Supporting communication in children with non-typical speech using Deep Learning

James Toop

Data Science, Capstone Project

Purpose –
The importance of communication

Communication is a basic human right and ensures that people can:

- Express themselves;
- Control their environment;
- Make friends and build relationships;
- Make sense of the world around them.

Source: Foundation for People with Learning Disabilities, National Health Service (NHS) UK and British Institute of Learning Disabilities (BILD)

Purpose –
The importance of communication

Communication has greater importance for a person with a learning disability.

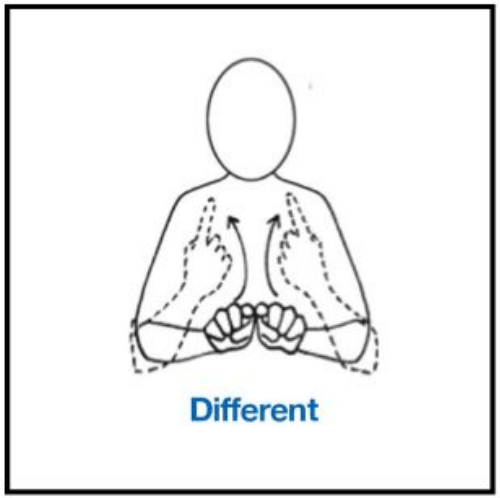
- May not able to interpret their environment as easily as others;
- Inability can cause frustration and behaviours that might be seen as challenging;
- Can lead to being marginalised or excluded by society.

Source: Foundation for People with Learning Disabilities, MENCAP

Purpose – Supporting communication

A variety of techniques have been developed to support children for whom speech is difficult:

- Communication systems such as Makaton or PECS;
- Easy read symbols;
- Speech and language therapy.



Example of Makaton signing

Source: Foundation for People with Learning Disabilities, MENCAP, The Makaton Charity

Purpose – Supporting communication

With supported communication a child with a speech disorder or severe learning disability will be better able to:

- Express their needs, feelings or opinions;
- Make choices;
- Interact with others and form relationships;
- Develop skills which will help them access learning opportunities.


Martha
U
S
R

Objective –

Build a model that can identify an audio sample of non-typical speech and suggest possible words as a tool for helping parents support their child’s communication.

Approach – The raw datasets

2 datasets totalling almost 140,000 - one second audio samples of single spoken words

- **Speech Commands:** 105,829 audio samples of 35 spoken keywords;
- **Ultrasuite:** 14,286 audio samples from 86 different children, 28 with speech disorders;
- After processing, 33,800 audio samples of 991 spoken words;

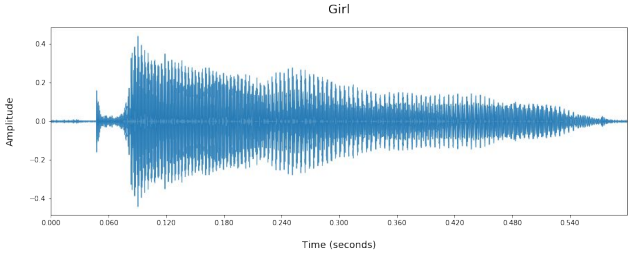
Speech Commands audio example

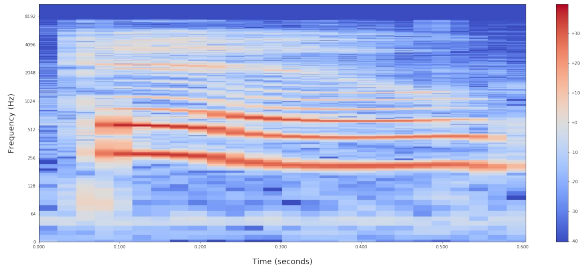
Ultrasuite audio example

Approach –
Preparing the data and creating the models

Convert audio to spectrograms and extract features

- Processed audio to isolate, resample and standardise samples then extract spectrograms and other features
- Create simple, baseline model based on reduced dataset as proof-of-concept and observed results.
- Final model uses a Convolutional Neural Network.



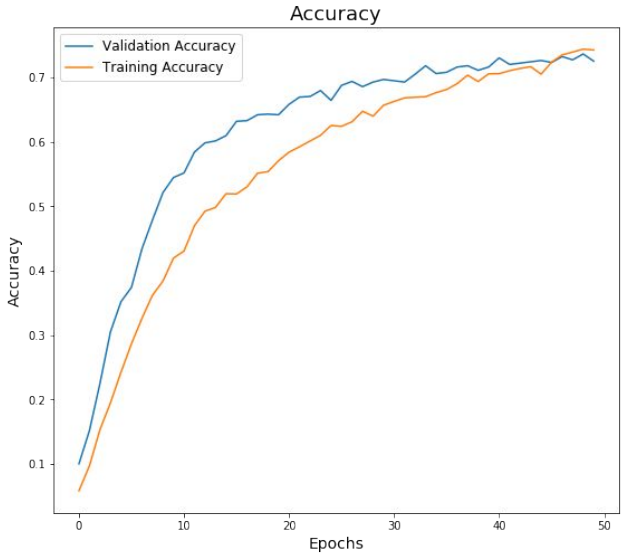


Example spectrogram for “Girl” audio sample

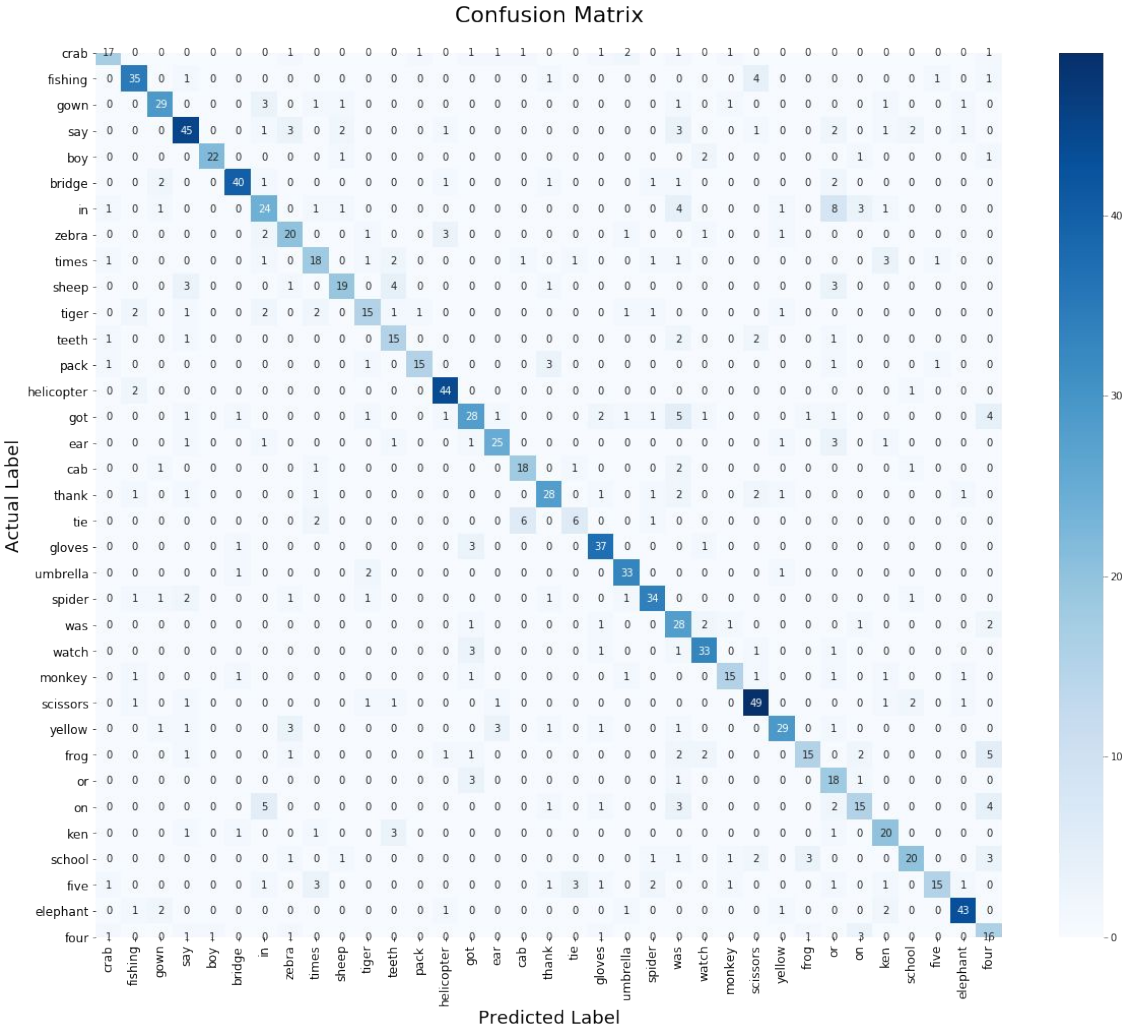
Conclusion –
Lower accuracy, minimised overfitting

Lower accuracy not a hindrance to usefulness of the app

- Most models did not generalise well, tending to overfit the training data. Likely because of the nature of speech impairments.
- Final model produced the most balanced results correctly identifying the spoken keyword for 70% of the “unseen” audio samples.



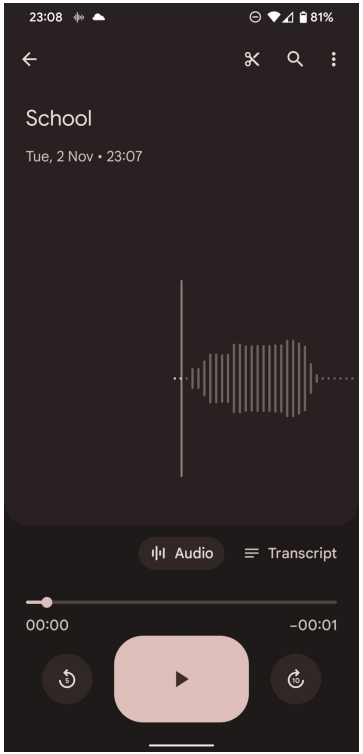
“Final” model accuracy during training


Confusion Matrix
Actual Label
Predicted Label
crab
fishing
gown
say
boy
bridge
in
zebra
times
sheep
tiger
teeth
pack
helicopter
got
ear
cab
thank
tie
gloves
umbrella
spider
was
watch
monkey
scissors
yellow
frog
or
on
ken
school
five
elephant
four
0
10
20
30
40

Conclusion – Potential next steps

Recommendations –

- Used as part of mobile app similar to Google Recorder.
- Parents capture and crop samples of speech and receive suggested words in response.
- Display up to 3 possible words in order of likelihood.
- Final model accuracy enough for soft launch of the app to a limited test group of parents.



Google Recorder App

Conclusion – Potential next steps

Future work could include –

- Alternative model architectures.
- Extend the models to accept audio samples longer than 1 second.
- Source additional data.
- Use data augmentation when training the model.

Thank you.

Any questions?

Appendix –
Model results highlights

Model Iteration	Model Details	Training Accuracy	Validation Accuracy
1	Baseline model with single layer	81%	67%
2	Baseline model varying learning rate and batch size	94%	65%
3	Deeper network model	96%	72%
4	Deeper network model but adding dropout layers	85%	70%
5	CNN	83%	70%
6	CNN #2 (descending number of neurons over the first 3 layers)	93%	70%
7	CNN #3 (change dropout layer from 0.3 to 0.5)	94%	69%
8	CNN #4 (Neuron tweaks etc.)	90%	70%
9	CNN #5 (changes to epochs and batch size.) - FINAL	89%	72%

Appendix –
Model summary and classification report

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 42, 11, 32)        320
_________________________________________________________________
batch_normalization_8 (Batch (None, 42, 11, 32)        128
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 21, 6, 32)         0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 18, 3, 32)         16416
_________________________________________________________________
batch_normalization_9 (Batch (None, 18, 3, 32)         128
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 9, 2, 32)          0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 8, 1, 64)          8256
_________________________________________________________________
batch_normalization_10 (Batc (None, 8, 1, 64)          256
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 4, 1, 64)          0
_________________________________________________________________
flatten_6 (Flatten)          (None, 256)               0
_________________________________________________________________
dense_13 (Dense)             (None, 32)                8224
_________________________________________________________________
dropout_1 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_14 (Dense)             (None, 35)                1155
=================================================================
Total params: 34,883
Trainable params: 34,627
Non-trainable params: 256
_________________________________________________________________
```

	precision	recall	f1-score	support
crab	0.74	0.61	0.67	28
fishing	0.80	0.81	0.80	43
gown	0.78	0.76	0.77	38
say	0.74	0.73	0.73	62
boy	0.96	0.81	0.88	27
bridge	0.89	0.82	0.85	49
in	0.59	0.53	0.56	45
zebra	0.62	0.69	0.66	29
times	0.60	0.58	0.59	31
sheep	0.76	0.61	0.68	31
tiger	0.65	0.56	0.60	27
teeth	0.56	0.68	0.61	22
pack	0.88	0.68	0.77	22
helicopter	0.85	0.94	0.89	47
got	0.67	0.57	0.62	49
ear	0.81	0.74	0.77	34
cab	0.69	0.75	0.72	24
thank	0.74	0.72	0.73	39
tie	0.55	0.40	0.46	15
gloves	0.79	0.88	0.83	42
umbrella	0.80	0.89	0.85	37
spider	0.79	0.79	0.79	43
was	0.47	0.78	0.59	36
watch	0.79	0.82	0.80	40
monkey	0.75	0.65	0.70	23
scissors	0.79	0.84	0.82	58
yellow	0.81	0.71	0.75	41
frog	0.75	0.50	0.60	30
or	0.39	0.78	0.52	23
on	0.58	0.48	0.53	31
ken	0.62	0.74	0.68	27
school	0.74	0.61	0.67	33
five	0.83	0.48	0.61	31
elephant	0.88	0.84	0.86	51
four	0.43	0.64	0.52	25
accuracy			0.72	1233
macro avg	0.72	0.70	0.70	1233
weighted avg	0.73	0.72	0.72	1233

Appendix –
Making a prediction

```
# Run inference on the unseen audio file
mfccs = librosa.feature.mfcc(audio_sample,
                             sr,
                             n_mfcc=13,
                             n_fft=2048,
                             hop_length=512)
mfccs = mfccs.T
mfccs = mfccs[np.newaxis, ..., np.newaxis]

prediction = us_cnn_model.predict(mfccs)
predicted_index = np.argmax(prediction)

predicted_keyword = us_keywords[predicted_index]
print('Martha says...',predicted_keyword, '!')
```

```
Martha says... umbrella !
```