QUEEN'S UNIVERSITY BELFAST

Multimodal Phrase Recognition With Neural Language Models

A dissertation submitted in partial fulfilment of

the requirements for the degree of

BACHELOR OF ENGINEERING in Computer Science

in

The Queen's University of Belfast

by

Jason Burgess

12/05/2022

SCHOOL OF ELECTRONICS, ELECTRICAL ENGINEERING and COMPUTER SCIENCE

CSC3002 – COMPUTER SCIENCE PROJECT

Dissertation Cover Sheet

Work submitted without a cover sheet will **NOT** be marked.

Student Name Jason Burgess Student Number: 40233633

Project Title: Multimodal Phrase Recognition With Neural Language Models

Supervisor: Dr Barry Deveraux

Declaration of Academic Integrity

Before submitting your dissertation please check that the submission:

1. Has a full bibliography attached laid out according to the guidelines specified in the Student Project Handbook
2. Contains full acknowledgement of all secondary sources used (paper-based and electronic)
3. Does not exceed the specified page limit
4. Is clearly presented and proof-read
5. Is submitted on, or before, the specified or agreed due date. Late submissions will only be accepted in exceptional circumstances or where a deferment has been granted in advance.

By submitting your dissertation you declare that you have completed the tutorial on plagiarism at http://www.qub.ac.uk/cite2write/introduction5.html and are aware that it is an academic offence to plagiarise. You declare that the submission is your own original work. No part of it has been submitted for any other assignment and you have acknowledged all written and electronic sources used.

Student's signature Jason Burgess *Date of submission* *12/05/2022*

Acknowledgements

I would like to acknowledge my supervisor, Dr Barry Deveraux for his guidance and assistance

Abstract

Machine lipreading is a problem that can be separated into two distinct parts: Firstly identifying visual movements and secondly constructing phrases from these identified visemes. While high performance has been achieved in the first of these two parts, advancements in the latter have been lacking, resulting in loss of system performance due to poor viseme to word conversion models. In this paper we will evaluate a three novel approaches to this issue by using a language model based on the BERT transformer architecture to convert sequences of visemes to English sentences

1 Background And Problem Description

1.1 The Value of Machine Lip Reading

Automated systems capable of accurately reading lips is something that many have considered to be more grounded in science fiction than reality, perhaps most notably in the scene from Stanley Kubrick's film adaptation of Arthur C. Clarke's science fiction novel 2001: A Space Odyssey. However, recent advancements in deep learning and neural networks have brought these ideas closer to reality, with a recent paper[6] scoring a word accuracy rate of 79.6% on the BBC's LRS2 dataset [4] - a dataset consisting of video snippets of people talking from a variety of camera angles labelled with their subtitles, gathered from footage of BBC TV shows. Machine lipreading is something that if done accurately, has many real world use-cases in a variety of industries such as healthcare or mobile digital assistants. The first occurrence of lipreading technology in the healthcare industry was unveiled recently with Liopa being awarded a licence to use their lip reading mobile app SRAVI(Speech Recognition For The Voice Impaired) in UK hospitals. This software allows patients who have underwent tracheotomies or other voice-affecting procedures to communicate with hospital staff by "mouthing" the phrase they are trying to communicate. This is one example of how a successful lipreading system can be of great value and can improve the lives of others. Machine lipreading can also be used to improve the transcription accuracy of any automatic speech recognition(ASR) system, particularly in noisy environments such as places with loud machinery or outdoors in windy settings.

1.2 Phonemes And Visemes

A phoneme is defined as the smallest unit of speech that can distinguish one word from another in a particular language. According to the International Phonetic Alphabet(IPA,) the English language contains 44 unique phonemes. Visemes however, are not as well researched and have yet to be formally defined. In the context of this project we will define a viseme as a set of visually indistinguishable phonemes. That is, if two phonemes cannot be distinguished by their associated lip movements then they are considered to be part of the same viseme set. There exists many different phoneme-to-viseme mappings, with this paper[2] mentioning 120 unique mappings. In section 2.2.3 we will examine existing literature to decide which mapping to use for this project If every word in the English dictionary can be expressed as sequence of phonemes, then it can also be expressed as a sequence of Visemes. This is useful in the context of machine lipreading with no audio input, as a computer vision system trained to classify phonemes would not be able to correctly classify the phonemes of phrases with the same or very similar lip movements, for example "I love you" and "Elephant juice" are very different phonetically but to a computer vision system (or atleast a human level vision system) look very similar leading to poor classification, whereas these phrases would be more accurately classified if they were classed as viseme sequences rather than phoneme ones. However, using viseme sequences to represent words and phrases is not without its complications. For one, visemes are largely speaker-dependant. What is meant by this is that lip movements may vary widely amongst individuals, especially amongst people with differing accents, just as pronunciation does. [1] proposes a system for generating speaker-dependant visemes, however this is not the focus of this project.

1.3 The Problem Of Ambiguity

Despite the great variations of phoneme-to-viseme mappings suggested and their methods of creation, they all have one similarity; A many-to-one relationship of phoneme to viseme. Whenever trying to decode a viseme sequence to an English word or phrase a rather glaring complication arises: Multiple words can map to the same viseme sequence. This is due to the loss of information caused by the many-to-one nature of phoneme-to-viseme mappings. This ambiguity is the problem that this paper is trying to address. This ambiguity or 'fuzziness' is not a problem as unique as one might initially believe, and actually appears in English as well. Consider the word "lie" in the following two sentences: "The jury believed the defendant was telling a lie", and "After a long day of work they decided to lie down for a while". In these two sentences the word "lie", written (and pronounced) exactly the same has been used to mean two different things. This by definition is ambiguity, however English speakers will have no trouble inferring the correct meaning of this word in each sentence despite there being a many-to-one relationship of meaning to word. How is this so? By looking at the surrounding words in the sentence, aka the context. We believe the same idea can be applied to a language consisting of visemes by using a neural language model capable of learning the contextual relationships between visemes.

1.4 Attention And the Transformer Architecture

In 2017, a new neural network architecture called the transformer was introduced in the highly-influential paper [11]. In the paper they showcase a transformer model that achieves sate-of-the-art results on a number of NLP tasks including machine translation. The key component of the transformer model is the attention mechanism, which we will now explain in brief detail. The attention mechanism consists of query, key, and value vectors. Each token in the input sequence gets it's key-value pair from linear transformations of the token's input embeddings. For any given query vector the dot product of the query and each input token's key is calculated, with a larger dot product representing more similarity between the query and the key. After being scaled then normalized by a softmax function the resulting vector can be used as a weighting of each key. This weighting of the keys is then used to sum each key's corresponding value vector. This is the output of the attention mechanism, which is used to update the vector representation of the query token. This is done in parallel for each token in the input sequence in a process known as multi-head attention. The attention mechanism was not introduced in the transformer paper and has existed as a component in other previous network architectures. What makes transformers unique is their sole reliance on the attention mechanism for token representation, dispensing with recurrence and convolution, two prominent features of previous state-of-the-art models. Relying solely on attention allows sequences to be processed in parallel rather than sequentially which greatly reduces training time

1.5 BERT And The Rise Of Transfer Learning

In 2019 Google released BERT [5]. This model took the encoder part of the transformer architecture and trained it on a massive corpus of unlabelled text data scraped from the internet. The encoder-only model was trained on two tasks: 1)Masked language modelling (MLM), where 15of input tokens are randomly masked and the model must predict the masked token, 2)Next sentence prediction (NSP), where a pair of sentences are fed to the model as inputs and it must predict if the two sentences follow each other or not. The result is a model able to produce deep, bidirectional representations of words in a language. They demonstrate how fine-tuning a single additional output layer on a specific task was able to make use of the strong language understanding to achieve state-of-the-art results. This process is known as transfer learning and has become a very popular method for achieving great results on a variety of NLP tasks such as question answering or named entity recognition. Making use of a pretrained model greatly reduces training time and cost, as a pretrained model already has strong contextual embeddings of the relationships between words. However in the case of this project we will pretrain our own BERT-like model on a large amount of unlabelled data consisting of viseme sequences, rather than English. We hope that this model will be able to learn the relationship between visemes, and we plan on fine-tuning this encoder on the task of converting viseme sequences to English.

1.6 Related Works

While no known paper has used a BERT-like model for this task, a number of similar methods have been investigated. [7] used perplexity values calculated by GPT to deem which word was

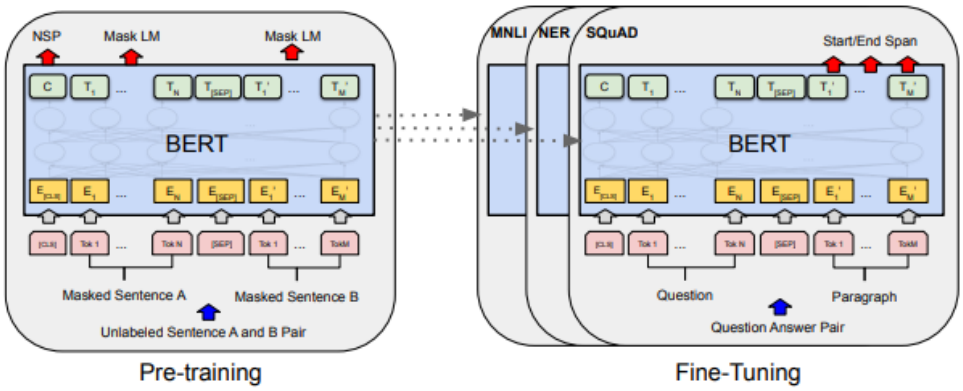

Figure 1: Overview of the BERT pretraining and fine tuning process. Apart from output layers, the same architectures are used in both pre-training and fine-tuning [5]

the most likely given the previous words from the output sequence. The conclusion mentions that it is this viseme to word conversion that is bottle necking the system with a very high viseme classification accuracy of 95rate of only 65.5% due to poor viseme-to-word conversion: "As such, it is important to explore any other possible approaches for the conversion". Another recent paper, (An Effective Conversion of Visemes to Words for High-Performance Automatic Lipreading, 2021) Investigated multiple methods of viseme to word conversions and found that an attention based gated recurrent network performed the best, allowing them to improve the state-of-the-art of the LRS2 Dataset. It is worth noting that in their conclusion they mentioned the possible value of transformer models, stating "There is also merit in considering the use of either Attention-Transformers or Temporal Convolutional Networks as conversion models because they can process inputs in parallels as opposed to RNNs, which process inputs sequentially". Both of these papers highlight the usefulness of a more accurate viseme-to-word conversion model.

1.7 Societal Implications

There are important ethical and societal implications to consider when developing lipreading software. While there are most certainly positive use cases for machine lipreading, for example Liopa's use case, bad actors could use this technology for evil. Like any new technology, machine lipreading is not inherently "good" or "bad", but has the potential to be used for both. This can be seen even in one of humanities first discoveries, fire which allowed us to cook our food and provided warmth and shelter, but it also has caused great amounts of suffering throughout history. Certain states could use this technology as a form of surveillance to suppress free speech. While this technology is still some while away from being able to accurately transcribe speech from low quality CCTV footage where the the target is not directly facing the camera and consciously making an effort to have their lip movements visible to the camera, it is not unreasonable to imagine a situation in the near future where this technology could be used to detect specific word or phrases deemed "undesirable" by the bad actor. Given the exponential growth of new technologies such as deep learning and their ability to do things to a superhuman level most would have considered impossible in the past, it is important to consider how this technology could be used for evil in the wrong hands.

2 Methodology

2.1 On The Existence Of Words

We will pretrain a transformer model on a large amount of unlabeled viseme sequences before fine tuning the pretrained model on the task of identifying the words in a given viseme sequence. First, we must identify what form the input viseme sequence should take. The existence of words in continuous speech is a debated topic among linguists. Some linguists argue that when observing someone speaking it is impossible to determine the point at which one word ends and another begins without prior knowledge of the words being spoken, as there is no gap between spoken words, in contrast to written words which are separated by a space. For example, the phrase "some times" and the word "sometimes" are distinguished in written form by the space in the

latter, but when spoken by a person they are indistinguishable, unless the speaker deliberately pauses in between words. Therefore, when performing viseme to sentence conversion it may be unreasonable to expect the viseme identifications used as input to the system to have gaps in between words, unless the speaker takes deliberate pauses between words. For our experiments we will investigate both scenarios, and see how the absence of spaces affects the models ability to correctly identify words. We will also investigate a third approach that identifies phonemes in the viseme sequence, rather than words. These three approaches are described in more detail in the following sections

2.2 Three Modelling Approaches

2.2.1 Word Classification Of Space-Separated Viseme Sequences

One approach we will investigate will be fine tuning a model pretrained on space-separated viseme sequences on word classification, with each token being the cluster of visemes representing that word. It is expected that this model will give us the best metrics, but it may be unsuitable for real world use due to the reasons discussed in the above section. The main goal of this model is to serve as a benchmark for the following two, which are much more applicable in a real-world two stage machine lipreading system

2.2.2 Word Classification Of Viseme Sequences

This approach is similar to the above, except there are no spaces between words. This scenario is more suitable to real-world applications, but will undoubtedly be more challenging due to the loss of information of spaces between words.

2.2.3 Phoneme Classification Of Viseme Sequences

A third classification model was developed, with classification done at the viseme character-level, where instead of having every word in the dictionary as a label/tag, we instead use the 44 English phonemes as tags. After the model has predicted a phoneme sequence, we can devise a lookup method to convert the phoneme sequence to a sequence of words, given a lexicon of phonetic transcriptions generated from the eng-to-ipa library. One thing to consider however is multiple words with the same phonemes (homophones) for example "hours" and "ours". In section 2.9 we provide our solution to this problem

This method would not have to use any sort of vocabulary limitation like the previously described one, and in theory should be easier to train due to a much smaller label set requiring less data and compute power, both of which are limited for this experimental project

2.3 Choice Of Dataset

Pretraining of large language models requires a large dataset of unlabelled text data for the model to develop strong contextualised word embeddings. The original BERT paper[5] was pretrained on mostly on the Wikipedia corpus. For our experiments we have chosen to train on the oscar dataset [10]. there are two main reasons for this. 1) [10] states that language models trained on the OSCAR dataset perform better than those trained on the wiki corpus. 2) Accessibility: The oscar dataset is available via the huggingface datasets hub, which allows for the streaming of datasets. This means a very large dataset does not have to be downloaded on the machine, as partitions of it can be streamed and accessed as required. This leads to a huge save of compute cost and time when generating data for training of models

2.4 Choice Of Phoneme-To-Viseme Mapping

Visemes are still not formally defined, and there is certainly no consensus of a phoneme to viseme mapping similar to what the International Phonetic Alphabet has done for phonemes. This leaves it up to the authors to decide which phoneme to viseme mapping to use, so we consulted the literature. [2] found that out of the 120 mappings studied, the mapping devised by Lee [8] performs the best for word classification. This mapping was also found to be the best for machine lipreading in [3]. Because the literature seems to indicate that this mapping has been shown to work well for machine lipreading, we have chosen to use it as our mapping for this project

2.5 Pretraining Data Generation

2.5.1 Data Cleaning

Data entries streamed from the oscar dataset were split into sentences. Any sentences containing punctuation or digits were removed. The reason for this was the eng-to-ipa library used does not provide phonetic representations for digits, and simply removing digits from sentences would damage the quality of the data through loss of meaning. For example, the sentence "I bought it for 14 pounds" would become "I bought it for pounds". This sentence does not make sense, and including examples like this might damage the models contextual embeddings. Punctuation was removed to better model spoken sentences, which do not contain punctuation

2.5.2 Word To Phoneme Conversion

After this, these cleaned sentences are passed to eng-to-ipa to get the phonetic transcription of each sentence from the CMU pronouncing dictionary. If a word in the sentence does not have a phonetic transcription in the CMU dictionary, the entire sentence is dropped, not just the word for the same reason mentioned above regarding digits

2.5.3 Phoneme To Viseme Conversion

From the phonetic representation of each sentence we then use Lee's viseme map to convert each phonetic sequence to a sequence of visemes. In the case of pretraining the space-separated word-level model, the viseme sequences of each word are separated by a space. In the case of the other two models, the spaces are removed, leaving a sequence of characters with each character representing one viseme

2.6 Pretraining

A masking algorithm was developed that randomly masks 15% of the non-special tokens in every input sequence by replacing them with a mask token. The goal of the model during training is then to try and predict the masked token. This is how the model is able to learn the contextual relationship between tokens and is the core of BERT's ability to learn about a language. Dynamic masking as described in the roBERTa paper[9] was applied, as this paper states that dynamic masking improves the models performance when fine tuned on downstream tasks. Dynamic masking is where the masking is done at the beginning of each epoch, rather than at the data preproccessing stage. This means the masks will be different for each sequence every epoch, meaning the model has more chances to learn contextual information about the language

The original BERT architecture also used a second training goal known as next sentence prediction, this is when a pair of sentences is fed to the model simultaneously and it must decide based on the similarity of the word representations if the sentences appear one after another in the training data or not. It was theorized that this task would help the model learn about inter-sentence context. but the roBERTa paper[9] reported that this task had a negligible effect on the models ability to understand language. It is for this reason that our model is not trained on this task

The hyperparameters used for our pretraining were largely similar to those used in the roBERTa base model. We implemented a model with 10 hidden layers (two less than roBERTa base), each with 768 dimensions and 12 attention heads. A learning rate of 1e-5 was used with the AdamW optimizer. Models were trained on 100k viseme sequences for 15 epochs, and training typically took around 280 minutes. A batch size of 16 was originally used, but increasing this to 32 led to improvements on downstream tasks. An attempt was made to increase the batch size again to 64, but this was not possible due to GPU memory limitations

2.7 Word Classification Of Space-Separated Viseme Sequences

Viseme sequences were generated the same way as described in section 2.5, with each group of visemes labelled with an id representing its respective word. All words in the training data were ranked in terms of frequency, with those falling outside the top N being labelled as OOV. Three versions of the data were created, for N=1k, 5k, 10k to determine how well the model performs with varying vocabulary sizes. A word2id dictionary was also generated for converting the predicted word ids back to words, to be used for model evaluation and inference.

2.8 Word Classification Of Viseme Sequences

The data generation for word level tagging on non-spaced viseme sequences follows the same pattern as above, also with the top 1k, 5k and 10k words. The main difference is as there are no spaces indicating gaps between words, each viseme character is labelled individually with the id of the word it belongs to.

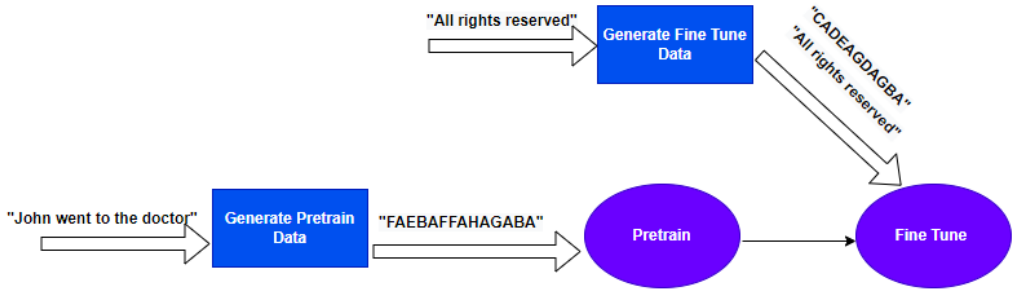

Figure 2: Training Pipeline for Word Classification Of Viseme Sequences

2.9 Phoneme Classification Of Viseme Sequences

The process for generating the data for phoneme classification differs from the above two. Each character in the viseme sequence is labelled with its respective phoneme. A lexicon of the phonetic representation of every word in the training data is generated, so that identified phonemes can be mapped to English words. One issue with this approach is words that have the same phonetic representation (homophones), e.g "site" and "sight". This was handled by adding the word that was more frequent in the training data, for example "gym" appeared more frequently than "Jim", so every time the phonemes for this homophone were detected, "gym" was predicted by the model. This approach will impact the models ability to correctly classify words and alternative methods for handling these cases should be investigated, as discussed in (REFERENCE). However, simply adding every word that matches a phonetic representation in the lexicon is not sufficient, as the same phoneme sequence could map to many words. For example the phonemes for "restaurant" also contains the phonemes for "rest" and "are", and each predicted phoneme can only match to a single word. The solution to this was the following algorithm:

1: Get all matches in the phoneme2word lexicon, and their start and end positions
2: Starting at the beginning of the list of matches, for every match, get all matches that overlap with it and add the longest of these overlapping matches to wordPredictionsFront
3: Starting at the end of the list of matches, for every match, get all matches that overlap with it and add the longest of these overlapping matches to wordPredictionsEnd
4: Check for any predicted phonemes not being used in wordPredictionsFront, and check if any matches in wordPredictionsEnd contain these unused phoneme characters. If so, add these to finalPredictions
5: Add all matches from wordPredictionsFront that do not overlap with matches already in finalPredictions to finalPredictions

This algorithm is not perfect and does not work for 100% of cases, but was found to perform best on the test set out of all other algorithmic methods experimented with. When the ground truth phonemes from the test data are given to this algorithm the sequences output have a 91% word accuracy, meaning this is the best possible performance we can expect from our phoneme tagging model on the test set. One advantage of the phoneme-level method is the vocabulary size is only limited to the number of unique words in the training data (minus the less-frequent homophones), as we can add phonetic representations for every word in the dataset without increasing the number of classes the model has to predict, as words are inferred from the output of the model, not by the model itself

2.10 Fine Tuning

All three types of models were trained very similarly for their respective tasks, each model was trained on 10k entries and trained for a maximum of 10 epochs. Early stopping was implemented

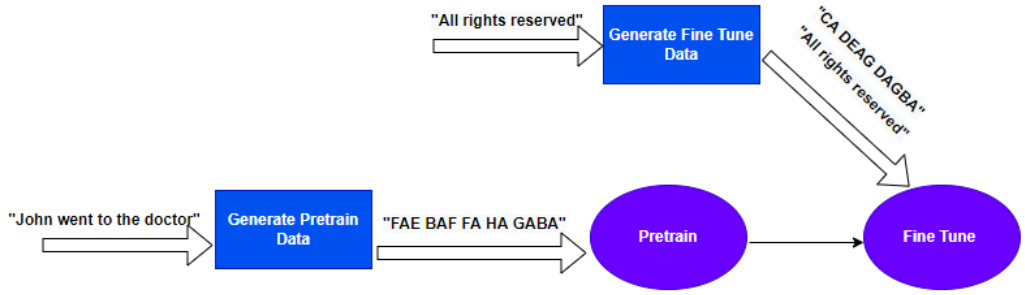

Figure 3: Training Pipeline for Word Classification Of Space-Separated Viseme Sequences

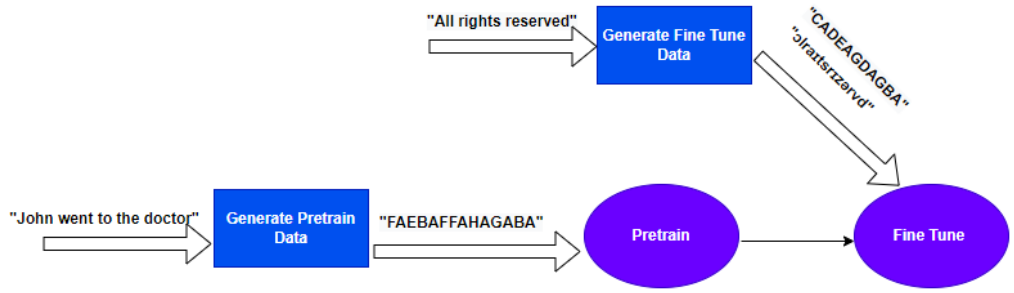

Figure 4: Training Pipeline for Phoneme Classification Model

with training stopping early if the loss on the validation split failed to decrease for 2 consecutive epochs. Learning rate was set at 1e-5 with the AdamW optimizer. Increasing batch size from 16 to 32 improved performance of all finetuned models on the validation split of the training data

2.11 Evaluation

2.11.1 Test Set

In order to compare all three model types fairly, a single test set of viseme sequences was generated using data from a portion of the oscar dataset unseen by any of the models at either pretraining or finetuning stage. The viseme sequences were used to create the data needed for each model type, labelling each character with its phoneme for the phoneme level tagging model, labelling each character with the word it belongs to for the word level tagging without spaces model, and labelling each space-separated viseme cluster with its associated word for the word level tagging of space-separated viseme sequences model. This allows each model to be evaluated on the viseme representations of the same English sentences. For the two model types with a vocabulary of top N words, all words falling outside of the top N were labelled as OOV

2.11.2 Evaluation Metrics

All three models were evaluated using average word accuracy and average sentence accuracy, the formulas for which can be seen below:

$$S_{accuraccy} = \frac{S_{correct}}{S_{correct} + S_{incorrect}}$$

Where $S_{correct}$ is the number of sentences correctly predicted and $S_{incorrect}$ is the number of sentences incorrectly predicted

$$W_{accuraccy} = \frac{W_{correct}}{W_{correct} + W_{incorrect}}$$

Where $W_{correct}$ is the number of words correctly predicted and $W_{incorrect}$ is the number of words incorrectly predicted

The Phoneme prediction model will also be evaluated on Phoneme Accuracy:

$$P_{accuraccy} = \frac{P_{correct}}{P_{correct} + P_{incorrect}}$$

Where $P_{correct}$ is the number of phonemes correctly predicted and $P_{incorrect}$ is the number of phonemes incorrectly predicted

2.12 Technologies Used

All code was written in python as notebooks on Google Colab. Model training was done using the GPUs available on google colab+. SKlearn was used for splitting of datasets into train and validation splits. Data was streamed using Huggingface's Datasets library. Modelling architecture was implemented using the Transformers Library, and model training was implemented using pytorch. Phonetic transcriptions of words were accessed using the eng-to-ipa python library, which uses the CMU pronunciation dictionary. Models and data were saved in Google Drive

3 Results And Analysis

Top N Words	Avg. Word Accuracy	Avg. Sentence Accuracy
1k	81%	21%
5k	70%	11%
10k	57%	7%

Table 1: Word Classification Of Space Separated Viseme Sequences Model

Top N Words	Avg. Word Accuracy	Avg. Sentence Accuracy
1k	4%	0%
5k	4%	0%
10k	4%	0%

Table 2: Word Classification Of Viseme Sequences Model

As expected, our best performing model was the Word Classification Of Space Separated Viseme Sequences Model, indicating how much better the model performs when it knows where each word begins and ends. The surprisingly poor performance of the Word Classification Of Viseme Sequences Model in comparison further demonstrates this point. The strong performance of the Word Classification Of Space Separated Viseme Sequences Model indicate that this type of model is a viable candidate for a viseme to word conversion system, given of course the visemes are seperated by word. This could be achieved as long as the speaker takes a gap of perhaps half a second in between spoken words to allow the video to viseme model to recognise the beginning and end of words.

The Phoneme Classification With Space Separated Model performs fairly well on the task of correctly identifying phonemes, but performance drops significantly during phoneme to word conversion. This can partially be explained by the need to correctly predict multiple phonemes in a row to be able to infer the correct word, but the phoneme to word conversion algorithm has room for improvement.

On the issue of homophones, a number of different methods could be used. The context of surrounding words could be used to decide which homophone is most appropriate. This could be achieved using a skipgram or continuous bag of words model trained on sentences containing said homophones. The entire phoneme to word conversion process could be replaced by another neural language model trained on this task, perhaps similar to the way our models have been trained. A hidden markov model might also be appropriate.

Despite this models worse result in comparison to the best performing model, it is important to consider this model does not have as limited a vocabulary as the other two, its phonetic lexicon contains the phonetic representation of all words used in the training data, and can easily be expanded upon using the eng-to-ipa library. This means the model can be adapted for use in different text domains, outside of what it was trained on, by adding new words to the phonetic lexicon. This makes this the most flexible of the three models. Given the rate of decrease of word

Avg. Phoneme Accuracy	Avg. Word Accuracy	Avg. Sentence Accuracy
68%	26%	5%

Table 3: Phoneme Classification With Space Separated Model

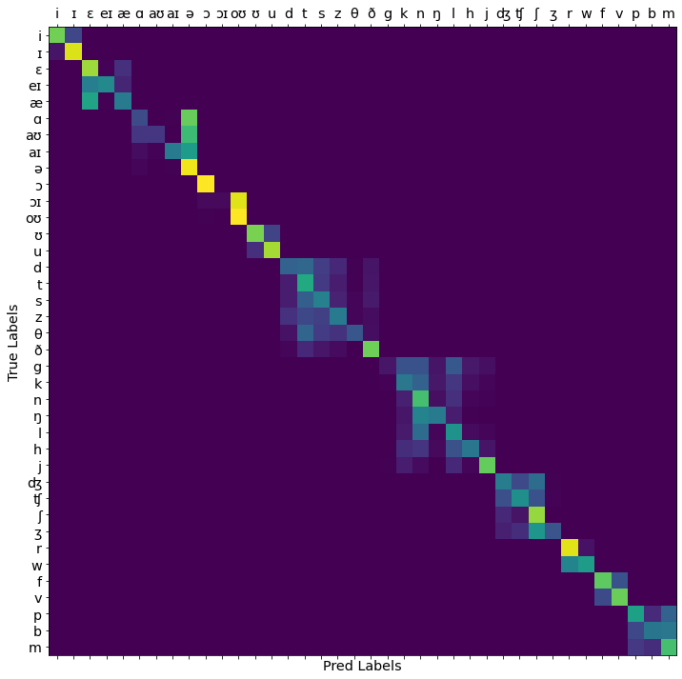

Figure 5: Confusion matrix of phoneme classification model.

Accuracy of the Word Classification Of Space Separated Viseme Sequences Model, for very large vocabularies the Phoneme model may perform better.

4 Conclusion

In this paper we investigated three approaches for viseme to word conversion using pretrained transformer models. Our main takeaways were that if at all possible to receive data in the required format, word classification of space-separated viseme sequences is a suitable option, especially for limited vocabularies(<5k words). If data cannot be received in this fashion, phoneme level classification is a viable (albeit less accurate) alternative. Further work could be done to improve the phoneme to word conversion rate, as this is the bottleneck of the system. One approach would be to pretrain a transformer model on large amounts of unlabelled phonetic representations and fine tune on word classification, similar to what was done for visemes here. We also found that even with large amounts of training data, classifying continuous viseme sequences as words directly is not a viable option, even with limited vocabulary of 5k words

References

[1] Helen Bear, Stephen Cox, and Richard Harvey. Speaker-independent machine lip-reading with speaker-dependent viseme classifiers. 09 2015.

[2] Helen L. Bear and Richard W. Harvey. Phoneme-to-viseme mappings: the good, the bad, and the ugly. *CoRR*, abs/1805.02934, 2018.

[3] Helen L. Bear, Richard W. Harvey, Barry-John Theobald, and Yuxuan Lan. Which phoneme-to-viseme maps best improve visual-only computer lip-reading? In George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Ryan McMahan, Jason Jerald, Hui Zhang, Steven M. Drucker, Chandra Kambhamettu, Maha El Choubassi, Zhigang Deng, and Mark Carlson, editors, *Advances in Visual Computing*, pages 230–239, Cham, 2014. Springer International Publishing.

[4] Joon Son Chung, Andrew W. Senior, Oriol Vinyals, and Andrew Zisserman. Lip reading sentences in the wild. *CoRR*, abs/1611.05358, 2016.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[6] Souheil Fenghour, Daqing Chen, Kun Guo, Bo Li, and Perry Xiao. An effective conversion of visemes to words for high-performance automatic lipreading. *Sensors*, 21(23):7890, Nov 2021.

[7] Souheil Fenghour, Daqing Chen, Kun Guo, and Perry Xiao. Lip reading sentences using deep learning with only visual cues. *IEEE Access*, 8:215516–215530, 2020.

[8] Soonkyu Lee and Dongsuk Yook. Audio-to-visual conversion using hidden markov models. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 2417 of *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pages 563–570. Springer Verlag, 2002. 7th Pacific Rim International Conference on Artificial Intelligence, PRICAI 2002 ; Conference date: 18-08-2002 Through 22-08-2002.

[9] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[10] Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. A monolingual approach to contextualized word embeddings for mid-resource languages. *CoRR*, abs/2006.06202, 2020.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.