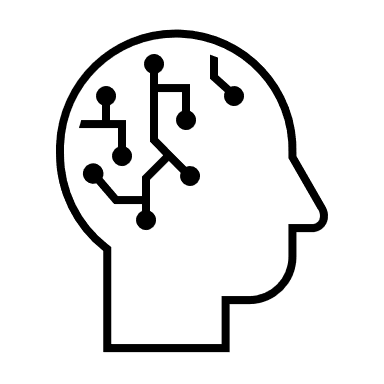
**FINAL REPORT**

UTILISING DEEP LEARNING IN SIGN LANGUAGE RECOGNITION & INTERPRETATION

**AUTHORS:**

CHAN JOSHUA JUAN YIN

BRANDON LIN ZHAN HONG

DUONG NGOC YEN

PHAM VAN LONG PHUOC

YU XINHUI

WANG YUJING

**DATE OF SUBMISSION:** 9th MAY 2021

**Table of Contents**

[1. Preface 3](#_Toc71495197)

[2. Product Showcase 4](#_Toc71495198)

[2.1. Features 4](#_Toc71495199)

[2.2. Value Proposition 5](#_Toc71495200)

[2.2.1. Merits 5](#_Toc71495201)

[2.2.2. Limitations 6](#_Toc71495202)

[2.3. Future Improvements 7](#_Toc71495203)

[2.3.1. Improve on Current Model Shortcomings 7](#_Toc71495204)

[2.3.2. Consecutive Gesture Interpretation 8](#_Toc71495205)

[2.3.3. Applicability with Natural Language Processing (NLP) 8](#_Toc71495206)

[3. Approaches & Outcomes 9](#_Toc71495207)

[4. Project Takeaways & Reflections 23](#_Toc71495208)

[4.1. Chan Joshua Juan Yin 23](#_Toc71495209)

[4.2. Brandon Lin Zhan Hong 23](#_Toc71495210)

[4.3. Duong Ngoc Yen 24](#_Toc71495211)

[4.4. Pham Van Long Phuoc 24](#_Toc71495212)

[4.5. Yu Xinhui 25](#_Toc71495213)

[4.6. Wang Yujing 25](#_Toc71495214)

[ANNEX A: Individual Contributions 26](#_Toc71495215)

[ANNEX B: References 27](#_Toc71495216)

[ANNEX C: Technical Aspects 28](#_Toc71495217)

[C.1. Comparison of GRU Model Accuracy using different hyperparameter values 28](#_Toc71495218)

1. Preface

The ability to hear is often taken for granted. However, auditory perception is not universal trait. The World Health Organisation (WHO) estimates that more than 5% of the world’s population – approximately 466 million people – have some form of disabling hearing loss, and this number is estimated to grow to in excess of 900 million people by 2050 [1].

In lieu of verbal communication, members of Deaf cultures use sign languages as their primary means of communication within their communities. Meaning is conveyed through a combination of hand shapes, hand orientation, and the movement of one’s hands, arms and/or body, as well as one’s facial expressions. Due to the distinct difference in communication mediums between hearing people and members of Deaf cultures, effective interaction is not straightforward. As human society consists primarily of hearing people, it is generally difficult for a hearing person and a deaf person to converse without the aid of a sign-language interpreter or technological accessibility, such as through texting. Members of the Deaf community may thus feel misunderstood or be discriminated against by those who do not understand sign language [2].

There have been attempts made in the past few decades to overcome this unique communication barrier. The introduction of cochlear implants has helped many users gain auditory perception to a certain degree, but their use is not widespread and has faced resistance from members of Deaf communities, requiring time to become more widely accepted and utilised. On the other hand, with regards to the use of text messages as an alternative means of communication, this method merely forces communication through a secondary medium, at the expense of the Deaf person’s convenience.

2. Product Showcase

In order to overcome the communication barrier, our Team believed that a better approach to ameliorating this problem is to enable hearing people to better comprehend sign language. We seek to do so via a PC-based application which, via the use of deep learning models capable of interpreting American Sign Language (henceforth referred to as ‘ASL’) in near real-time, will be able to interpret a video input of communication in ASL into American English (henceforth referred to as ‘English’).

2.1. Features

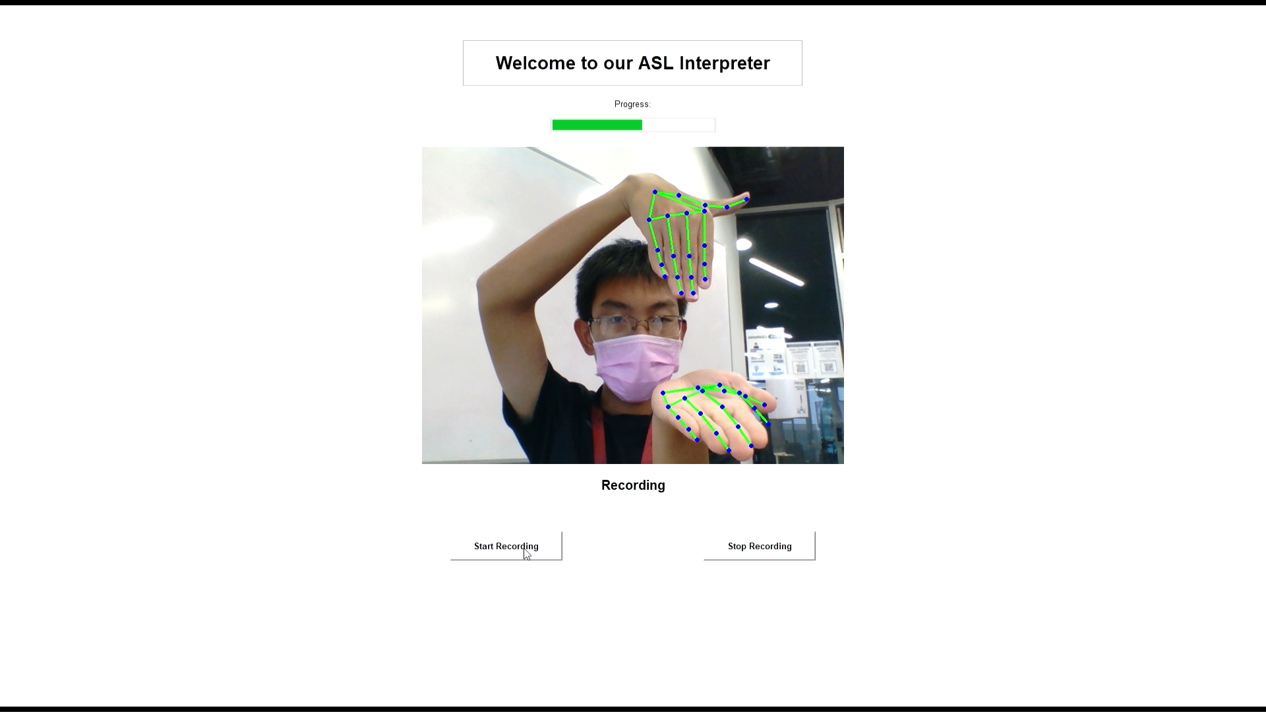


Fig. 1: Screenshot from Application, showcasing handtracking of webcam video input

When our PC-based application is opened, the user will be greeted by an aesthetically-pleasing interface, with options to ‘Start’ and ‘Stop’ recording. Upon clicking on the ‘Start’ button, the application will give the user some time to prepare, before recording a 150-frame video, submitted through the user’s webcam. The video input from the camera is simultaneously displayed to the user, with MediaPipe hand landmark feature extractions allowing the user to observe how their own recording is being understood by the application.

Upon completion of the recording, either through completion of recording of the 150-frame video or a manual stoppage, our application will run our completed deep learning model for interpretation of ASL video input into an English word corresponding to the ASL hand gesture predicted. In near-real time, our application will output the top three predicted words back to the user.

2.2. Value Proposition

2.2.1. Merits

Our product’s main merits are in greater interpretation capabilities over contemporary models, good accuracy in predicting ASL gestures, low cost of use to the user, and good ease of use.

Greater Interpretation Capability

In our research, we took note of two existing types of sign language interpretation models available on the Internet.

* The first type was the Alphabet-Based Model [3], which were only capable of interpreting the signed version of a particular letter in the 26-letter English alphabet. These models, while capable by themselves, offered limited real-life use as most sign language users used hand gestures to represent entire words, as opposed to spelling out words with singular letters.
* The second type of model available was Word-Based Model, our own model being of the type. However, models made available online offered limited interpreting capability, with a small number of English words available; for example, a model we encountered offered just 10 to 20 English words [4].

Our model has a greatly-expanded interpretation capability of 111 English words.

Good Accuracy

In addition to the expanded interpretation capability over previous models, we are additionally able to achieve a test case prediction accuracy of 52.99%, and a top-three-words prediction accuracy of 71.79%. Taking into consideration the complex nature of identifying hand gestures and the limited data available for training, we firmly believe we created an adequately-accurate model.

Lower in Cost

Another existing product is the wearable glove that uses embedded sensors to recognise signed gestures [5]. Compared to this product, our software-based product minimises the need for expensive hardware, as a user would only require a PC with a webcam attached. As such, the cost of the product is greatly reduced over.

Easy to Use

Our product also enjoys the advantage of ease of use and user-friendliness. The aesthetic is simple and uncomplicated, requiring minimal technical know-how to use and thus does not limit its potential user base.

2.2.2. Limitations

Our product is limited primarily by the limited data freely available for use in training our deep learning model, as well as current technologies.

* While our training dataset from the WLASL Database [6], which features 2,000 common different words in ASL, is of reasonable size, certain words have greater amounts of data available, introducing significant bias which had to be corrected for. In doing so, we reduced our initial word count from more than 2,000 to just 111 ASL words, improving prediction accuracy but with the trade-off of limiting our model’s interpretation capabilities.
* We are limited by the technologies of today. While MediaPipe is one of the best handtracking frameworks available, it can be affected by a multitude of factors, such as video image contrast, image proportions, recording hardware, and limited tracking capabilities with hands in certain orientations. As such, it is unable to be thoroughly accurate in tracking all hand gestures.
* Our test accuracy of 52.99%, while respectable, has much need for improvement before it can be utilised as a full-fledged interpreter. Furthermore, the real-time accuracy of our model will likely be lower than our test accuracy.

2.3. Future Improvements

2.3.1. Improve on Current Model Shortcomings

We can improve on the current shortcomings of our current model, which include a small size of database, overreliance on a single method of data extraction, limited word interpretation capabilities, and inadequate accuracy.

Small Database

We can expand our database to include sign language recorded videos from other data sources. This may require investment from other relevant parties.

Overreliance on a Single Method of Data Extraction

We are limited by the technology of our time. However, in a few years, there will be better performing APIs which can perform the function of hand landmark feature extraction. This may allow us to improve our model training, enabling improved tracking of hand gestures and thus, improved prediction accuracy.

Limited Word Interpretation Capability

We are limited by the small size of dataset available to us. With a larger database, it is possible for us to reduce bias in our model, thereby allowing for expansion of interpretation capabilities to cover more ASL words.

Inadequate Accuracy

We noted that our model’s accuracy is very much reliant on the quality and quantity of database. With the proposed improvement mentioned above, in time, we can improve on our model’s prediction accuracy.

2.3.2. Consecutive Gesture Interpretation

Currently, our model is only capable of interpreting one individual ASL gesture at a time. This model could be useful for individual sign language learning and training purposes, but would be less useful in real-life applications, as most of the real-life scenarios would involve sign language conversations in complete sentences, i.e. consecutive performance of sign language gestures.

Therefore, future improvements can focus on expanding the capability of our application to be able to interpret consecutive hand gestures and form word strings from these gestures.

2.3.3. Applicability with Natural Language Processing (NLP)

Building upon Consecutive Gesture Interpretation, the next step would be to apply Natural Language Processing (NLP) in order to convert word strings into coherent English sentences. The difficult in this lies in the fact that ASL deviates from Standard American English in terms of grammar, and as such, time, effort, and careful consideration must be taken to ensure the successful deployment of NLP.

3. Approaches & Outcomes

Our product development can be separated into 3 Phases:

1. Data Preparation
2. Model Training
3. Application Development

In the process of development, we experimented with different methods to obtain a suitable deep learning model to interpret ASL hand gestures into separate English words. The following, in chronological order, are the various Phases of Development we went through, with Roman Numerals (I, II, III, IV, etc.) used as indication of our nth attempt.

**Data Preparation I**

**Approach:**

We utilised Word-Level American Sign Language (‘WLASL’), which features 2,000 common different words in ASL [6]. We extracted the hand features as NumPy arrays using MediaPipe, a framework developed by Google Research in 2019 [7] used to provide real-time body, facial and hand tracking and extract important landmark features.

**Outcome:**

We were able to obtain word-level hand features in Four-Dimensional (4D) NumPy array for use with two hands. The four dimensions are: *number of frames*, *two hands, 21 hand landmarks per hand per fram*e, and *3 dimensions for each landmark (x, y, depth)*. Hence, the shape of the array is [n, 2, 21, 3] where n is the number of frames of a video.

**Model Training I**

**Approach:**

We tested three different models, *Multilayer Perceptron (MLP) Classifier*, *Long Short-Term Memory (LSTM)* and *Gated Recurrent Unit (GRU)*, using a very small subset of **Data Preparation I**.

* **MLP Classifier:** A type of feedforward artificial neural network (ANN). MLPs are useful in research for their ability to solve problems stochastically, allowing for approximate solutions for extremely complex problems such as fitness approximation.
* **LSTM:** A type of recurrent neural network (RNN) model, developed to deal with the vanishing gradient problem. They are well-suited to classifying, processing, and making predictions based on time series data.
* **GRU:** Another type of RNN model which targets the problem of vanishing gradient during backpropagation. Compared to LSTM, it is newer and uses fewer parameters in order to increase the speed of training.

**Outcome:**

Based on our small subset test, predictions using the MLP model were very unstable, and as such, we dropped its use in further modelling.

* Comparison of Different RNN Models (LSTM vs. GRU vs. Simple RNN)

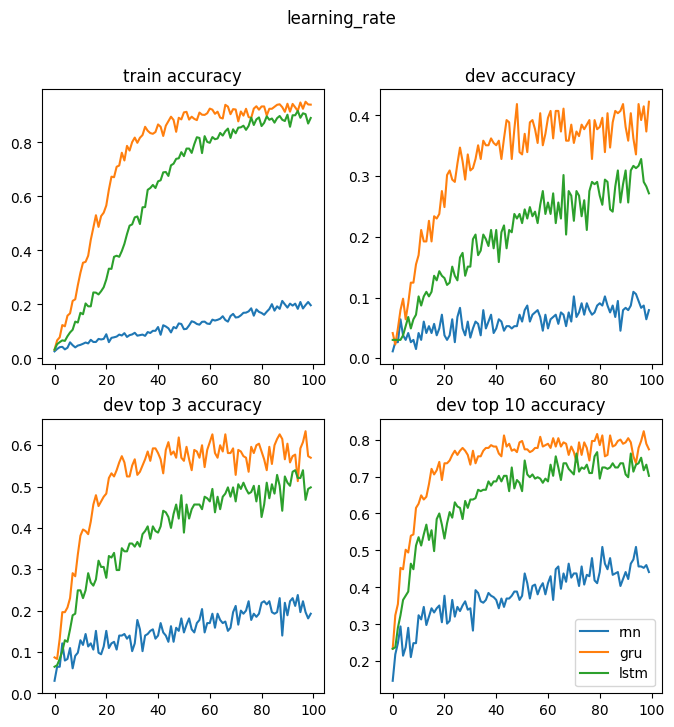
****

Fig. 2: Graph of Prediction Accuracies against Epoch [LSTM vs. GRU vs. Simple RNN]

The above graph shows the change of accuracy with respect to epochs, utilising the three different models. GRU proved to be the best model amongst the three with fastest training speed and best performance. Combining our online research and this trial result, we decided that GRU would produce the most suitable model.

* Problems Faced

Our group discovered that there were NaN (Not a Number) values in the dataset. These arrays containing NaN values, corresponding to frames in which the hands did not appear, result in computations with NaN outputs, confusing the model and thus reducing model accuracy.

Another problem faced was that the video files we utilised had varying numbers of frames, resulting in non-homogeneity in dimensions, resulting in errors in NumPy manipulations.

**Data Preparation II**

**Approach:**

In order to resolve the issue caused by the NaN values, we replaced all NaN values with zeroes. Our initial thoughts were that because of the characteristics of GRU, a frame consisting zeroes, as opposed to NaN values, would not have a significant impact on the model prediction.

**Model Training II**

**Approach:**

Utilising Data from **Data Preparation II**, we re-trained the GRU model after padding the arrays with zeroes so that each array representing a video file contains 233 frames, homogenising dimensions.

**Outcome:**

The model achieved an accuracy around 3%. We believe this is due to a large number of insignificant frames (padded by zeroes), as well as the inherent bias in the dataset from certain words having few corresponding videos while others having multiple. This resulted in more unreliable predictions for less-represented words.

**Data Preparation III**

**Approach:**

Based on the model from **Model Training II**, each array consisted of 233 frames, with zeroes present. We decided to exclude frames with zeroes to eliminate their impact.

We noted that processing all 233 frames in each video file would result in significant repetition, slowing down the model training process. Hence, we sampled randomly 16 non-zero frames for each video, with repetitions allowed for assured randomness. We introduced randomness in our (non-zero) frame selection based on the rationale that all non-zero frames will have equal chance of being selected, thereby eliminating bias in frame selection during our data preparation stage. This approach is asserted in this article by De Coster et. al. (2020) [8].

Repetition is introduced here as we noticed that a significant number of videos have less than 16 non-zero frames: it would not be otherwise possible to homogenize the number of frames without allowing for repetition. In cases where videos have more than 16 non-zero frames, we also allowed for repetition of frames to apply a consistent policy on frame selection across all videos.

Last but not least, we noted that many American English word labels had less than 16 videos associated with specific words in the original WLASL database. For instance, the word “able” has only 7 videos in the original database. Since bias arises from differences in the number of training videos, to mitigate data bias, we dropped words which had less than 16 corresponding videos such that the model could predict more accurately.

**Outcome:**

The refined dataset contained hand features for 111 words, from the original 1971 words. Each of these words have more than or equal to 16 corresponding videos. Each video contained 16 frames, and was represented by a 4D array in the shape [16, 2, 21, 3].

**Model Training III**

**Approach:**

Using the dataset from **Data Preparation III**, we trained the GRU model again and predicted the words with Top 3 Possibility and Top 10 Possibility. We then calculated the accuracy of our model’s predictions.

**Outcome:**

The training (train) set accuracy was 97.46%.

In the development (dev) set prediction, the accuracy was 16.23%.

There was a 34.34% chance the correct word was among the top 3 probable words, and a 61.89% chance that the top 10 probable words included the correct word.

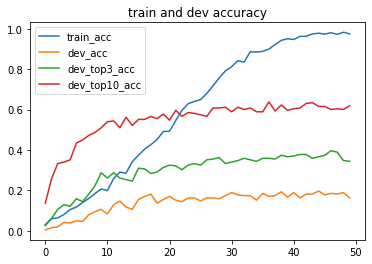


Fig. 3: Graph of Prediction Accuracies against Epoch [MT III]

**Model Training IV**

**Approach:**

As we trained the GRU model using the dataset from **Data Preparation III**, the same 16 frames for each video were used, which meant that some frames of a video were never utilized to train the model. Thus, we decided that random sampling of the 16 frames should occur at the beginning of each epoch. This is to ensure the model can learn from different sets of 16 frames for each video.

**Outcome:**

Using this approach, we managed to achieve a train set accuracy of 91.33%, a dev set accuracy of 29.06%, 48.30% for top 3 probable words and 73.96% for top 10 probable words.

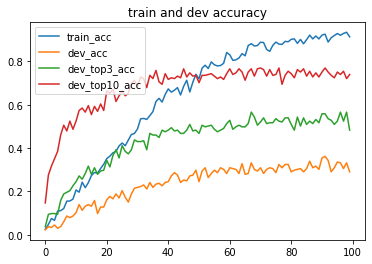


Fig. 4: Graph of Prediction Accuracies against Epoch [MT IV]

**Data Preparation IV (Data Augmentation)**

**Approach:**

The issue of overfitting was observed as the epoch increased. To resolve this, we tried regularisation, dropout, and data augmentation by adding small random shifts to x and y values of hand landmarks.

Additionally, we noticed that the previous method for train/test split could be improved. We performed random train/test split on a word-by-word basis to better balance the dataset. This guaranteed that the model would encounter all the words in the training process.

**Model Training V**

**Outcome:**

Regularisation and dropout did not improve prediction accuracy significantly. As such, we decided not to use them in future model training. On the other hand, word-level random train/test split and data augmentation improved accuracy to 54.7% on dev set. The accuracy for top 3 and top 10 probable words were 77.78% and 89.74% respectively, a significant improvement from the last performance.

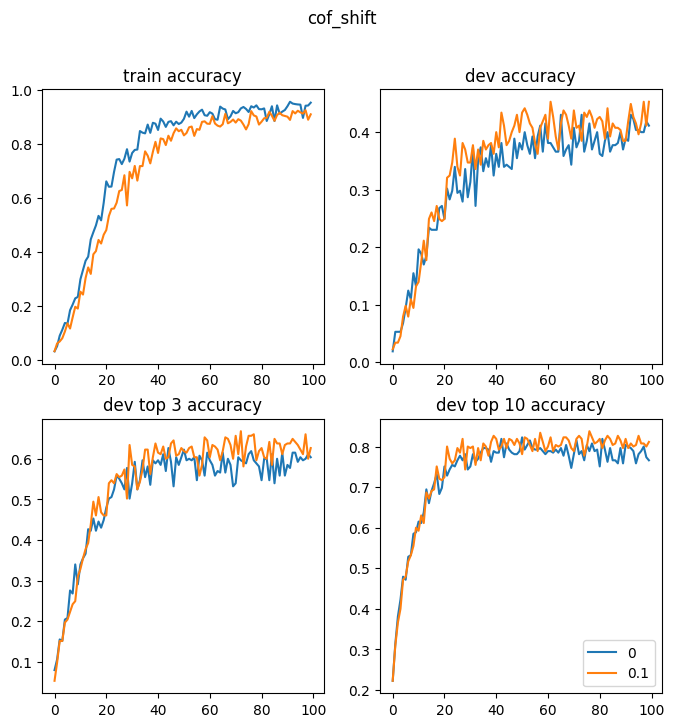


Fig. 5: Graph of Prediction Accuracies against Epoch [MT V, with no shift, and within range ±0.1 shift in x, y]

**Data Preparation V**

**Approach:**

Another attempt at data augmentation was made through constructing mirror images for the frames. This was done to examine if the model understands the mirror image of a hand gesture. A mirrored image of a hand gesture does not change its meaning.

**Model Training VI**

**Outcome:**

Utilising the same model, we obtained a train set accuracy of 91.06%, dev set accuracy of 45.28%, top 3 accuracy of 62.64%, and top 10 accuracy of 81.13%. The resultant accuracy obtained did not improve from the previous approach. As such, we did not use it in the final model.

**Final Model Parameters**

**Data Preparation:**

* NaN to zero
* Exclude zero frames
* Word level random sampling
* Data augmentation by adding small random shift to x, y values of each hand landmark
* Number of words: 111

**Type of model used:**

GRU

**Hyperparameters (Refer to ANNEX C):**

* Seq\_len: 8 frames per video
* Hidden\_size: 512
* Number of layers: 2
* Batch\_size: 32
* learning \_rate: 0.001
* Bi-directional: False

**Model Performance:**

* Train accuracy: 91.43%
* Test accuracy: 52.99%
* Top 3-word accuracy: 71.79%
* Top 10-word accuracy: 85.47%

**Application Development I**

**Approach:**

As we require the processing power of a PC to process webcam input data, we decided on a rudimentary application created with Tkinter, a standard Python interface to the Tk GUI toolkit. Utilising Tkinter, we were able to customise our proposed application layout.

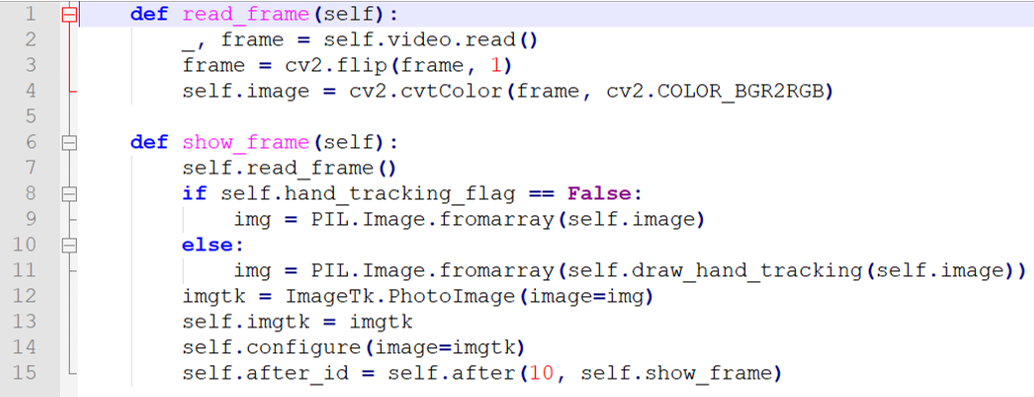


Fig. 6: Snippet of Code from Application Programming

**Outcome:**

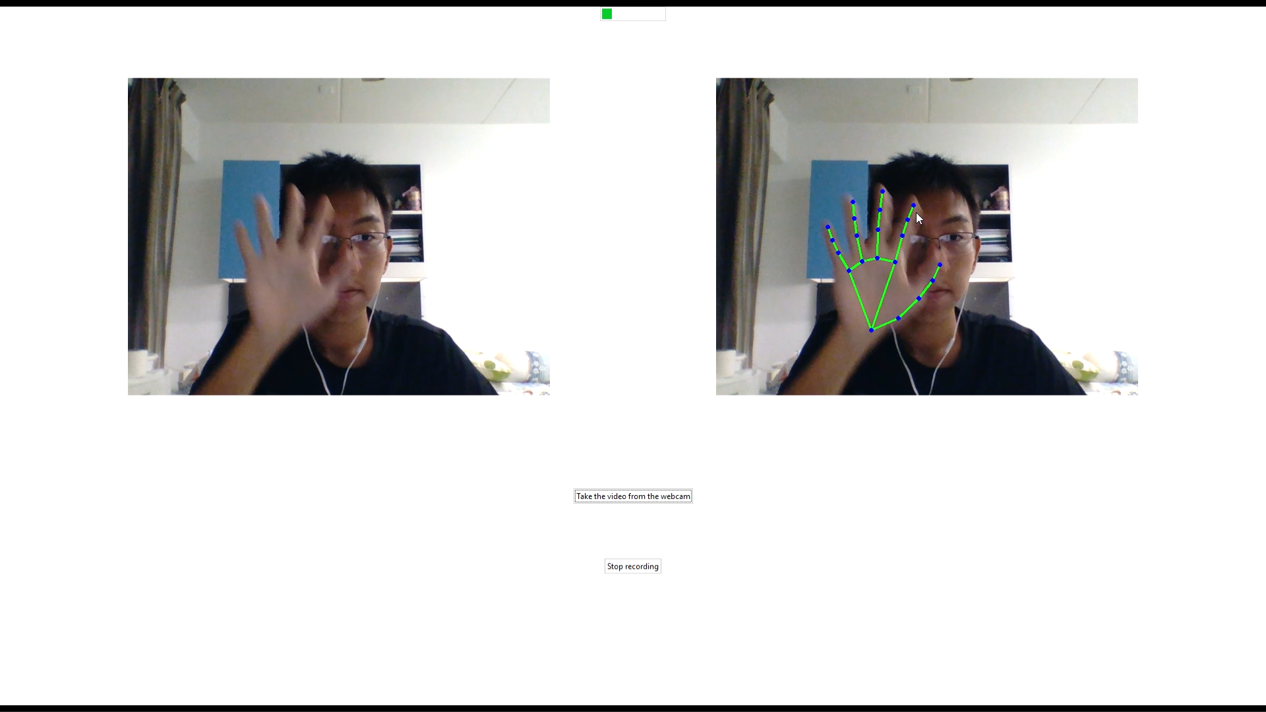


Fig. 7: Screenshot from First Iteration of the Application

Fig. 7 showcases the First Iteration of the Application. Two videos are shown, one for raw video and the other for showcasing handtracking. The two buttons in the Figure, ‘Take the video from the webcam’ and ‘Stop recording’ will start and stop recording webcam input, respectively. The user is prompted to prepare for recording with a 3-second timer upon clicking the ‘Take the video from the webcam’ button.

On experimenting with frame times, we decided that recording for 150 frames would provide adequate time hand gestures to be performed and registered by the application. At this point, we had not tested how to deploy the model into the app or how to save the hand landmark arrays yet.

**Application Development II**

**Approach:**

On consideration that it may not be easy for users to view both video outputs at once, we combined the two outputs into a single output which will first display the raw video in the first 3 seconds, and show hand-tracked frames for the remainder of time allocated.

The application will, additionally, show when webcam input is being recorded. It will also run our trained deep learning model and output on screen the predicted word.

**Outcome:**

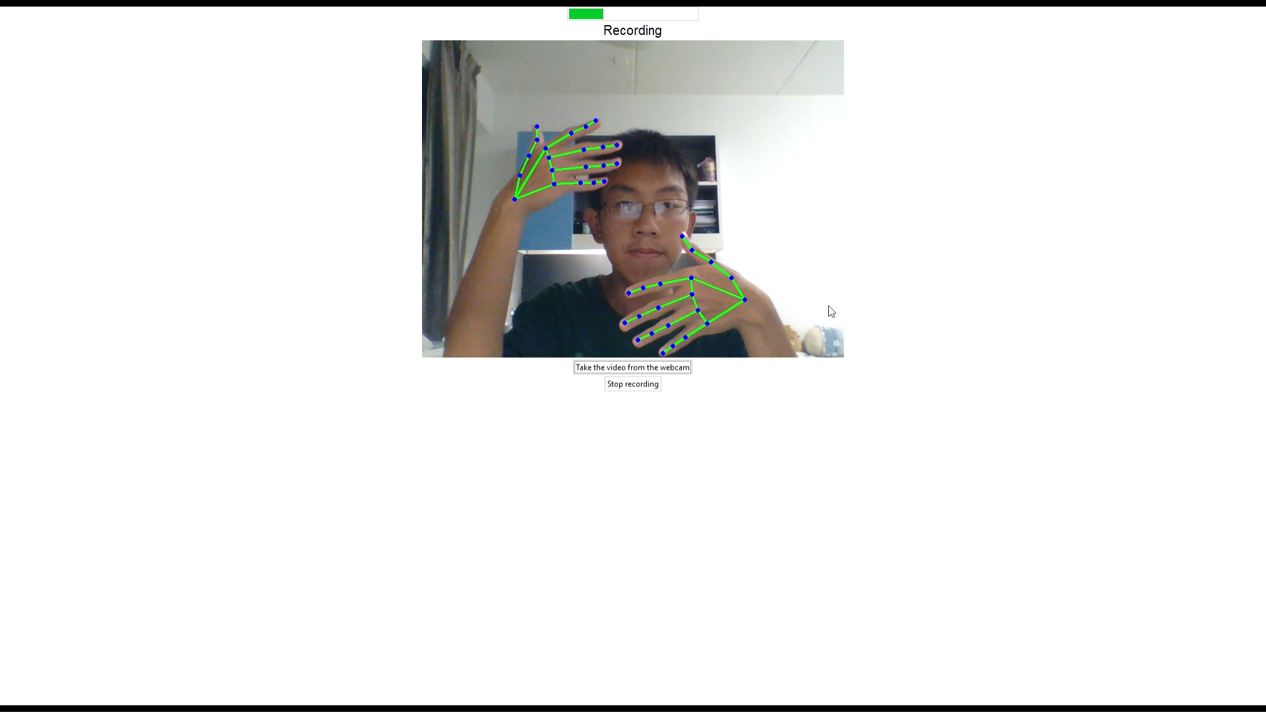


Fig. 8: Screenshot from Second Iteration of the Application

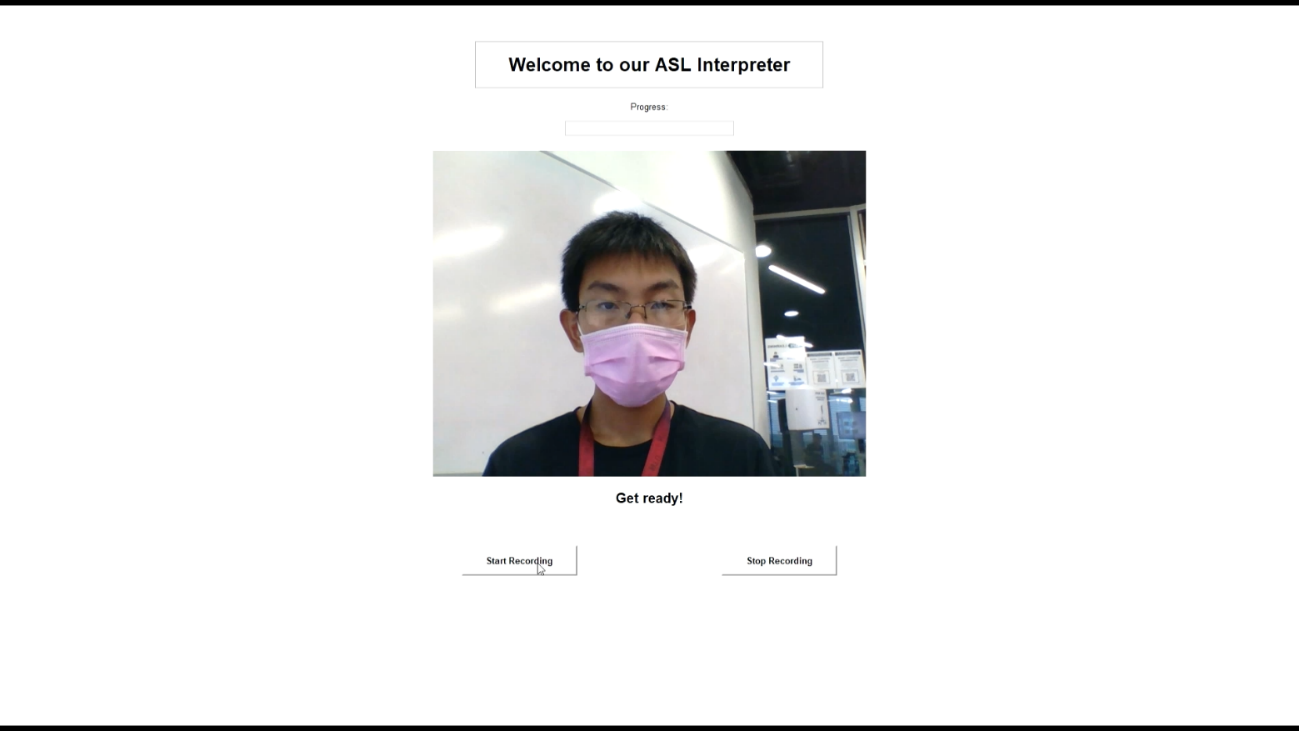
The application worked as intended, recording input via the webcam, and outputting a predicted word. However, we noted that the predicted word was not always accurate, and thus decided to, for the next iteration, replace the predicted word with the top 3 words predicted by our model in terms of descending (starting from highest) probability.

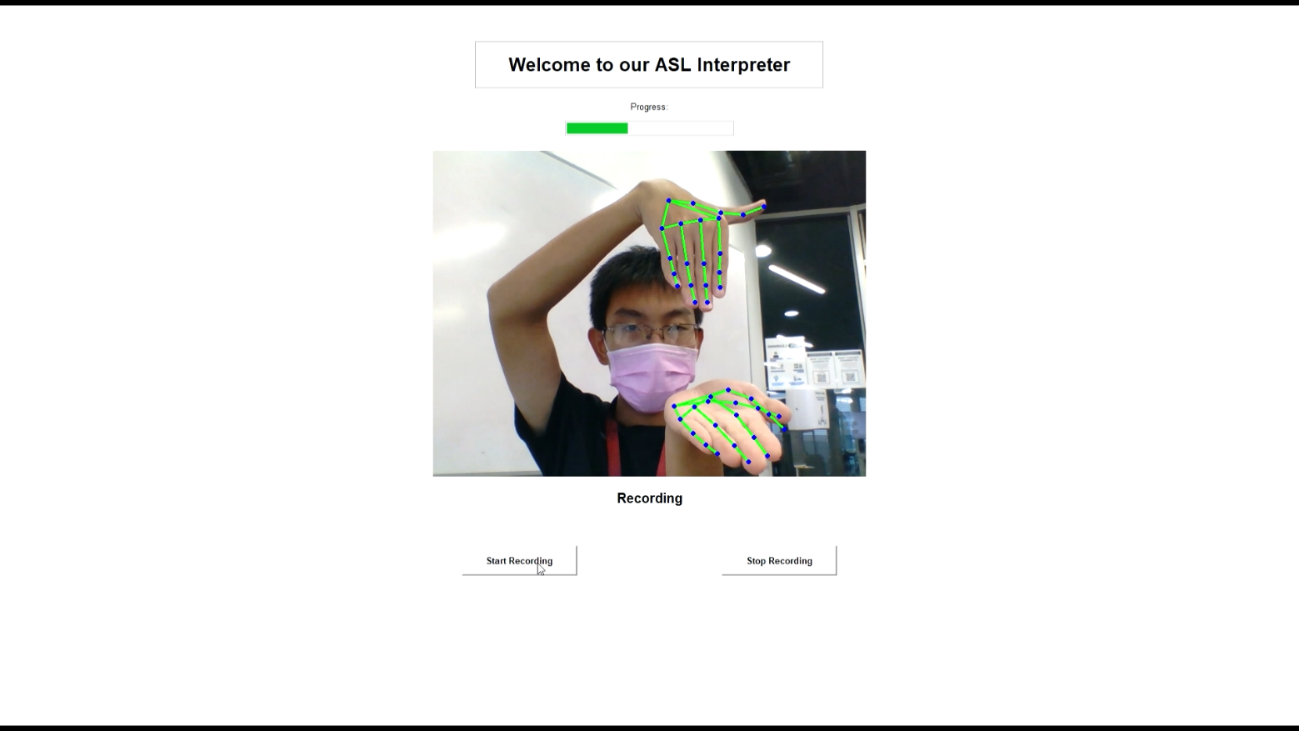
**Application Development III**

**Approach:**

The general aesthetic of the GUI was improved upon. Furthermore, the top three words with the highest probability were shown, instead of merely the word with highest probability.

**Outcome:**





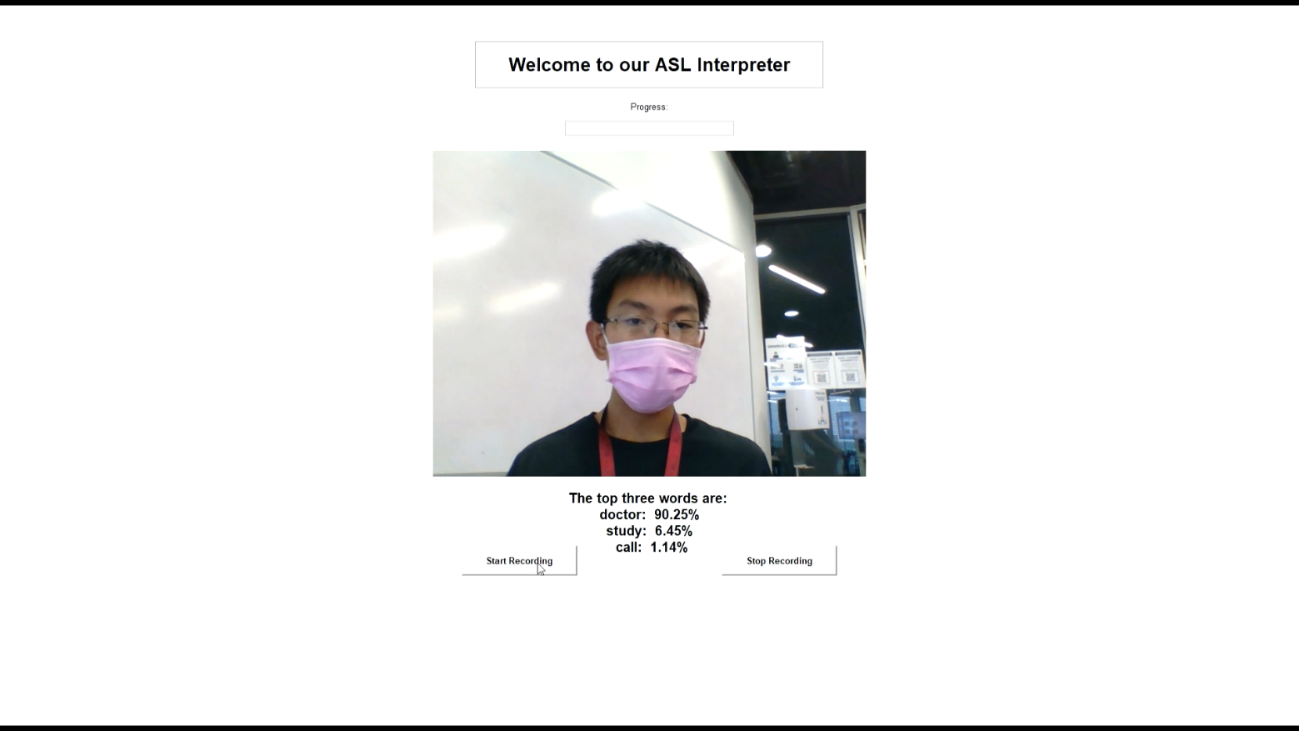


Fig. 9: Screenshots from Third (Final) Iteration of the Application

**Outcome:**

The application worked as intended, recording input via the webcam, and outputting the three top predictions. We decided that this iteration was suitable for our needs. The code for the application can be found at the following link: <https://github.com/phuocchubeo123/Dep_Project>

4. Project Takeaways & Reflections

4.1. Chan Joshua Juan Yin

For me, the greatest reward from this project is the experience of developing a product: from the earlier stages of data collection and pre-processing to the later stages of RNN training model at the backend and desktop application development at the frontend. This experience will be applicable to other software-related projects in the future.

I also gained exposures to the coding techniques and model implementation from my peers. Data quality and quantity determines the accuracy of our training model: this phenomenon is observable throughout our project, and I am glad that I could contribute to parts of the stages of data processing. This project also prompted me to conduct research on several variations of RNN models and their associated theories. It also gave me the opportunity to observe how hyperparameter adjustments are conducted in actual model implementation. I also observed how improvements in user interfaces can make decisive impact to the user experience overall: in this project, it is the impact to the audience of our final group presentation.

I am glad to have partnered with Yen on a lot of aspects in this project, especially on the initial research and coding work. Also, gratitude to Phuoc, Yu Jing and Brandon for their contribution to the dirty work of coding (I learnt a lot just by reading through your code), and Xinhui for her explanation on model selection.

Overall, I am very satisfied with our team dynamics, which gave equal and adequate opportunities for every one of us to demonstrate our areas of strength.

4.2. Brandon Lin Zhan Hong

Over the duration of this Course, I have gained a great many things. Firstly, I actively participated in and contributed to a meaningful project with tangible benefits to the Deaf Community; should a greater amount of data and resources be available in the future, interested parties would be able to build upon our work and create a high-value product to improve the ease of communication between members of Deaf Communities and the hearing. Secondly, I learnt much about Machine Learning as a whole; with the skills I have picked up during this Course, I am now better prepared to work on Machine Learning & Artificial Intelligence projects back in the armed services. I also gained an appreciation for the amount of hard work and effort that is required to achieve a good end result, and that Machine Learning is not a one-size-fits-all panacea which can solve all our problems; we need to continually innovate and improve upon our past achievements in order to maximise its benefits. And finally, last but not least, I made good friends who will continue to be my peers as we expand our understanding in the field. To end off, I would like to sincerely thank the Profs and Project Mentors who have helped us to achieve our goals.

4.3. Duong Ngoc Yen

Learning new Deep Learning knowledge: Besides learning required knowledge on the internet, I have also learned so much new knowledge from Phuoc, Yujing and Xinhui’s code, the way they code, the way they tune parameters and the way they evaluate the code (drawing the graph, consider the top accuracy, etc). For me, that is so professional. I also gained some intuition on training the data. I found that the dataset plays an important role on the accuracy of the model. Most of the modification on the dataset in our project improve the performance of the model the most. I also gained some knowledge in data preparation, how to make the data more structural for the model to learn.

The ability to read research paper: Joshua was the person with this style, and he told me to read research papers to improve knowledge. After this project, I have a new skills of reading research papers and can find the targeted information that I need.

This project also gave me an opportunity to meet new friends that I can share and help each other in and after the project.

4.4. Pham Van Long Phuoc

Through this course, I made a new team of friends, and all of them are extremely good at their job and fun to interact with. My teammates are nice and responsible; they helped me grow, and I learned a lot from them.

The second thing I learned from this course is the mindset of “doing” things. I delayed coding the GUI till later in the project. Watching the GUI tutorial, I did not think that I could do it. However, through exploring each component of the app, I gained confidence in myself and found that the coding flow, while very challenging, was also very fun. During the coding sessions, I encountered hundreds of bugs, and the many ideas I had took me days to figure out and implement, but through experimenting a lot, I managed to get them done. I am grateful for the abundance of resources available to us. If I ever need to learn about anything, for example – ‘how to create a button for the app’, I just need to go to StackOverflow.

The biggest breakthrough for me was when I stopped only reading the code; I put it into practice, running the code locally on my laptop and changing things on-the-fly to see how my code is put into action. The process of doing, as opposed to merely reading, was far more effective.

In conclusion, this course really helped me: this was my first coding project, after all, and now I am no longer scared of debugging.

4.5. Yu Xinhui

During the project, I deepened my understanding of deep neural network, in particular, recurrent neural network (RNN) and its more advanced versions: long short-term memory neural network (LSTM) and gated recurrent unit (GRU). I learnt about their structures, capabilities, and limitations.

Data preparation was the most important part of our project and affected the model prediction accuracy significantly. I learnt how to extract, clean, filter, save and load data, as well as some more advance operations on dataset to resolve the problem of overfitting. Namely, regularization and data augmentation. Also, I am now more aware of the effect of hyperparameters on the performance of the model. I now know how to combine theoretical and experimental methods to find the most appropriate hyperparameter values catering for the specific dataset.

Last but not least, the teamwork experience has given me a better idea on how to foster collaboration and mutual growth in a team where everyone has different skillsets. I find this equally valuable, if not more, than the technical knowledge obtained.

4.6. Wang Yujing

I am happy to be part of this team to identify the need from the deaf community and develop a sign-language interpretation model. From Oct 2020 to April 2021, I have learnt a lot about machine learning, including dataset preparation, features extraction, simple machine learning models such as linear regression, SVM, KNN, as well as deep learning models such as CNN, RNN. I also have learnt something about Natural Language Processing, but it is a pity that we decided not to use this in our project due to time constraint. During the model building and training, I have practiced my coding and debugging skills, and have a more solid understanding of the GRU model, about how the model works, and how the hyperparameters would affect the model complexity and training process. I am proud to be part of this team as we made full use of the advantages of each one of us. I mainly did the coding work and teammates, especially Xinhui, have helped me a lot in converting my ideas into proper written English. Area of improvement might be that we could have met more frequently to discuss our own progress so that we could understand the work done by each other better.

ANNEX A: Individual Contributions

|  |  |
| --- | --- |
| CHAN JOSHUA JUAN YIN | * Supplementary programmer * Contributed to parts of the stages of data processing * Team proposal research + write-ups on classification processes * Final report drafting on value propositions and improvements |
| BRANDON LIN ZHAN HONG | * Overall Team Scribe & Editor * Graphic User Interface (GUI) Designer * Supplementary Programmer * Final Report & Presentation Writer |
| DUONG NGOC YEN | * Supplementary programmer * Contributed on drafting the outline of classification process * Contributed ideas to data preparation and model selection * Drafted the Merits & Outcomes section in final presentation |
| PHAM VAN LONG PHUOC | * Extracted the hand landmarks from the videos * Contributed ideas to data preparation * Written the code for the GUI * Provided demo videos for the product showcase |
| YU XINHUI | * Contributed to the NLP idea for Future Improvement section in the presentation * Contributed ideas to data preparation and model selection * Assisted in coding for data augmentation * Drafted the Approaches & Outcomes section for the final report and presentation |
| WANG YUJING | * Main Programmer for model building & training, and data augmentation * Plotted all accuracy graphs for comparison * Drafted the Approaches & Outcomes section for the final report and presentation * Data preparation (reduce class size to 111 words, random frame sampling) |

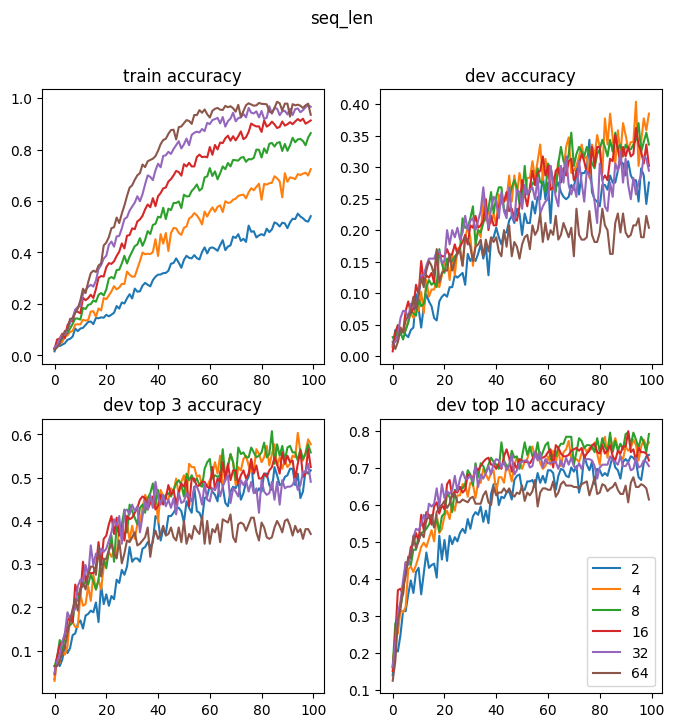
ANNEX B: References

|  |  |
| --- | --- |
| [1] | World Health Organisation, “Deafness and hearing loss,” 1 March 2020. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss. [Accessed 6 January 2021]. |
| [2] | J. E. Nash and A. Nash, Deafness in society, Massachusetts: Lexington Books, 1981. |
| [3] | National Institute of Standards and Technology (NIST), “Sign Language MNIST | Kaggle,” National Institute of Standards and Technology (NIST), 20 October 2017. [Online]. Available: https://www.kaggle.com/datamunge/sign-language-mnist. [Accessed 8 May 2021]. |
| [4] | T. Sang, “Vietnam Sign Language Recognition using Hand MediaPipe framework and LSTM model,” Ho Chi Minh University of Technology, 9 March 2021. [Online]. Available: https://github.com/thanhsang298/Viet-Nam-Sign-Language-Recognition-using-Hand-MediaPipe-framework-and-LSTM-model. [Accessed 8 April 2021]. |
| [5] | S. A. Mehdi and Y. N. Khan, “Sign language recognition using sensor gloves,” *Proceedings of the 9th International Conference on Neural Information Processing,* no. ICONIP'02, 2002. |
| [6] | D. Li and H. Li, “Welcome to WLASL Homepage | WLASL,” GitHub, Inc., 20 Jan 2020. [Online]. Available: https://dxli94.github.io/WLASL/. [Accessed 9 January 2021]. |
| [7] | Google LLC, “Hands - mediapipe,” Google LLC, 2020. [Online]. Available: https://google.github.io/mediapipe/solutions/hands. [Accessed 9 January 2021]. |
| [8] | M. De Coster, M. Van Herreweghe and J. Dambre, “Sign Language Recognition with Transformer Networks,” *European Language Resources Association (ELRA), Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020),* p. 6018 – 6024, 2020. |

ANNEX C: Technical Aspects

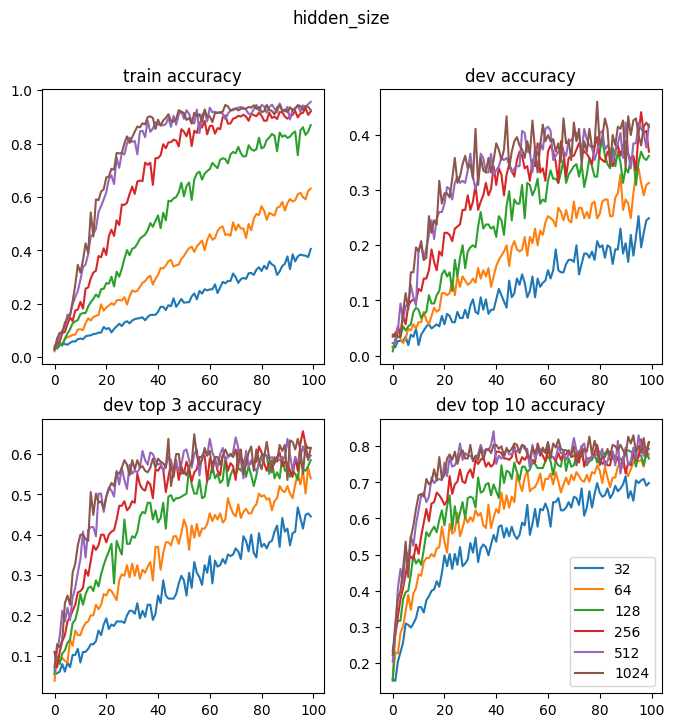
C.1. Comparison of GRU Model Accuracy using different hyperparameter values

**Seq\_len:**



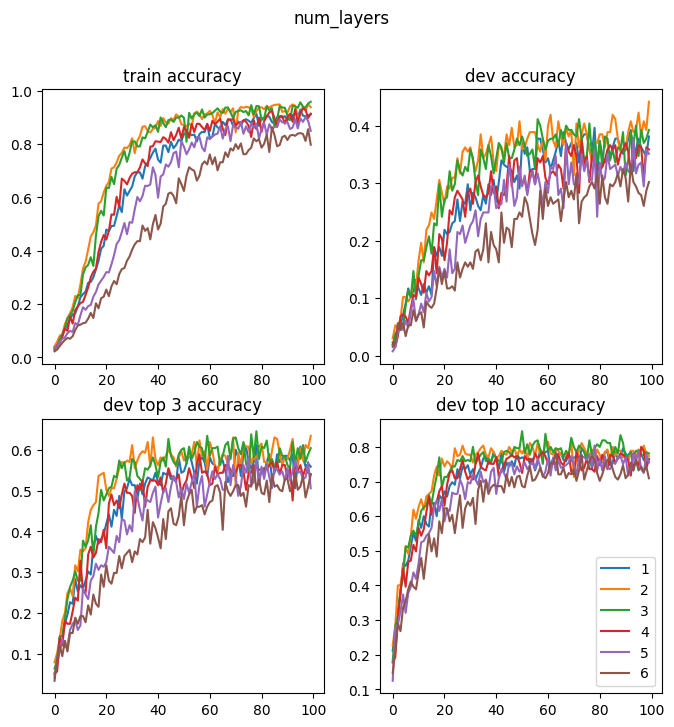
The above graph is the change of accuracy with respect to epochs, when the model is trained using different numbers of frames for each video. From the graph, we could see that 8 frames per video allowed for highest accuracy. There was a trade-off between the number of frames and the amount of information represented. Too few frames would not provide enough information for the model to learn. However, with too many frames, the GRU model would be processing a very long sequence and might forget information encountered earlier, reducing its prediction accuracy. As each video was between 2 to 3 seconds, 8 frames per video resulted in lowest repetition while still retaining sufficient information. We thus decided to use 8 frames per video for training of the GRU model.

**Hidden\_Size:**



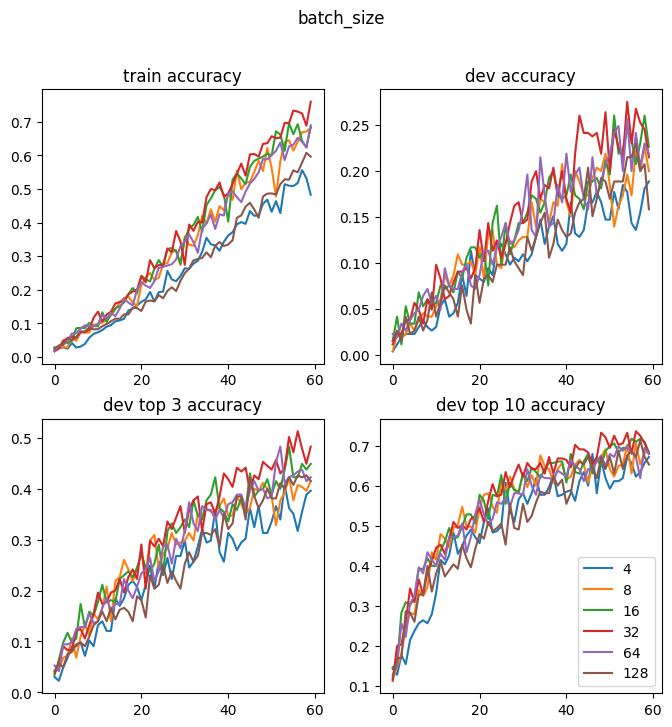
Since the model was to make predictions among 111 words, a relatively larger hidden size was preferred. According to the graph, we decided to use hidden size of 512.

**Num\_layers:**



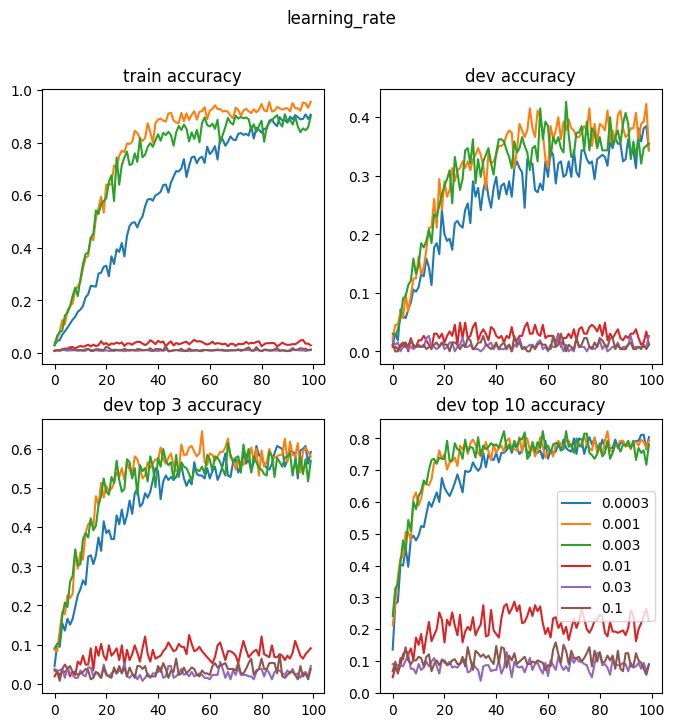
According to the graph, we chose num\_layers = 2 for better performance. A GRU model that had too many layers would be too complex and would have a higher chance of overfitting on the train dataset.

**Batch\_size:**



We chose to use 32 as batch size. In order to fully utilize the power of GPU to accelerate the learning process, we should not use a very small batch size.

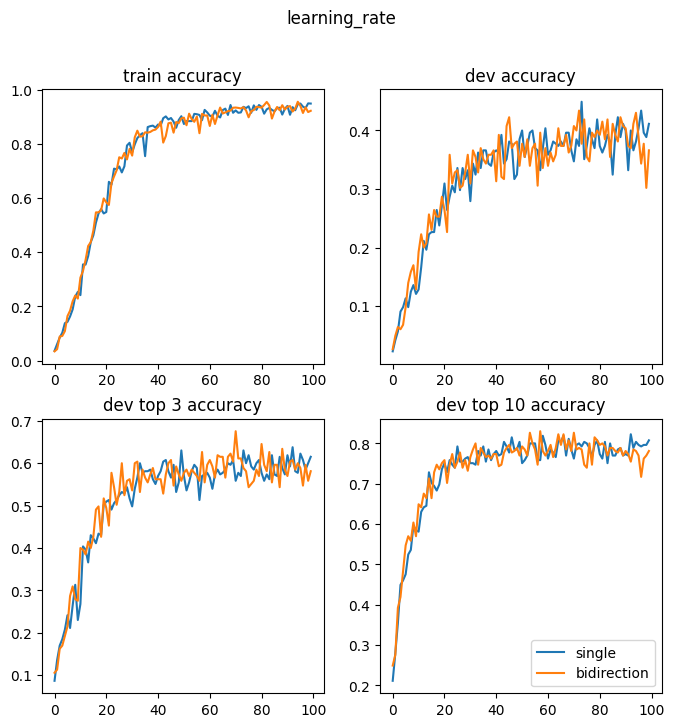
**Learning\_rate:**



If the learning rate is too small, the loss would not improve much over many epochs and would get stuck in the local minima. If the learning rate is too large, the loss might overstep the global minimum.

Hence, the appropriate learning rate must be selected using the graph above and we eventually used a learning rate of 0.001.

**Bi-directional:**



Bi-model did not improve the accuracy because the data was a time series and should not be interpreted in the reverse order. Thus, we decided to use a single direction model.