Accent Recognition with Neural Networks

Aristophanes Albertus Alvin (2928175),
Georgios Christopoulos (2789175),
Andrei-Alexandru Ionescu (0792918),
Gustavo Luis Luchetti (9871969),
Peter Alex Mahhov (4717708),
Niels Thomassen (6535674)

***Abstract*—In this paper, we focus on training a Convolutional Neural Network model and a pre-trained model to detect the native languages of people speaking in English, to accuracies of 29% and 22% respectively, out of a set of 53 different possible languages. The models perform with much larger accuracies (75% and 65%) when classifying only the primary language group of the native language of the speakers.**

We tested the same classification task on human evaluators and found that the performance of our models matched closely the accuracy of the humans, both for language group and specific language classification. Our models were trained, validated, and tested on the Speech Accent Archive database. The robustness and generalizability of the models were further tested by having them classify samples from the IDEA database, to expectedly poor results.

Codebase is available at:
https://github.com/PMahhov/Accent-Detection

I. Background

Neural networks have recently become very popular for classification, pattern recognition, and clustering among different disciplines, due to their ability to infer features without them being explicitly set [1]. They are also particularly valuable when models have complex features, which is the case for the objective of our project, accent recognition. Accent recognition (or detection), meaning identifying the native language of a speaker when they are speaking a different one, can be useful when it comes to the development of automatic speech recognition (ASR) systems, or most commonly nowadays, digital assistants.

In that regard, there is a vast amount of prior work in the domain of spoken language processing, especially accent recognition. A paper by Jiao, Tu et al. [2] combines neural networks and recurrent neural networks trained on long and short-term features. It is noteworthy that the dataset used in their project is very noise heavy, the samples being recorded through low-quality head-mounted microphones, whilst our data source, the Speech Accent Archive [3] (see section III-A), has vetted the available samples as to only keep higher-quality audio samples. However, their model had trouble identifying between similar languages, such as some that come from the same region. Since these languages use similar phonemes, it is possible that our models suffer from similar shortcomings due to this as well, as further discussed at the end of section VI-A.

Hautamäki et al. [4] attempted to characterize speech samples into specific attributes, facilitating the process of identifying foreign accents in English and in Finnish, achieving greater accuracy improvements in the latter (20%+). Al-Jumaili et al. [5] analyzed the use of neural networks to identify American, British, and Indian accents from publicly available resources, with a focus on the security domain, specifically, to detect lies in border control. Their experimental results indicate that deep learning algorithms can be robust and highly accurate when applied to this specifically constricted problem domain. Finally, in their article, Blackburn et al. [6] attempted to determine the accent of subjects whose first languages are Arabic, Chinese, or Australian English. The classification is performed in stages: all speech is divided into segment types, namely ”voiced”, ”unvoiced”, ”stop”, or ”energy dip”, and for each of the segment types an artificial neural network is used to classify the accent. The downsides of the approach of this article are the fact that the used dataset is small, and that the segmentation process could have been more developed.

II. Our Work

In this experiment, we create a convolutional neural network from scratch and train it with the Speech Accent Archive dataset (as further explained in subsections III-A and IV-B). As a performance benchmark, we implement a second model that utilizes transfer learning, which enables it to take advantage of a pre-trained model that functions as its feature extractor (as further described in subsection IV-C).

We train and test the performances of both the custom-built and the pre-trained models using the same dataset to make a valid comparison. To establish a better understanding of the accuracy of humans in the task of accent recognition, we surveyed humans who were given the task of identifying the native language of the speaker (after selecting its primary and secondary language groups) based on a randomly selected sample from the same dataset. Finally, we test the robustness of our trained models on a different dataset (International Dialect of English Archive, or IDEA, for short) that contains phrases other than the ones the models were trained with. The purpose is to see how well the model is able to generalize to new and unseen data.

As briefly explored in Section I, our project attempts to overcome a few of the challenges that previous projects have faced. In using a training dataset that has already gone

through a thorough vetting process, we can focus our efforts on modifying and tuning our neural network. Secondly, by training both with specific languages and language groups we can have a more comprehensive understanding of what human capabilities are in comparison with neural network based models. Lastly, by using a completely different dataset to test our model, we can properly identify if it is capable or not of performing outside its initial training characteristics, a weakness that is very common to neural network models.

III. Data Description

A. Speech Accent Archive

The Speech Accent Archive [3] is a large dataset of 3002 audio samples (as of the beginning of January 2023). Each audio sample contains a recording of a single speaker saying a particular four-sentence 'elicitation paragraph' out loud (see Appendix I for the full text). This passage was carefully constructed by a professor of Linguistics to contain only common English words, but with a variety of "difficult English sounds and sound sequences", while making sure that it contains "most of the consonants, vowels, and clusters of standard American English" [3].

The samples are labelled according to the speaker's native language and are also accompanied by biographical data for each speaker. Other available metadata includes the speaker's birthplace, age, sex, English learning method (naturalistic or academic), age of English onset, and if they are resident in a majority English-speaking country, then where and for how long they have been residing there. Any of these parameters could theoretically be used to construct class labels for our model to predict, but we are focusing on native language as our primary prediction goal. The database also contains a phonetic transcription of the speech, and a comparison of said speaker's accent with the General American English dialect, but this information will remain unused in our analysis.

Not all languages are equally represented in the database, so we limit our dataset to only contain those languages that have at least ten or more samples in the Speech Accent Archive. As illustrated in Figure 1, this results in only keeping 53 out of the 228 languages represented in the Archive, substantially lowering the number of class labels required in the model. However, this does also have the effect of lowering the total number of samples from 3002 down to 2456, which is why the dataset had to be artificially augmented to increase the sample count (see Section IV-A).

Furthermore, the number of native English speaker samples (651) is much larger than any other language represented in the dataset, more than the number of samples of the three most represented non-English languages put together (these being Spanish, Arabic, and Mandarin, with 235, 200, and 156 speaker samples respectively). This unequal distribution of languages can skew a random sampling towards having a disproportionately large number of native English speakers.

B. IDEA Archive

The IDEA Archive [7] is an additional dataset that we use for robustness testing. The methodology behind this database is significantly different from the Speech Accent Archive; while it still contains voice recordings, the spoken phrases are different, variable, and much longer. Quite a few samples do not have complete information about the speaker, so we do not check our models against the entirety of the IDEA dataset, but a manually curated selection of it, which is 93 samples of 9 languages in total. This check can provide valuable information about the generalizability of our models, which are initially trained on the single 'elicitation phrase' of the Speech Accent Archive as explained above.

IV. Methods

A. Data Editing and Augmentation

In order to increase the amount of data our models can train on, and to also improve the quality of the dataset, we performed a number of augmentations on the audio samples in the dataset.

Firstly, we downsampled the audio files to 16 kHz and removed leading and trailing silent segments. Then, we applied the following transformations on each sample, generating 4 new derived samples:

- **White noise**: We added randomized (normally distributed) white noise to each sample. [8], [9]
- **Pitch scaling**: We scaled up the pitch of each sample by 4 semitones. [9], [10]
- **Time dilation**: We sped up each sample by 25%. [9], [10]
- **Random gain**: We increased the volume of each sample by a randomized factor between 2x and 4x. [11]

These transformations are commonly used, and often serve the additional purpose of potentially increasing the robustness of audio-based predictive models, at the cost of occasionally lowered model accuracy. [9]

After augmenting the samples, we divided the dataset into a training set, containing 10,400 samples (80% of the total), and validation and test sets, each containing 1,165 samples (10%). When dividing the dataset, we took care to maintain the proportion of languages within each subset and made sure to allocate the augmented samples in the same set as the original, to avoid data leakage.

For the purposes of our custom Convolutional Neural Network, the Fast Fourier Transform was applied to each of the audio files to transform the samples into Mel-Spectrogram images (see Figure 2), which provide representations of the sound's amplitude (loudness) over different frequencies as it varies over time. This enables the neural networks to identify and distinguish different patterns in accent, phoneme pronunciation, and intonation.

We extracted the Mel-Frequency Cepstral Coefficients (MFCCs), and the training set was transformed to a tensor of 10,400 samples, with each sample containing an array of 20x3932 as an input to the input layer. Due to the fact that

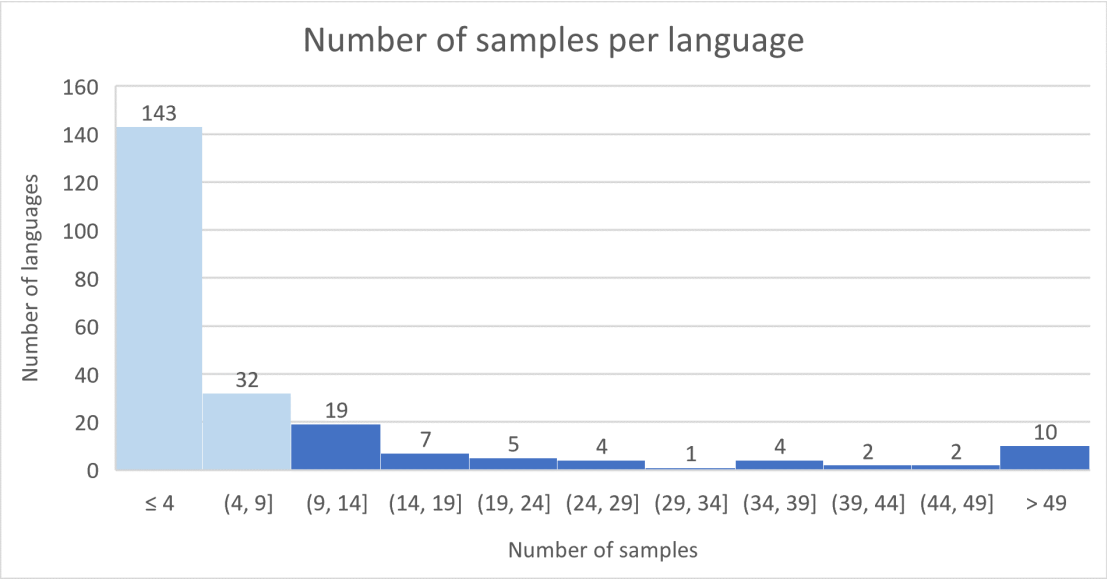

Fig. 1. A histogram showing the number of languages in the Speech Accent Archive per the sample counts.

the audio clips all have different lengths, the MFCC would have produced variable-sized arrays in the dataset if this was not addressed. Therefore each array was zero-padded so they can reach the same size and be fed into the network which requires a fixed-size input shape.

B. Custom Convolutional Neural Network

Convolutional Neural Networks (CNNs) are well-suited for sequential data, such as images, which have an ordered structure based on the topology of the object. In our case, these are the distinct frequencies of the spectrogram that we get after converting the audio file into an image (as described above in section IV-A. CNNs are well-suited for such tasks thanks to their use of convolutional layers, which apply a set of filters to the image in order to extract features at different scales. Additionally, CNNs use pooling layers to down-sample the feature maps, which helps to reduce the dimensionality of the data and makes the network more computationally efficient.

Designing a custom CNN involved striking a careful balance between complexity and performance. The final result was a 12-layer architecture, of which the first two layers of the network are convolutional 2D layers, as is common in works that tackle similar issues [12] [13]. A larger number of layers would have led to a network that needs more data to perform well, and a low number of samples is a limitation of our dataset. Even with the augmentation work described in section IV-A, we still end up having 12,731 samples for 53 languages, which on average results in only 240 samples per language (discounting the disproportionality of the dataset discussed at the end of section III-A). Adding more convolutional 2D layers would have also increased the amount of time it would have taken to train the model, which was not a trivial concern, considering our limited computational resources.

The 2D convolutional layers both use a kernel size of (3,3), but they differ in the number of filters, the best values of which were found through hyper-parameter tuning to be 32 for the first and 64 for the second. By choosing an odd kernel size, we make sure that all the pixels of the previous layer will be symmetrically set around the output pixel, and by keeping it small, we reduce computational costs. It is reasonable that the deeper convolutional layer has twice as many filters as the first one, as that way the model can extract more abstract features from the images.

In between the layers we used batch normalization to standardize the output of the layers by maintaining the mean output close to 0 and the output standard deviation close to 1. Normalizing the batches should allow the network to converge faster and to be more stable when training [14].

The 2D convolutional layers use the RelU activation function, which simply returns 0 for negative inputs, and leaves positive inputs unchanged.

The layers after that are max-pooling layers with a size of 2x2, which downsize the feature maps to 9x1966, and 3 dropout layers, which (in order) activate only 50%, 20%, and 10% of the neurons in order to prevent overfitting.

After the convolutional, max pooling, and first two dropout layers (but before the two dense layers and the third dropout layer, see table I), we apply a flattened layer to make the result an array. This is necessary because the following layers expect one-dimensional arrays, while the convolutional layers work with multi-dimensional ones.

After this, there are 2 fully connected dense layers, one of size 128 using RelU, and an output layer of size 53 (representing the number of classes), which uses the softmax activation function to produce a probability distribution over the possible classes. This helps with mapping the raw output of the dense layer to a normalized probability distribution by

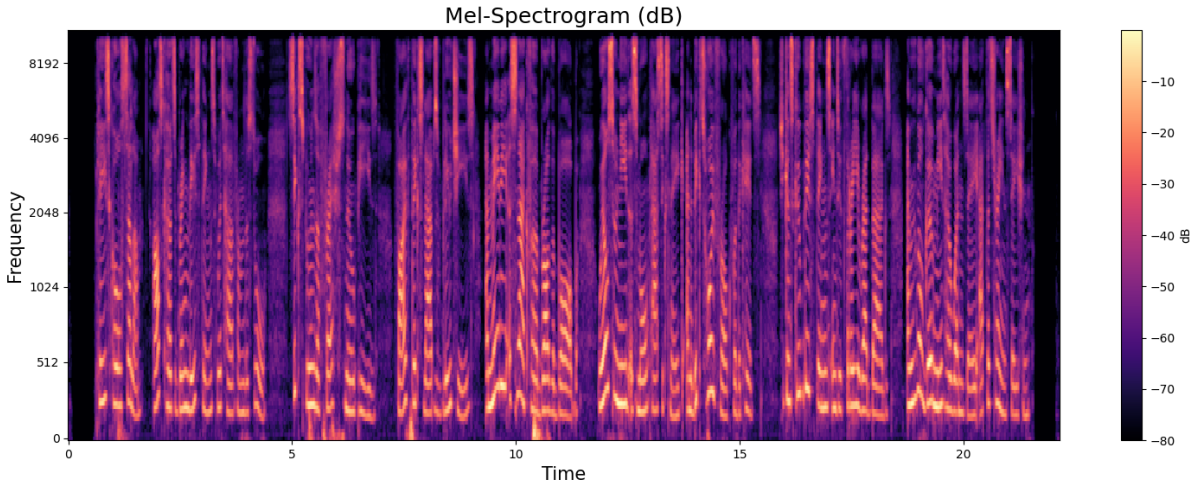

Fig. 2. The Mel-Spectrogram of one of the samples transformed from the Speech Accent Archive.

squashing the outputs through an exponential function, which ensures that the output probabilities always sum up to 1.

For this model, we use the 'Adam' optimizer, a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments [15]. From hyperparameter tuning, we found the best parameters for the optimizer to be a learning rate of 10^{-5} and a decay value of 10^{-4}. We use the loss function of 'categorical cross entropy for the optimizer (because all our variables are categorical data) and 'accuracy' as the metric for training and validation, as recommended by Keras documentation [14].

After each epoch, when the model trains, the callback function checks for early stopping conditions. These are criteria that we have set to avoid overfitting. The early stopping conditions have three parameters: validation loss (val loss), patience, and minimum delta. The first parameter represents the loss value of the validation step. The program monitors the change of val loss over the training epochs and stops if it does not increase in the patience period. The minimum delta determines the smallest increase of val loss that counts as an improvement, and the patience parameter determines the number of epochs that the program is willing to wait between increases until it triggers an early stopping. Our work uses a minimum delta of 0.05 and a patience of 10, therefore if the val loss increases by less than 0.05 for 10 epochs in a row, then the training is stopped early. Early stopping helps avoid overfitting when the network starts to improve very slowly or plateau.

Finally, for model-fitting, we use mini-batches of 32 samples. This requires less memory than using the entire dataset, the network converges faster and the gradients remain more stable.

C. Pre-trained Neural Network

We compare our custom-built model to a state-of-the-art model called YAMNet, which is an audio event classifier trained on the AudioSet dataset to predict audio events from the AudioSet ontology [16]. AudioSet contains a wide variety of human-labelled sounds extracted from over 2 million YouTube videos. While about half of the samples in the database are of speech, which is relevant to our interests, the other half consists of music and various different environmental sounds, which are not [17]. Hence YAMNet is not a model designed to work on speech in particular, but audio in general.

YAMNet accepts a 1-D tensor of audio samples with a sample rate of 16 kHz. As output, the model returns a 3-tuple: scores of the classes, embeddings, and the log-mel spectrogram of the entire audio frame. For this section of our work, rather than training a model from scratch, we use transfer learning where we use YAMNet as our feature extractor.

When building the process and architecture of our pre-trained model, we follow the steps of a similar experiment performed by Keras, where YAMNet is utilized through transfer learning to recognize the English accents from Ireland and the United Kingdom [18]. The main difference is that our experiment involves not only native English speakers from the British Isles but also both native and non-native English speakers from across the globe. Furthermore, the target classification in our experiment is wider since we identify native languages and language groups, rather than local accents. In our work, all of the different native English accents distinguished by the Keras experiment (and others, such as accents from Australia and the USA) would be labelled under the same, Native English Speaker class label.

The process of preparing our pre-trained model is as follows. First, we re-sample the audio files to a 16 kHz frequency. YAMNet then takes the re-sampled audio and generates the embeddings, which are the average-pooled output that feeds into the final classifier layer. Finally, we train four pairs of dense and dropout layers which take the embedding data

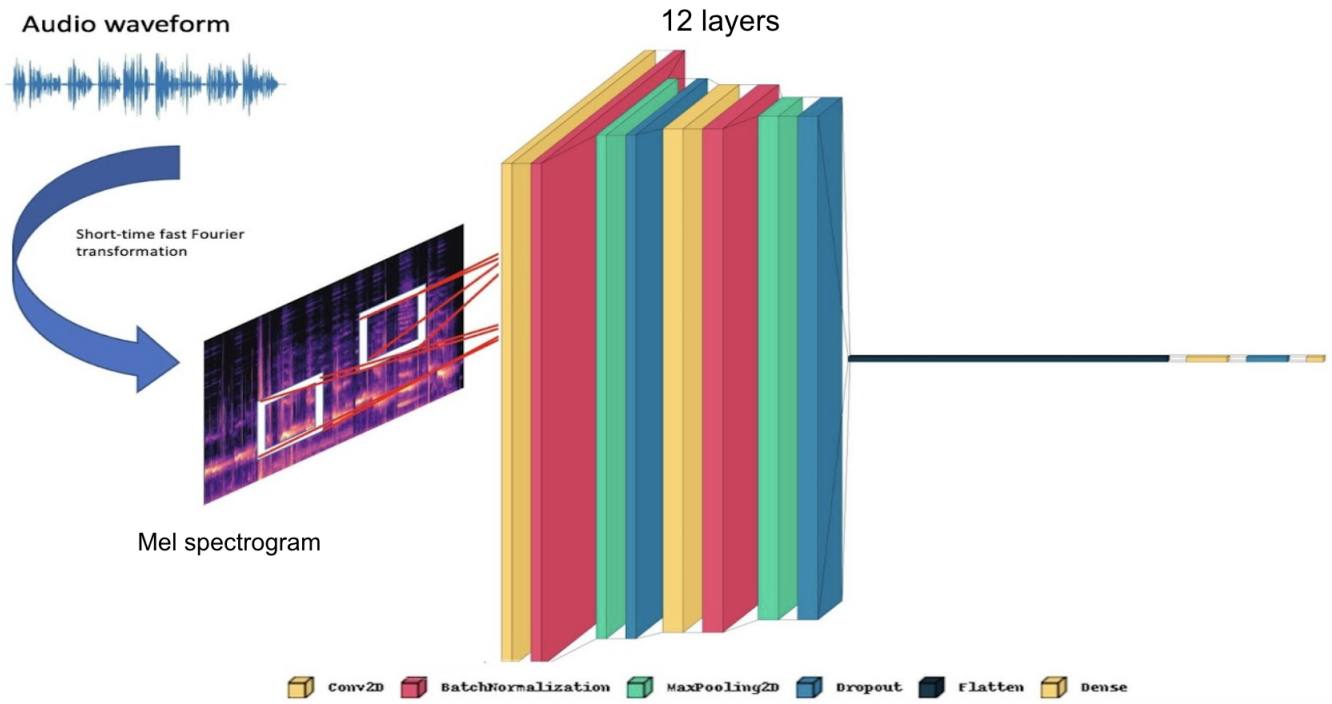

Fig. 3. A visual summary of the architecture of our custom convolutional neural network.

Layer	Output Shape	Number of Parameters
conv2d	(None, 18, 3932, 32)	320
batch_normalization	(None, 18, 3932, 32)	128
max_pooling2d	(None, 9, 1966, 32)	0
dropout	(None, 9, 1966, 32)	0
conv2d_1	(None, 7, 1964, 64)	18496
batch_normalization_1	(None, 7, 1964, 64)	256
max_pooling2d_1	(None, 3, 982, 64)	0
dropout_1	(None, 3, 982, 64)	0
flatten	(None, 188544)	0
dense	(None, 128)	24133760
dropout_2	(None, 128)	0
dense_1	(None, 53)	6837

TABLE I
AN ORDERED LIST OF THE LAYERS OF OUR CUSTOM CNN, CONTAINING FOR EACH LAYER ITS SHAPE AND NUMBER OF PARAMETERS.

as input and return the classification of the audio (see Fig. 4). We use the same neural network architecture as in the Keras experiment [18] where there are four pairs of dense and dropout layers with 256, 384, 192, and 384 units (in order respectively).

D. Human Evaluation

The goal of human evaluation is to have some performance data which we can compare to the classification results of our neural networks. In order to achieve this we set up our human evaluation process in a way that best emulates the testing task set to our neural network models. We created a survey that provides the evaluator with an audio file that they can listen to and guess the native language of the speaker.

The samples used in our evaluation were representative of our entire sample set. To achieve this we wanted the number of samples used to be relatively high, while at the same time not giving the evaluators too many questions to answer. Thus, we randomly sampled 40 samples from our data set and split these among 4 different possible survey sets, one of which would be randomly assigned to each evaluator. The data set we sampled from is the same data set used in the training and testing of our neural network, which is to say that all languages used in the set are those that contain at least 10 samples of the same language in the full Speech Accent Archive dataset. However, we did exclude any samples created by us through data augmentation processes, choosing to use only the unchanged speech recordings for human evaluation purposes.

The neural network's task was to classify a language from a sound file, out of a total of 53 different language options to choose from. In order to make the human evaluation task compatible with the model classification task, we also gave

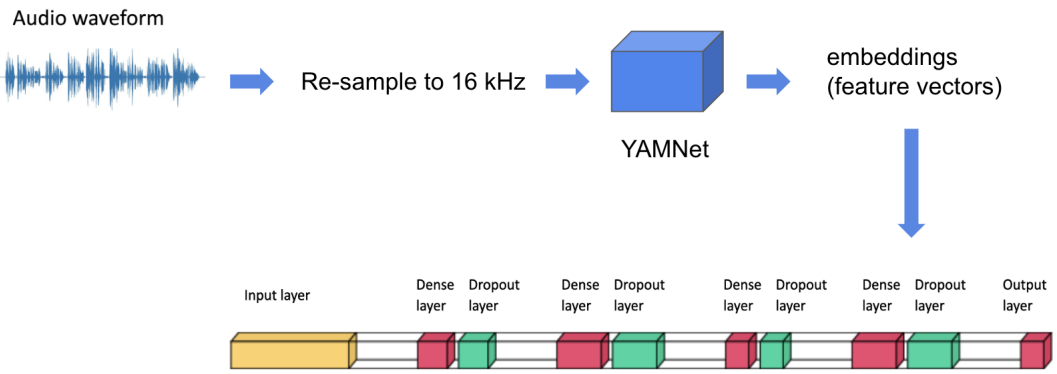

Fig. 4. Process and architecture of the pre-trained model.

the evaluators 53 languages to choose from. In order to make it clearer for the evaluators we split these languages into language groups, creating a decision tree interface that would make the language picking a lot clearer. We used the language groups as defined by the Ethnologue database (25th edition) [19].

Languages within the same language group tend to exhibit similar sounds. As an example, French, Spanish and Italian are all of the Indo-European primary language group (and the Italic secondary language group) and thus have some common or similar words and pronunciations. This can make it harder to detect the differences between the English accents of native speakers of two such languages. However, the main problem as reported by our mostly European-based evaluators is the non-European languages present in the sample, which could be harder to detect due to unfamiliarity with how they sound. However, people might still have an idea of the sounds prevalent in wider language groups, even if they do not know the difference between the specific languages themselves. Thus, we will record both the human evaluation accuracy of both specific languages and also language group classification for further comparison with our models' performance.

V. Results

A concise summary of our model accuracies for each task and database combination can be found in Table II.

A. Custom CNN

The tuning of the hyperparameters of the custom CNN was implemented using the Hyperband algorithm.[20] Hyperband provided good results, but due to hardware limitations, the resources were exhausted after tuning for 8 hours on a cloud-based computer, so we decided to halt the process. There were many parameters that had to be tuned. To determine the best filter sizes we ran the Hyperband algorithm for the first two Convolutional layers with the values of (32 and 64) for the first layer and for the second layer we defined the search space to be either 32, 64, or 128. The options for the activation functions of the layer were more limited. We excluded sigmoid and tanh because they tend to saturate the neurons by constraining the values between [0,1] for the former and [-1,1] for the latter. Instead, we used RelU which is more efficient, simpler and does not saturate. For the dropout layers, we experimented with a range of values from [0.1,0.5]. Additionally, there were attempts to use the Adadelta optimizer in the beginning since it seemed to produce better results, but later Adam proved to give higher accuracy on unseen data. The learning rate was tuned with the values [1e-2, 1e-3, 1e-4,1e -5] and the weight decay with the values [0.0, 1e-2, 1e-3, 1e-4,1e-5] to encourage the network not to accumulate large weights and start overfitting. Last but not least, the dense layer before the output was tuned in the space [128,512] with a step of 128 every time. The chosen hyperparameter values are displayed in Table III.

Hyperparameters	Values
First Conv filter size	32
Second Conv filter size	64
First Droupout layer	0.5
Second Dropout layer	0.2
Third Dropout layer	0.1
First Dense layer	128
Learning rate	10^{-5}
Weight decay	10^{-4}

TABLE III
BEST HYPERPARAMETER VALUES FOR CUSTOM CNN

The 12-layer custom CNN reached an accuracy of 29.35% and a final validation loss of 3.16 on the augmented Speech Accent Archive database for specific language classification (see Figure 5), and a 75% accuracy and validation loss of 1.59 for classifying the primary language groups (see Figure 6).

When tested on the IDEA dataset, the model achieved an accuracy of 1.3% for specific language classification and 5% when classifying the primary language groups.

B. Pre-trained Model Results

We attempted to tune the hyperparameters of the pre-trained model with the Keras Tuner. However, after more than 10 hours of tuning, we decided to abort it due to time constraints.

Dataset	Model	Primary Language Group Classification	Specific Language Classification
Speech Accent Archive	Custom CNN	75%	29.35%
	Pre-trained NN	65.51%	22.22%
	Human evaluation	67%	27%
IDEA	Custom CNN	5%	1.3%
	Pre-trained NN	0%	0%

TABLE II

A SUMMARY OF THE CLASSIFICATION ACCURACIES ACHIEVED BY BOTH MODELS AND HUMAN EVALUATION AT IDENTIFYING SPECIFIC LANGUAGES, PRIMARY AND SECONDARY LANGUAGE GROUPS.

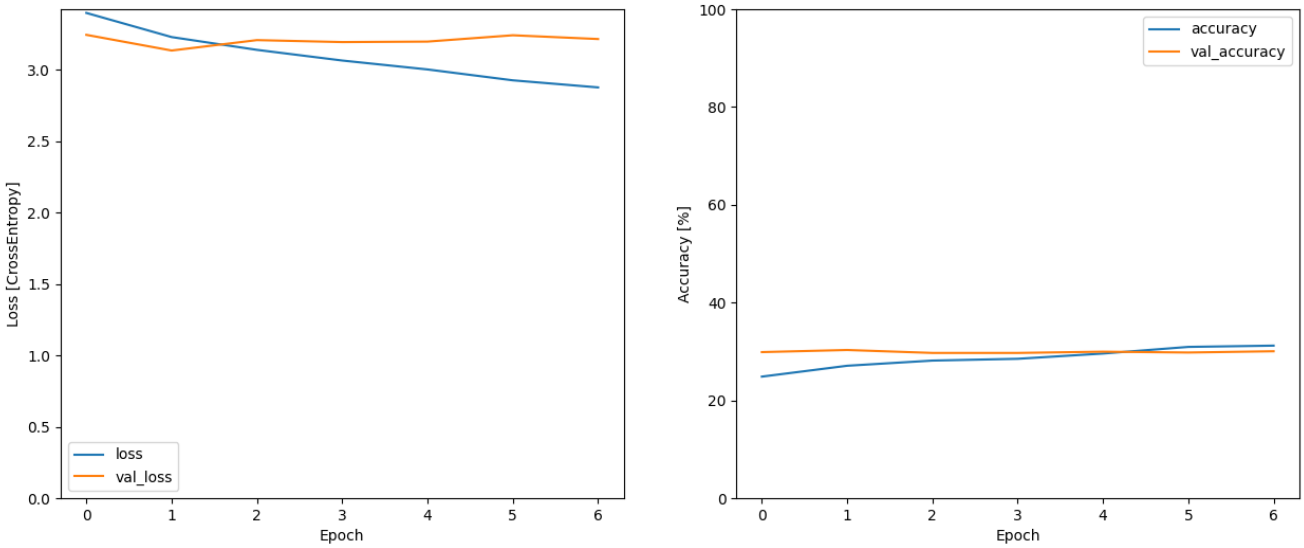

Fig. 5. The cross-entropy loss (left) and the accuracy (right) exhibited on the training and validation sets during the training of the custom CNN in individual languages.

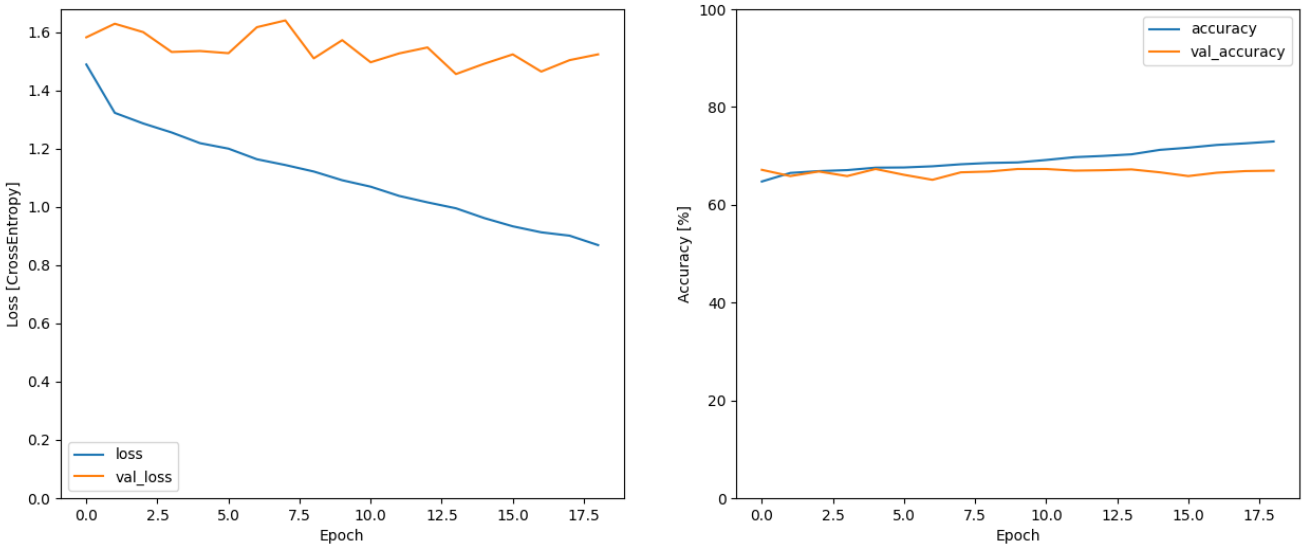

Fig. 6. The cross-entropy loss (left) and the accuracy (right) exhibited on the training and validation sets during the training of the custom CNN in language groups.

We experiment with several values of the dropout rate (0.25, 0.3, 0.5, 0.8), learning rate (10^{-3} and 1.9644×10^{-5}), and type of optimizer (Adam and Adadelta) in order to optimize the model's accuracy. We managed to obtain the best specific language classification accuracy of 22.22% and loss of 3.96 using a dropout rate of 0.8 across all layers, a learning rate of 1.9644×10^{-5}, and the Adam optimizer (see Fig. 7).

The same model architecture and configuration are trained and tested on the same audio dataset but with different classifications, this time targeting the primary language groups instead of the specific language, resulting in fewer classes (14 instead of 53). The model achieved 65.51% accuracy and 2.51 loss on the test set, with performance on the training and validation, set depicted in Fig. 8. It was also tested on the IDEA dataset to see its capability to generalize but only resulted in 0% accuracy and 2.64 loss.

C. Survey Results

We had a total of 23 respondents, the majority of respondents were Danish and German, as can be seen in figure 9.

The respondents were quite confident with their English ability. They were asked to self-grade themselves in English proficiency on a scale from 1 to 10, with 10 being native speakers. The lowest given score was a six and the highest was a ten with an average grade of 8.4.

The survey results are summarized in Table IV. Due to the decision-tree style structure of our answering choices, in addition to gathering data about the classification of specific languages and primary language groups, there is also information about the accuracy with respect to the secondary language groups, such as Germanic or Italic for the Indo-European group, for example. Observe that the accuracy of specific languages goes down when we remove all English samples. The resulting percentage of correct specific language guesses corresponds to 3 correct answers out of 148 possible guesses.

VI. Discussion

A. Neural Network models

The custom CNN model performed better in classifying individual native languages, compared to the pre-trained model, with an accuracy of 29.35% as opposed to 22.22%. Furthermore, the custom CNN surpassed the pre-trained model when predicting primary language groups with an accuracy of 75%, in contrast to the 65.51% of the pre-trained model. This could be explained by the fact that when training with a small dataset, a simpler model can outperform a more complicated one due to the smaller number of trainable parameters.

A key difference between the two approaches is that the custom CNN transformed an audio classification problem into image classification. The audio files were transformed into Mel-Spectrograms, which are often used to represent accent features in audio [12] [13]. In terms of time, the pre-trained model is easier to implement since there is rich documentation that explains its functionality and is also more easily customizable.

In addition to testing the classification of individual languages, we also tested the models with a simplified version of the task, where they only had to classify the language group of the speaker's native language. This modification reduced the number of class labels from 53 languages to only 14 different language groups. Since languages that are of different language groups tend to be more dissimilar than the languages within one, we expected the models to perform better at distinguishing the former than the latter. While the limitations of the dataset (mainly the small size and disproportionate class label distribution) remain mostly intact, each individual class does have more samples overall. Therefore it is not surprising that both of the NN models have a high accuracy in correctly classifying primary language groups, with the custom CNN being more accurate than the pre-trained model once more (75% and 65.51% respectively, as seen in Table II).

These results do bring our models more in line with the state-of-the-art in accent recognition, considering the differences between our task and the most common approaches. The CNN with data augmentation developed by Sheng et al. [13] manages an accuracy of 88% when differentiating between Chinese, English, and Korean, while the CNN with unaugmented training data by Chionh et al. [12] achieves an accuracy of 77.9%, to distinguish between Arabic, Italian, Japanese, and Korean. These and other similar works only classify between three to five languages at a time, a drastically reduced number of classification labels as compared to our 53 specific languages or 14 primary language groups. Even a random guessing model will be right 20% of the time when there are only five possible classes to choose from. Additionally, the languages differentiated in state-of-the-art models tend to be all from different language groups, and would thus skip the most difficult part of our task, differentiating between intra-group language features.

While both models performed rather poorly on the robustness test (predicting on the IDEA dataset), the custom CNN achieved accuracies of 1.3% and 5% (for specific language and language group classification respectively) compared to the 0% for either of the pre-trained model. The low performance is no surprise, given the difference in the data in the training set as opposed to the samples from IDEA. As was with the Speech Accent Archive dataset, it seems reasonable that the simplicity of the custom model again won over the complexity of the pre-trained NN, in being able to correctly identify the language class at least some of the time. It appears that even though YAMNet was trained on a wider variety of sounds, it did not make our pre-trained model that was based on it any more transferrable towards differently formatted data.

B. Human Evaluation

Since the survey sample set was taken from our subsection of the Speech Accent Archive dataset, the problems with that dataset propagate to this one. Therefore, for example, our survey sample set also has a disproportionately large number of native English speaker samples, due to this disproportion being present in the Speech Accent Archive as well.

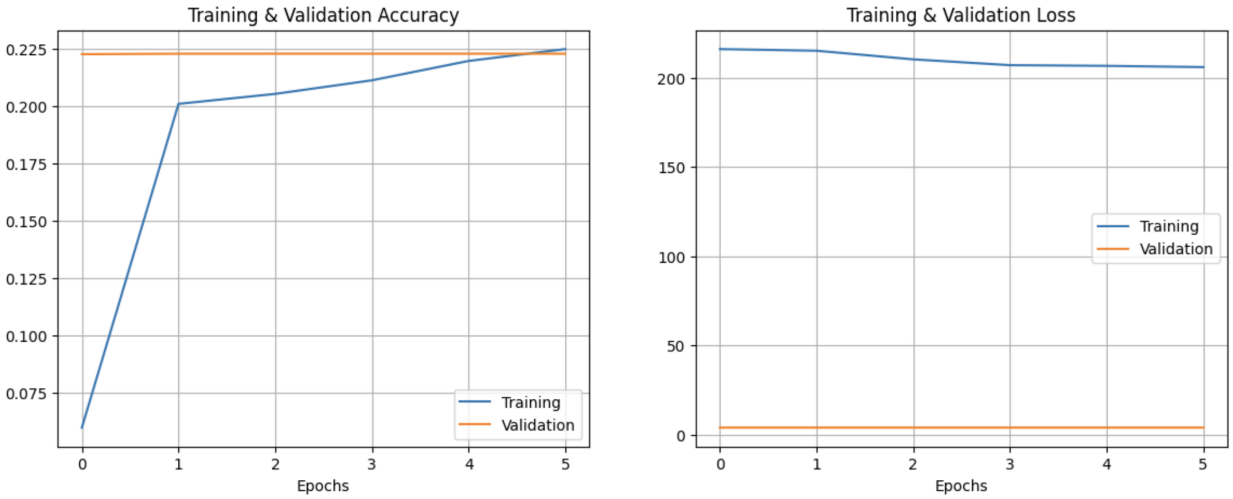

Fig. 7. Performance and parameters of the pre-trained model on classifying specific languages.

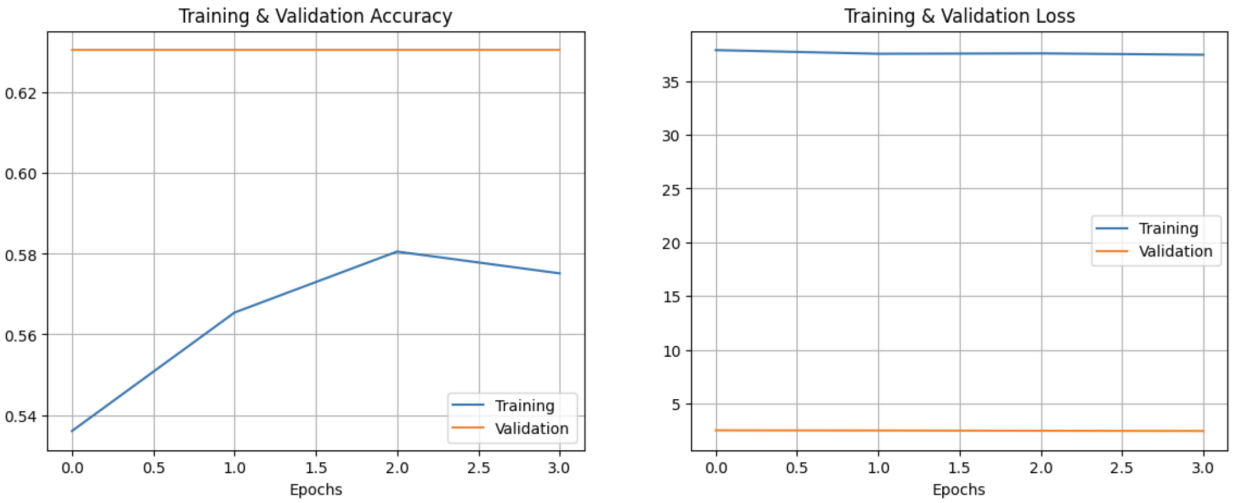

Fig. 8. Training and validation performance of the pre-trained model on the primary language group.

Human Evaluation	Primary Language Group Classification	Secondary Language Group Classification	Specific Language Classification
All languages included	67%	41%	27%
Native English excluded	53%	18%	2.0%

TABLE IV
ACCURACY OF SURVEY RESULTS, INCLUDING AND EXCLUDING NATIVE ENGLISH SPEAKER AUDIO SAMPLES. CONTAINS SECONDARY LANGUAGE GROUP CLASSIFICATION RESULTS, WHICH WERE NOT TESTED FOR IN THE MODELS.

We found that people were much better at detecting native English speakers, as opposed to classifying the language of a non-native speaker (see Table IV), which is likely to have skewed the accuracy of our results due to the unequal proportions.

Another limitation is that our survey sample was a limited subsection of our model training and testing dataset, so not all of the 53 languages used were represented in our survey. Thus, there were some native languages that the classification models were tested on, that the human evaluators never had the chance to try to classify. However, the reduced relative prevalence of all non-native English speaker samples in both sample sets might have reduced the impact of this specific problem.

Additionally, the self-graded English proficiency showed that the group of evaluators considered themselves above average to extremely good English speakers. This might have had an impact on their ability to detect native English speech. Since these individuals are more likely to consume English-spoken media and thus could likely be more experienced with different English accents. This might be a reason for the high accuracy of Native English accent detection by the evaluators.

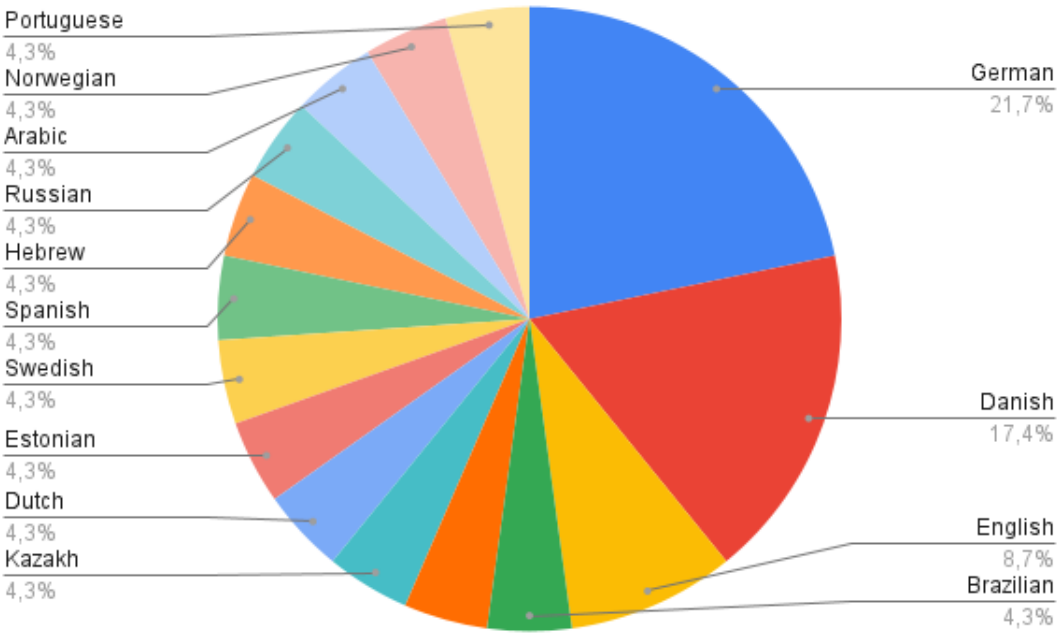

Fig. 9. Distribution of the native languages of the survey respondents

Next to that, the total responses were quite small. This results in one of our surveys not being responded to as much as the other. The mode amount of responses per survey was 7 but one of our surveys only got 3 responses due to chance. This means that this last survey is not as well represented in our final result and could've impacted our final results. Either by having too high accuracy since this last survey was relatively harder. Or by having too low accuracy since the last survey was relatively easy.

Lastly, the background of our evaluators might have affected the result of the survey. A majority of our respondents had a European native language (see Figure 9). This could potentially have added a bias to the results of our survey since people might be able to detect accents more accurately when they are similar to their own. However, due to chance, none of the respondents had their native language as a sample except the native English speakers. Additionally, most respondents are non-native speakers, and so will also commonly speak English with non-native speakers. Thus one might think that they could be more likely to recognize non-native accents. However, Table IV shows that almost no survey respondents were able to successfully identify the native languages of non-native speakers, so this effect, if present, is likely to be very minor.

When we compare the results between the human evaluators and our neural networks, we can see that they perform with similar accuracy. Our custom neural network performs slightly better than the human evaluators in both primary language group and specific language classification, and our pre-trained neural network only performs slightly worse than the human evaluators.

VII. Conclusion and Future Work

The results of this paper show the capabilities of the neural networks to perform equally as well if not better than humans, which makes Neural Networks a powerful tool for accent detection due to having been able to detect specific patterns in speech. However, there are some distinct limitations to the applicability of this technology as it stands right now. While our neural network models were trained for the sole purpose of recognizing accents, they fail to generalize and perform equally as well on a different dataset. This highlights the difficulty of building robust neural networks, a key weakness in the technology so far.

Another key limitation to our neural network models is that similar to the surveyed humans, the models had a difficult time distinguishing between the native speakers of similarly sounding languages within a language group. However, one possible way to improve on this shortcoming could be through multi-step classification. One could first run a primary language group classification model and then have trained separate models for each language group, models that would be specifically created to distinguish between the languages in that single particular group.

One possible improvement could combine a pre-trained model such as YAMNet, with a custom-build convolutional neural network such as the one built in this experiment. By utilizing transfer learning, we could take advantage of the feature extraction capability of a pre-trained model, and train the upper layers to fit to our specific dataset.

REFERENCES

[1] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, p. e00938, 2018.

[2] Y. Jiao, M. Tu, V. Berisha, and J. M. Liss, "Accent identification by combining deep neural networks and recurrent neural networks trained on long and short term features.," in *Interspeech*, pp. 2388–2392, 2016.

[3] C. M. Weinberger, Steven, "Speech accent archive." http://accent.gmu.edu.

[4] V. Hautamäki, S. M. Siniscalchi, H. Behravan, V. M. Salerno, and I. Kukanov, "Boosting universal speech attributes classification with deep neural network for foreign accent characterization," in *Sixteenth annual conference of the international speech communication association*, 2015.

[5] Z. Al-Jumaili, T. Bassiouny, A. Alanezi, W. Khan, D. Al-Jumeily Obe, and A. Hussain, *Classification of Spoken English Accents Using Deep Learning and Speech Analysis*, pp. 277–287. 08 2022.

[6] C. S. Blackburn, J. P. Vonwiller, and R. W. King, "Automatic accent classification using artificial neural networks," in *Third European Conference on Speech Communication and Technology*, 1993.

[7] P. Meier, "International dialects of english archive." http://www.dialectsarchive.com.

[8] G. An, "The Effects of Adding Noise During Backpropagation Training on a Generalization Performance," *Neural Computation*, vol. 8, pp. 643–674, 04 1996.

[9] V.-V. Eklund, "Data augmentation techniques for robust audio analysis," Master's thesis, 2019.

[10] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal processing letters*, vol. 24, no. 3, pp. 279–283, 2017.

[11] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient training of audio transformers with patchout," *arXiv preprint arXiv:2110.05069*, 2021.

[12] K. Chionh, M. Song, and Y. Yin, "Application of convolutional neural networks in accent identification," *Project Report, Carnegie Mellon University, Pittsburgh, Pennsylvania*, 2018.

[13] L. M. A. Sheng and M. W. X. Edmund, "Deep learning approach to accent classification," 2017.

[14] F. Chollet *et al.*, "Keras." https://keras.io, 2015.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.

[16] "Yamnet." https://tfhub.dev/google/yamnet/1.

[17] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, (New Orleans, LA), 2017.

[18] F. Badine, "English speaker accent recognition using transfer learning." https://keras.io/examples/audio/uk_ireland_accent_recognition/, 2022.

[19] "Ethnologue," 1 2023.

[20] A. Aceves, "Utilizing the hyperband algorithm for hyperparameter optimization." https://2020blogfor.github.io/posts/2020/04/hyperband/,, 2020.

APPENDIX 1: THE SPEECH ACCENT ARCHIVE ELICITATION PARAGRAPH

Please call Stella. Ask her to bring these things with her from the store: Six spoons of fresh snow peas, five thick slabs of blue cheese, and maybe a snack for her brother Bob. We also need a small plastic snake and a big toy frog for the kids. She can scoop these things into three red bags, and we will go meet her Wednesday at the train station. [3]