AirUI Final Project Report

Aman Bhargava (1005189733) Adam Carnaffan (1005069435)
Alice Zhou (1004838697)

December 6th, 2020

Total Word Count: 1819 words

Background

Introduction

AirUI stands for ‘Artificially Intelligent Reality User Interface’. We aim to create a classifier for gestures (e.g. swipes, taps, etc.) made on wooden surfaces based on the audio users produce. This classifier can be used in a user interface system to make human-computer interaction more accessible, both economically and for those with physical disabilities.

We choose to use a Neural Network approach because the state-of-the-art in the field [1] yields a sub-optimal classification accuracy of 89.5% using a shallow decision tree. It is a difficult problem to hard-code due to noise and intra-class variance, but humans are able to differentiate the sounds with ease, implying that the information necessary for classification exists in the data. Thus, a neural network approach was chosen to solve the problem.

Illustrations

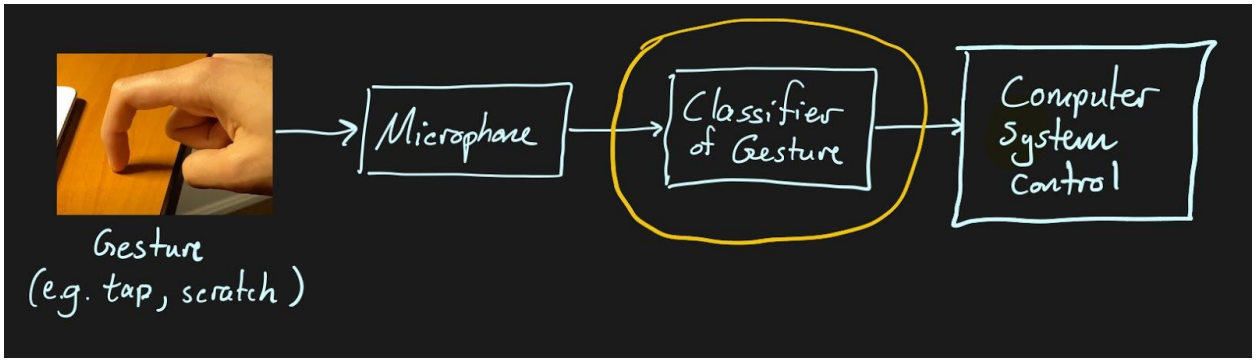


Figure 1: Block diagram for a scratch based user interface. Our classifier, circled in yellow, is the focus of our investigation.

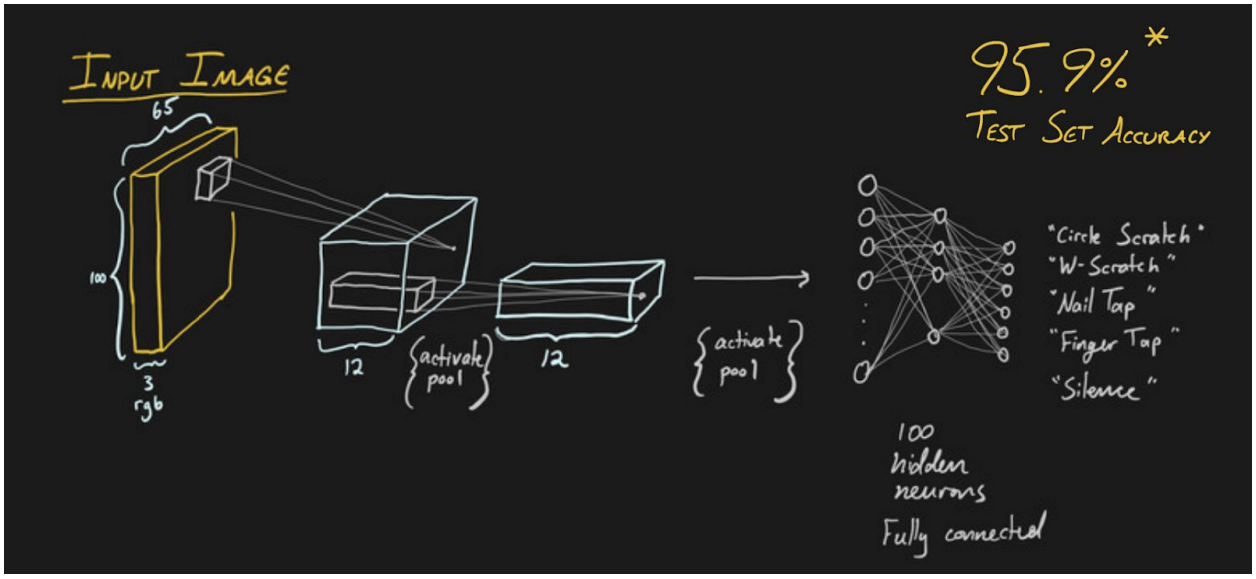


Figure 2: CNN model diagram. An image spectrogram representing the audio is used as input, and the output classes are estimated with a test accuracy of 95.9%.

Related Works

The state-of-the-art in scratch user interface technology is established in [1]. Using a custom-built contact microphone apparatus, they achieve 89.5% classification accuracy on 6 gesture types via a shallow decision tree algorithm with energy peak counting for feature extraction. This paper offered a benchmark for us to improve upon.

A key paper on the use of deep learning models for audio classification comes from Google's research division [2]. Using a CNN with Mel spectrograms for feature extraction, they classify a dataset of 70 million YouTube video audio samples into video categories. They find that CNN tends to perform best for the audio classification task when used with Mel spectrograms, which informed our model and feature extraction method selection.

Data & Data Processing

There were 6 gesture types we aimed to classify:

1. Fingertip Tap.
2. Fingernail Tap.
3. Vertical Scratch.
4. Circle Scratch.
5. W-Scratch.
6. Silence/Null class (*background activity and noise was encouraged*).

To collect our data, we sent a video with instructions on how to record each gesture to a metronome. The metronome made it possible to automate segmentation after the data was cleaned. Our 24 participants recorded 70 samples for each of the 6 input classes. To clean the data, the first 10 samples were discarded as they had substantial variation in timing as the users got used to the metronome tempo.

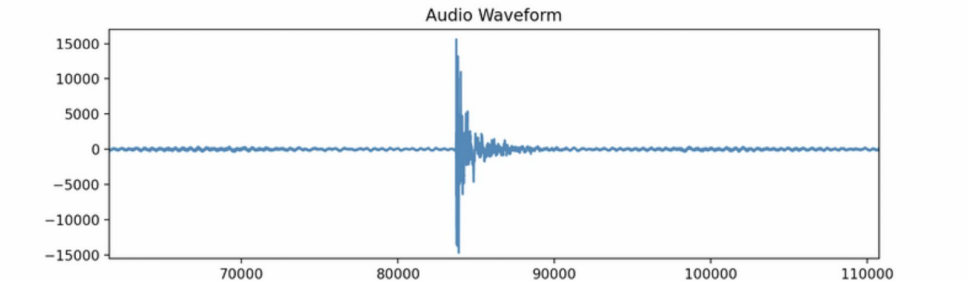


Figure 3: Example raw audio waveform for a low-noise sample of the 'tap' class. Note the dimensionality on the order of 10e5.

After segmentation, we utilized Mel spectrograms for feature extraction using the Librosa Python library. The Mel spectrogram (shown in *Figure 4*) plots frequency vs. energy vs. time. Importantly, the frequency scale is logarithmic. Because frequency responses are generally defined as multiples of some fundamental frequency, the logarithmic frequency scale enables the same kernels to be learned for samples with high and low fundamental frequencies alike.

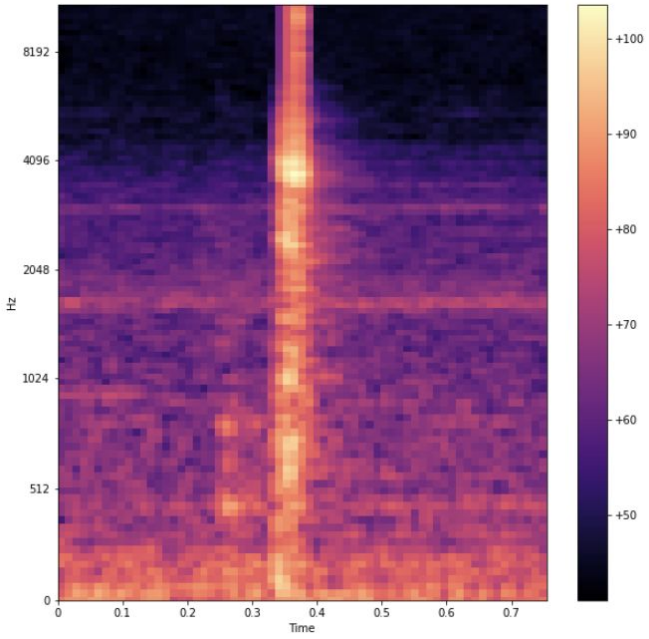


Figure 4: Example Mel spectrogram for a noisier tap gesture. Note the logarithmic frequency scale. Dimensionality is reduced to the order of 10e3.

Our final dataset consisted of 8640 training samples, evenly split between each of the 6 classes. The training set comprised 60% of the data, the validation set 20%, and the test set had the remaining 20%. To mimic the methodology of [1], we allow samples from the same individuals to appear in the three datasets.

Modelling

Architecture

The final model architecture is a Convolutional Neural Network (CNN). It consists of 2 convolutional layers followed by 2 fully connected layers. To obtain a probability-like output, we used 6 output neurons with a softmax function that was bundled with the Cross Entropy Loss function. Other hyperparameters are listed in *Table 1*.

Fixed Hyperparameters	**Variable Hyperparameters (Name + Range)**	
2 Convolutional Layers	Input Image Transformation	Cropping /padding/ squeezing/ stretching
2 Fully Connected Layers	Batch Size	[24, 48, 64]
Batch Normalization	Number of Kernels in the Convolutional Layers	[4, 8, 12]
Cross Entropy Loss function	Learning Rate	[0.1, 0.01]
3x3 Kernel Size	Seed	[6, 42]
100 Neurons in the First Fully Connected Layer		

Table 1: fixed and variable hyperparameters used in the CNN model

Grid search was used on every combination of the variable hyperparameters listed above. Since the final processed data had different sizes, all images were symmetrically cropped to the smallest image size in the dataset (100x65px). This resembles the real life implementation of cropping a long audio to shorter frames of spectrograms. To get a better understanding of how much information is lost due to cropping, other transformations were also used.

To study the model behavior in real life scenarios, we implemented a rolling average algorithm to the model output as audio was inputted in real time to the processing pipeline. The CNN essentially scans across a long input spectrogram with a 100x65 pixel-sized window. The rolling average smooths the probability estimation and allows us to assess the practical performance holistically.

Baseline Model

Baseline 1: Energy vs. Time

The first baseline model sums over **columns** of the mel-frequency spectrograms, which gives us inputs of size 65 vectors. Each vector is a representation of the audio waveform's energy as a function of time. A Multi Layer Perceptron (MLP) model was used to train the dataset. There were 2 fully connected layers in the model with ReLU activations, and 100 neurons were used in the hidden layer. Using the Adam optimizer, the model achieved a test accuracy of 71.2%.

Baseline 2: Energy vs. Frequency

The second baseline model uses the same MLP model as above. However, instead of summing over the columns, the input of the second baseline model sums over the **rows** of the spectrograms. The 100-dimensional feature vectors represent the audio energy as a function of frequency. The model achieved a test accuracy of 81.2%.

The baseline models were chosen to reflect common feature extraction/dimensionality reduction methods for waveform classification. Since the inputs are 1-D arrays with much smaller sizes compared to the spectrograms, MLP was used.

Results

Quantitative Results

Our quantitative results were taken from the cropped dataset. False positives refer to improper gesture recognition when a 'silence' class sample was input, and false negatives refer to the model incorrectly predicting 'silence' when another class was given.

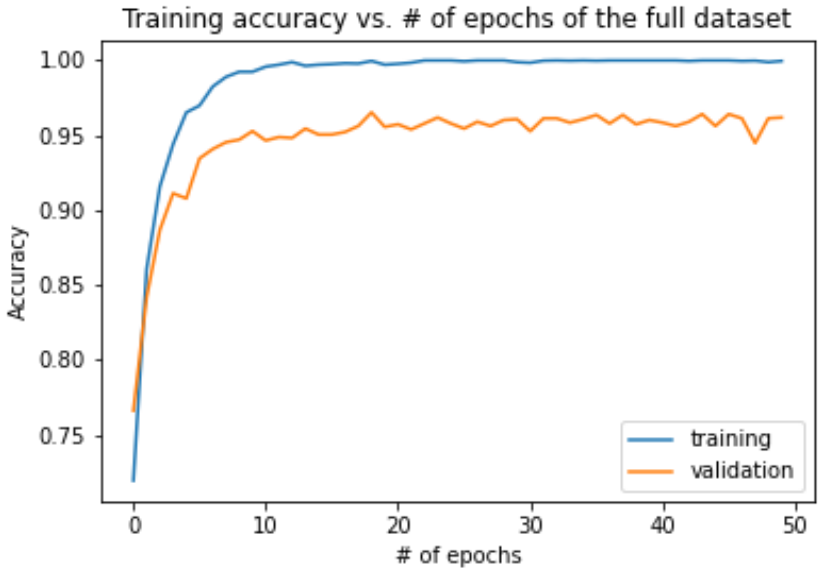


Figure 5: Training and Validation Accuracy vs. Epoch.

In the accuracies plot it is apparent that the model trained completely, and that the test accuracy of 95.9% was reasonable.

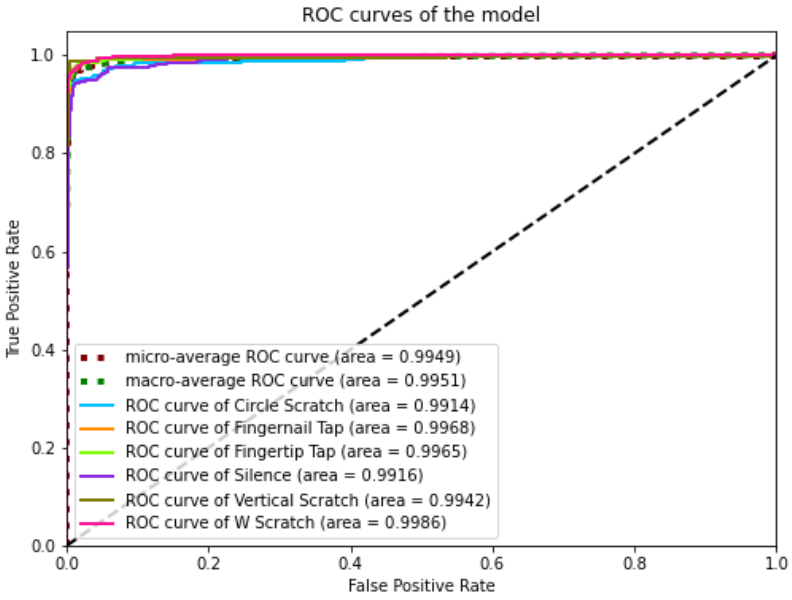


Figure 6: Receiver-Operating Curves for each class.

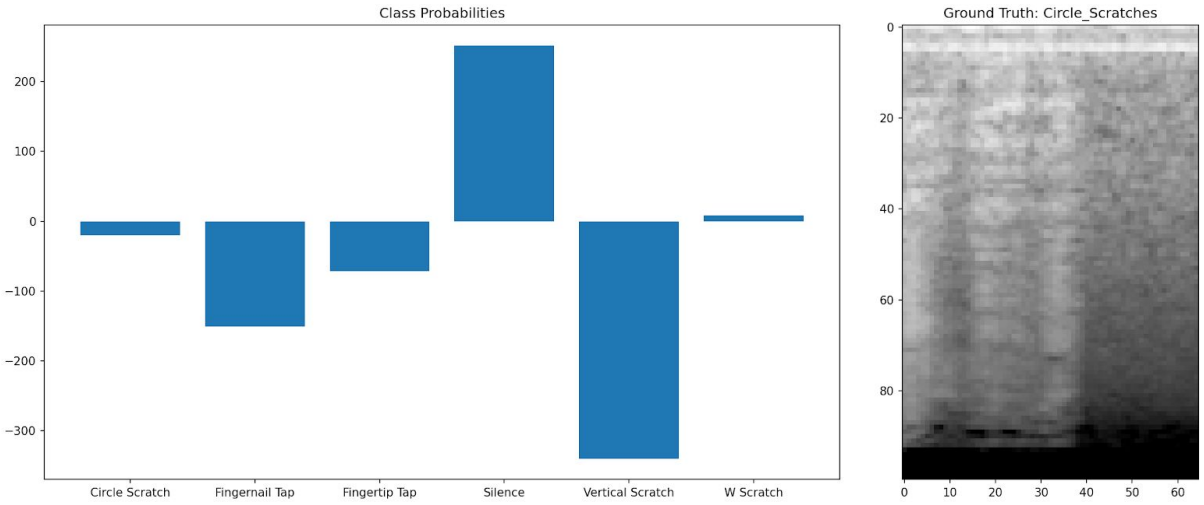
In *Figure 6*, The AUC is almost the maximum of 1, which indicates that the number of misclassifications was extremely low.

	Circle	Fingernail Tap	Fingertip Tap	Silence	Vertical Scratch	W Scratch
Circle	266	0	0	5	0	3
Fingernail Tap	2	246	5	2	1	2
Fingertip Tap	1	4	308	4	1	1
Silence	5	0	3	308	2	2
Vertical Scratch	2	1	0	2	288	1
W Scratch	3	0	0	6	0	283

Table 2: The confusion matrix confirms that in all classes, the false positive rate is below 5%.

Qualitative Results

The Circle class had the most confusion of the “positive result” classes by a wide margin, and was examined to determine whether the model’s flaw that caused this could be reasonably fixed.



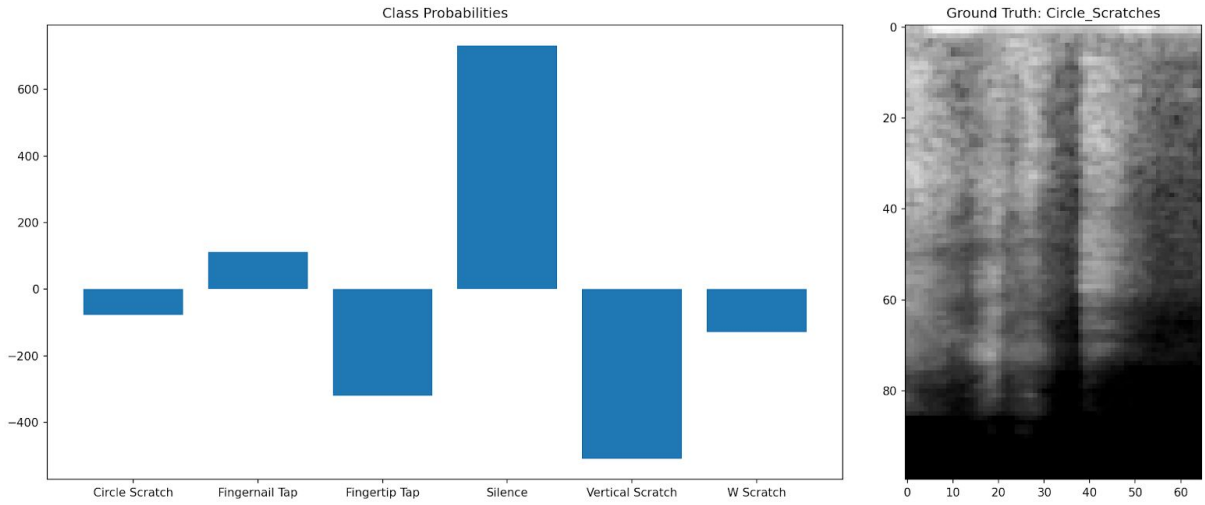


Figure 7-8: Ground truth for both samples is 'Circle Scratch', but the model incorrectly classifies them as 'Silence' with high confidence.

In *Figure 7-8*, the classifier mis-classified the circle class as silence.

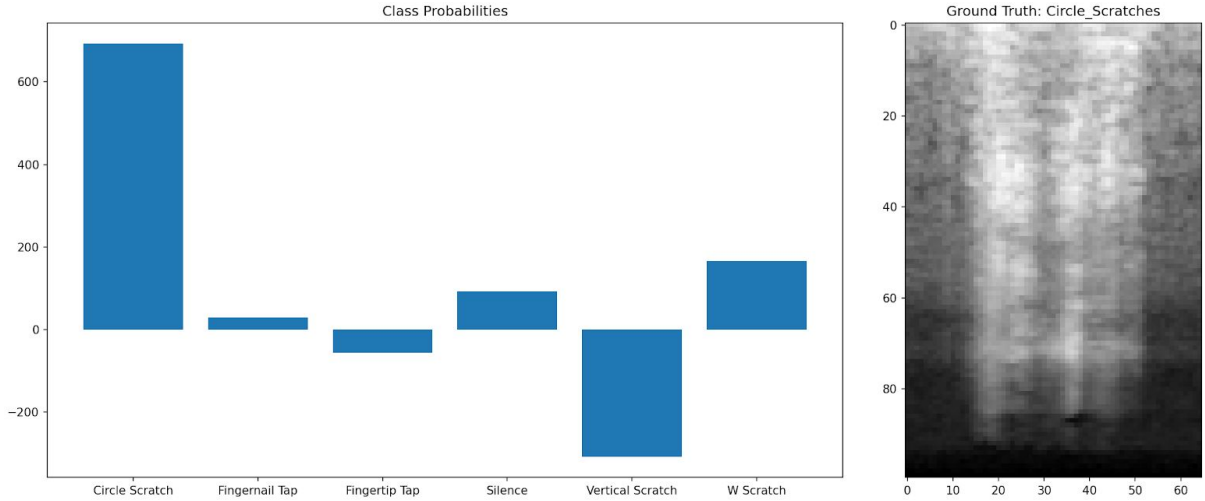


Figure 9: Correctly classified example from 'Circle Scratch' class.

In *Figure 9*, the classification occurred correctly.

Reflection

Discussion & Learnings

Given that the model achieved a final accuracy of 95.7% with few false positives cases, we considered the model to be well performed. From the user experience perspective, a false trigger is significantly worse than having to repeat a gesture, provided that the model is below the 5% confusion threshold for any class. The most confused in the model was Silence, which is ideal because the class carries a low risk false negative prediction.

Circle Scratch carried the most confusion among positive classes. One reason for this is that the duration of this class can exceed the width of the images being input to our model. It also has a low signal-to-noise ratio. Circle Scratch is simply a long drag, which is visually similar to white noise in some cases (*Figures 7-8*). In *Figure 9*, however, there was a clear definition of silence on both sides, which seems to help our model pick out the scratch.

Table 3 lists the best accuracies of the model using different image transformation methods, with the highest accuracy achieved from padding, lowest accuracy from cropping, and scaling (squeeze/ stretch) in the middle. This is reasonable since information was lost from cropping the images, and distorted from applying scaling methods, but was mostly retained in padding.

	Cropped	Padded	Scaled Small	Scaled Large	Scaled Medium
Seed	6	6	6	6	6
Learning Rate	0.1	0.1	0.1	0.1	0.1
Batch Size	48	48	48	64	48
Kernels	12	4	12	12	8
Accuracy	**95.90%**	**97.15%**	**96.41%**	**96.36%**	**96.07%**

Table 3: Best model accuracies for different image transformations

Unfortunately, alternatives to cropping require knowledge of the gesture length beforehand, which is infeasible for practical applications. To explore the best frame width in the future, one should obtain individual audio gestures with longer durations, such that different cropped lengths can be experimented.

Some other improvements of the project include adding more gestures to identify, and collecting more data to cover more test subjects and microphones. To improve the efficiency of the product, we would explore the best model with the least amount of parameters and processing time. Furthermore, though our procedure enables comparison to [1], further investigation is required to determine the model's ability to generalize to novel noise/data distributions.

Depending on performance on novel data, model calibration (e.g. via transfer learning) may be warranted.

Ethical Framework

Our stakeholders include ourselves, all users of touch devices, smartphone manufacturers and firmware designers, and those with damaged devices or disabilities that prevent them from using a conventional touch input.

Nonmaleficence applies evenly across all of our stakeholders. The most critical factor here is that the number of false positives of the model is as low as possible. False positives can be damaging and cause harm for any user of the application, and cause harm to the firmware designers who would implement our model as users would lose faith in their devices.

Beneficence best applies to the disabled who would need these accessibilities in order to use devices properly, and the manufacturers who offer additional accessibility features over competitors. Accuracy and speed of our model are the most important for this principle because they represent the lay man's perception of performance [3], which is the main factor that matters to users when measuring how good a tool is for them.

Autonomy is the most unbalanced, affecting almost exclusively those with damaged iphones and disabilities. False negatives are the greatest factor in autonomy because they will cause no input, where our key stakeholder has no other reliable source of input, thus the failure of our model would restrict this stakeholder's autonomy particularly.

Justice will affect those with disabilities most, as they are the most unjustly impacted in conventional device use. The accuracy of this model will improve accessibility for disabled individuals, and though this improvement does benefit all stakeholders, it is most beneficial to those who have no other option.

In summary, with the goal of helping the greatest number of stakeholders the most, it is best to focus on nonmaleficence, which also has a byproduct of being great for autonomy. All stakeholders will see great benefit from this, and later improvements can be focused on beneficence, which will also increase justice and fill the full intended purpose of the project.

References

[1] C. Harrison and S. E. Hudson, “Scratch input,” *Proceedings of the 21st annual ACM symposium on User interface software and technology - UIST '08*, Oct. 2008.

[2] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, “CNN architectures for large-scale audio classification,” *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2017.

[3] A. Murata, “Empirical evaluation of performance models of pointing accuracy and speed with a PC mouse,” *International Journal of Human-Computer Interaction*, vol. 8, no. 4, pp. 457–469, 1996.

Permissions

Adam Carnaffan

Permission to post video: Yes
Permission to post final report: Yes
Permission to post source code: Yes

Aman Bhargava

Permission to post video: Yes
Permission to post final report: Yes
Permission to post source code: Yes

Alice Zhou

Permission to post video: Yes
Permission to post final report: Yes
Permission to post source code: Yes