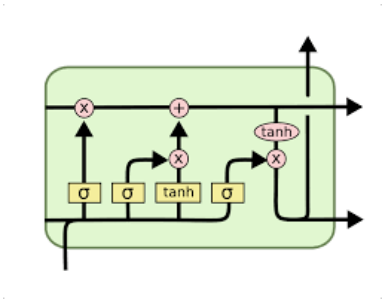

Figure 5. Reperesentation of a LSTM cell.

Since both the CNNs were pre-trained on ImageNet, we froze their weights and only train the LSTM layers. This essentially involved doing a single forward pass through the CNNs, and then reusing the extracted features as inputs to the LSTM for each epoch without updating the CNNs. Further, we employed batch normalization for quicker convergence and dropout for regularization. We also used shuffling with the purpose of reducing variance, and making sure that models remain general and overfit less.

4.3.3 GRU

We implemented some Gated Recurrent Unit (GRU) layers due to their advantages over traditional RNNs, such as not suffering from the vanishing gradient problem and moreover, they are simpler to implement and faster to run than the LSTM cells. GRU networks are a type of RNN that use a special kind of memory cell to store and output information. They use a single update gate to control the flow of information into the memory cell rather than three gates used in LSTMs. The update gate determines which information derives from the previous hidden state and which current input keep, moreover they use a reset gate to determine which information to discard. This makes GRUs easier to train and faster to run than LSTMs but they may not be as effective at storing and accessing long-term dependencies. The procedure for implementing GRU layers was the same as for LSTMs layers. We used the features extracted from the VGGNet in our first approach, and from ResNet in the second approach, as input to the first GRU layer in order to extract temporal information from the sequence. The final label was generated using a softmax classifier, which was compared with the ground truth to compute the metrics.

We also employed batch normalization to achieve faster convergence, shuffling to reduce variance and prevent over-

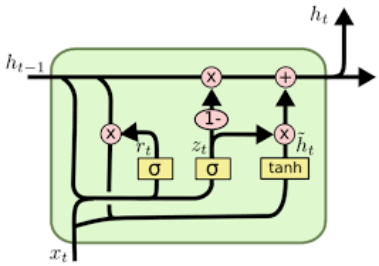

Figure 6. Reperesentation of a GRU cell.

fitting, and dropout for regularization.

5. Experimental Results

In the experiments, we trained different models with different architectures and compared their performance in terms of accuracy and top-5 accuracy. The Adam Optimizer with its default learning rate was used in all the models. The results are summarized in the following table:

The single-validation accuracies of each model can be seen in Figures 7, 8, along with the score of the state of the art SVM model which reaches 62.1% and 69.7% of accuracy for words and phrases respectively. With the 10 word classes, it can be seen that the behavior of the various models that implement RNNs is generally worse than the model that uses machine learning techniques. Despite this we managed to achieve similar accuracy regarding the speak independent configuration for visual speech recognition. In the speaker independent experiment, the training and the query data are from different speakers. We employ the leave-one-out strategy where data from a single speaker are used as the validation data, and the remaining speakers are used as the training data.

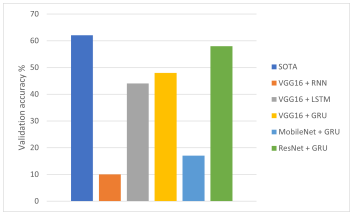

Figure 7. Test accuracy on words of various models.

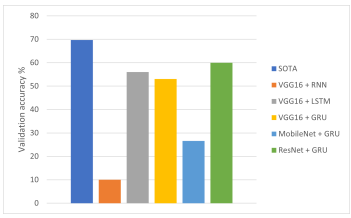

Figure 8. Test accuracy on phrases of various models.

The results from the best image model are shown in Table 2. We have achieved best test accuracy of 58% and 60% for words and phrases respectively.

	Training	Test
On words	84.3%	58%
On phrases	87.75%	60%

Table 2. ResNet50 + GRU results.

When comparing the accuracy of words and phrases, we notice that phrases have a higher accuracy as compared to words. We can attribute this to phrases being long and thus having a more data to differentiate between them whereas words are short and it would be more difficult to differentiate between them.

From the results, we can see that the ResNet50 + GRU model achieved the highest accuracy and top-5 accuracy, while the VGG16 + RNN model performed the worst. It is also noteworthy that the GRU models generally performed better than the LSTM and RNN models.

These results suggest that the combination of ResNet50 and GRU is a promising architecture for the task, while the choice of architecture can have a significant impact on the performance of the model. Further experiments and tuning can be conducted to further improve the results.

Figure 10 show the training loss with epochs. Although loss has not yet converged after training completes, we we already see a sort of stabilization in the accuracy (in Figures 9) and therefore we can expect that the model would not have improved much more compared to the one we obtained.

Figures 11 and 12 give the confusion matrices for the best model evaluated on train and test set. The matrices are quite informative in the kind of errors made. For instance, the word “Start” and the word “Stop” are getting mixed up in the test set. This is because the test set speaker’s mouth

Model	Test Accuracy	Top-5 Accuracy
VGG16 + RNN	10%	50%
VGG16 + LSTM	44%	88%
VGG16 + GRU	48%	91%
MobileNet + GRU	17%	71%
ResNet50 + GRU	58%	96%

Table 3. Words result for MIRACL-VC1 dataset.

Model	Test Accuracy	Top-5 Accuracy
VGG16 + RNN	10%	49%
VGG16 + LSTM	56%	95%
VGG16 + GRU	53%	87%
MobileNet + GRU	26.6%	82%
ResNet50 + GRU	60%	92%

Table 4. Phrases result for MIRACL-VC1 dataset.

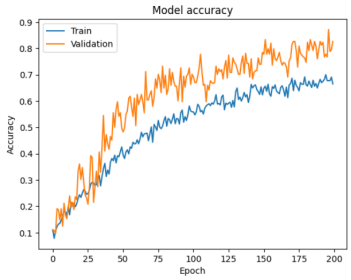

Figure 9. Accuracies over epochs.

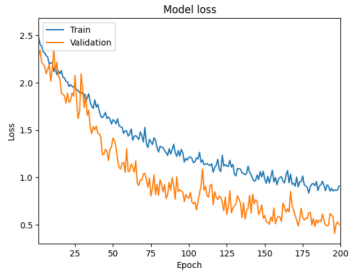

Figure 10. Training loss over epochs.

movement for both of them is quite similar, which is not true for the speaker in validation set. This reveals another

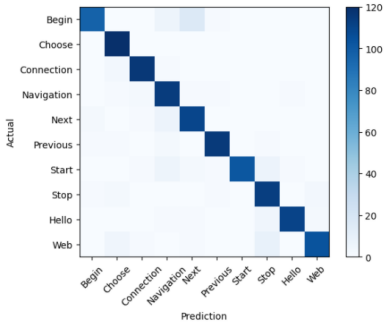

Figure 11. Confusion matrix of words on train set.

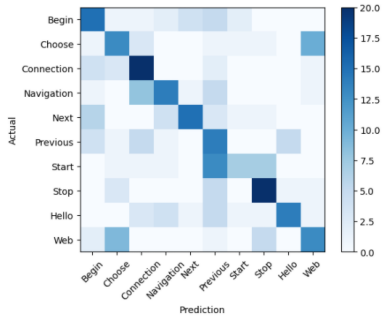

Figure 12. Confusion matrix of words on test set.

problem, that is difference in accents. People from different parts of the world have different ways of pronouncing the same word. This creates problems for a model that uses lip orientation to figure out the spoken word.

Similar problems can be individuated also considering the phrases inside the Miracl-VC1 dataset as shown in the confusion matrices in Figures 13 and 14. As we can see the sentence "Have a good time" is completely misclassified and identified as "Excuse me" and "I love this game".

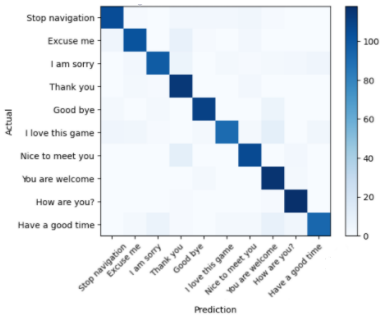

Figure 13. Confusion matrix of phrases on train set.

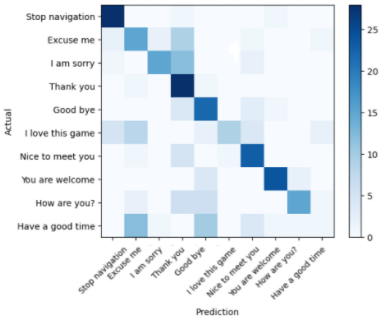

Figure 14. Confusion matrix of phrases on test set.

We also experimented with different parameter update strategies. We noticed that SGD itself was incapable of training the model in reasonable time. It gave no significant improvements even after 20 epochs. In comparison, Adam showed improvements right from the first epoch.

6. Conclusion

In this report, we have presented our work on lip reading, which is the task of transcribing speech from video sequences of a speaker's mouth.

our experiments showed that using pre-trained CNNs as feature extractors on the lip reading task using RNN models that capture the temporal information from the video sequences is a good starting point. Generally the performances are not the best also because our data augmentation did not provide any independent data, so this could not have provided a significant boost.

In future work, we aim to further improve the performance by using more advanced models that better capture the complex relationships between the visual information from the lips and the corresponding speech.

References

[1] C. Bregler and Y. Konig. Eigenlips for robust speech recognition. In *Proceedings of ICASSP '94. IEEE International Conference on Acoustics, Speech and Signal Processing*, volume ii, pages II/669–II/672 vol.2, 1994.

[2] Yuru Pei, Tae-Kyun Kim, and Hongbin Zha. Unsupervised random forest manifold alignment for lipreading. In *2013 IEEE International Conference on Computer Vision*, pages 129–136, 2013.

[3] Ahmed Rekik, Achraf Ben-Hamadou, and Walid Mahdi. A new visual speech recognition approach for RGB-D cameras. In *Image Analysis and Recognition - 11th International Conference, ICIAR 2014, Vilamoura, Portugal, October 22-24, 2014*, pages 21–28, 2014.

[4] Ahmed Rekik, Achraf Ben-Hamadou, and Walid Mahdi. Human machine interaction via visual speech spotting. In Sebastiano Battiato, Jacques Blanc-Talon, Giovanni Gallo, Wilfried Philips, Dan Popescu, and Paul Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, pages 566–574, Cham, 2015. Springer International Publishing.

[5] Ahmed Rekik, Achraf Ben-Hamadou, and Walid Mahdi. Unified system for visual speech recognition and speaker identification. In Sebastiano Battiato, Jacques Blanc-Talon, Giovanni Gallo, Wilfried Philips, Dan Popescu, and Paul Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, pages 381–390, Cham, 2015. Springer International Publishing.

[6] Ahmed Rekik, Achraf Ben-Hamadou, and Walid Mahdi. An adaptive approach for lip-reading using image and depth data. *Multimedia Tools and Applications*, 75(14):8609–8636, 2016.

[7] Michael Wand, Jan Koutník, and Jürgen Schmidhuber. Lipreading with long short-term memory, 2016.