UNIVERSITY NAME

DOCTORAL THESIS

Thesis Title

Author:
John SMITH

Supervisor:
Dr. James SMITH

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy

in the

Research Group Name
Department or School Name

June 24, 2018

Declaration of Authorship

I, John SMITH, declare that this thesis titled, "Thesis Title" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UNIVERSITY NAME

Abstract

Faculty Name
Department or School Name

Doctor of Philosophy

Thesis Title

by John SMITH

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

Acknowledgements

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

Contents

Declaration of Authorship iii

Abstract vii

Acknowledgements ix

1 Introduction 1
1.1 Motivation . . . 1
1.2 Background . . . 1

2 Preprocessing 5
2.1 Audio Format . . . 5
2.2 Mel Frequency Cepstral Coefficients . . . 5
2.2.1 Frame signal into short frames . . . 6
2.2.2 Pre-emphasis . . . 6
2.2.3 Periodogram . . . 6
2.2.4 Mel Filterbank . . . 7
2.2.5 Log and DCT . . . 7
2.2.6 Output . . . 8
2.3 Subtitles . . . 9
2.3.1 Pseudocode . . . 9

3 Learning 11
3.1 Learner Architecture . . . 11
3.1.1 Activation Functions . . . 11
3.1.2 Batch Normalisation . . . 12
3.1.3 Dropout . . . 12
3.2 Tuning . . . 12
3.3 Training Set Size . . . 13

4 Synchronising 15
4.1 Array Matching . . . 15

A Frequently Asked Questions 17
A.1 How do I change the colors of links? . . . 17

Bibliography 19

List of Figures

1.1 Log loss plot when matching full audio to full subtitles 2

2.1 Steps of MFCC Feature Extraction [7] 6
2.2 Filtering with Mel Filterbanks [4] 7
2.3 The MFCCs extracted from a Game of Thrones episode. Bluish shades indicate subtitles are absent, pinkish shades indicate they are present. Feature window index is used as a proxy for time. 8

3.1 Convolutions in 1d [3] 11
3.3 Validated on a separate episode of The Walking Dead 13
3.2 Model Architecture 14

List of Tables

List of Abbreviations

LAH	**L**ist **A**bbreviations **H**ere
WSF	**W**hat (it) **S**tands **F**or

Physical Constants

Speed of Light $c_0 = 2.997\,924\,58 \times 10^8\,\mathrm{m\,s^{-1}}$ (exact)

List of Symbols

a	distance	m
P	power	W ($\mathrm{J\,s^{-1}}$)
ω	angular frequency	rad

For/Dedicated to/To my…

Chapter 1

Introduction

1.1 Motivation

Deafness is considered to be the most prevalent impairment worldwide, affecting at least 278 million people [12]. However, current provisions for those who may struggle to fully hear and comprehend human speech in commercial settings is distinctly lacking. Cinemas routinely show only 2 or 3 showings a week with subtitles in the UK [11], restricting the range of films and times that can be enjoyed, adding to the sense of exclusion experienced by disabled people. There are technologies available to provide captions for individual users[5], but these rely on each institution having purchased these, and are far from universally available. Even those that do have them only provide them for a small selection of movies, 6/27 at one cinema chain in New Zealand [6]. Therefore, to ensure availability of these it is desirable to find methods that could be applied on the user-side so that subtitles can be made available in whichever context the user desires. Time stamped subtitle files are freely available as SubRip files (.srt file extension) that contain a series of text entries and associated start and stop times, but these rely on knowing the exact start time of a film so that the 2 files are synchronised. This is extremely difficult to achieve in a live setting where the exact commencement of the feature film is unclear, and the introduction of even a small time shift can make subtitle viewing an unwatchable experience. Hence, the demand arises to automatically synchronise a subtitle file using real time audio signals to locate the current position in a video.

1.2 Background

This problem was tackled in a laboratory setting[9], so that a user can synchronise audio and subtitle tracks for future viewing on a personal device. This method extracted and sampled the audio at 16kHz, extracted the features using the common automatic speech recognition (ASR) technique of splitting the signal into 20-40ms frames and extracting the Mel Frequency Cepstral Coefficients for each frame and using these as the predictive features. The subtitle track is then used to create a binary array indicating whether subtitles (and therefore human speech) occur in each corresponding frame to those used for MFCC, providing the target variable. A recurrent neural network (LSTM) was initially trained, a logical approach given the chronological nature of the data, but due to the fact that several frames must be used as input, the accuracy was limited to the duration of the number of input frames: if 10*0.05s samples are required, the accuracy is limited to 0.5s which is insufficient in practise. Therefore a Convolutional Neural Network (CNN) was trained, taking as input just 1 sample, with several one-dimensional convolutional layers followed by several dense layers to output an array containing the probabilities that human

speech (subtitles really) is present in each frame. Minimisation of the log loss function was then used to align the subtitles array with the probability array, the delay calculated and applied to the subtitle track; this loss function was chosen in order to maximise the accuracy by penalising false classifications. Using this process, after training the NN and hyperparameter tuning, the log-loss minimum is clearly defined:

Using dynamic programming to take account of the fact that every step must be evaluated across the signal, this process took 45 seconds for a tv show and 2 minutes for a film, the whole process using CPU's. This approach is promising and formed the basis upon which this study was formed. There were key differences in the aim that require further study in order to implement this in a live setting. Since this was performed on the recorded data rather than a live recording, the signals are pure and contain no significant noise, and so a model that could handle such settings needed to be identified. Furthermore, there would be more uncertainty in the start times of the film which would need to be accounted for – search can commence from the beginning and match the 2 full arrays since the aim is to match the full sound track with the subtitle track (with less concern for excessive processing time), whereas in real time, access to the sound track would occur incrementally as new audio is recorded. The Sabater study was informal, with the results released as a blog post and so lacked rigorous (or at least reporting of the) approach and testing environment, with no account of the relative complexities and possible different approaches that could be taken.

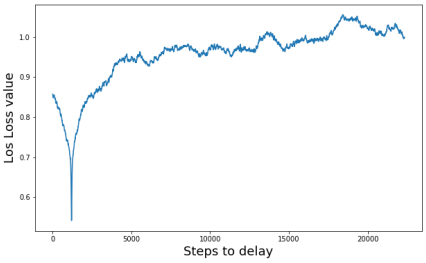

FIGURE 1.1: Log loss plot when matching full audio to full subtitles

A similar approach was adapted in [8] in order to align a draft script with a live presentation which loosely followed this but often presented changes in the script and its chronology (insertions, substitutions and deletions). To do so, 12 order MFCC's were used again, and "normalised energy coefficient augmented by the corresponding delta and delta-delta coefficients information as features"(p2), but an ASR engine formed of "continuous HMMs with context dependent acoustic units, where each unit is modelled with a 16 component Gaussian Mixture Model with diagonal covariance matrices" was used. The HMM's enable sequence data to be modelled and are not constrained in the same way as RNN's that must take multiple sequential samples in order to predict a sample, thereby reducing the accuracy achievable. This was a more ambitious objective in that the audio had not been prerecorded and so to compare the audio to the subtitles, the MFCC's were compared to a phoneme network in order to identify utterances and therefore alignment of the tracks. In addition, this system substituted automatically generated subtitles when no script was available due to insertions. This model achieved an accuracy of up to 100% on newscasts that adhered strictly to the script and naturally dropped to 30% when the programme became unscripted. In addition, due to the preprocessed nature of the scripts the latency was deemed to be negligible. Another approach that has been implemented [2] In which audio is generated from text by Text To Speech (TTS) systems and this is used to synchronise the text and audio. This is a very promising idea and in the context of this paper's problems would eliminate the requirement to have the film audio for training, and only the srt. Therefore the aim

will be to extend the study to evaluate this method in the noisy setting.

Chapter 2

Preprocessing

2.1 Audio Format

It was deemed most appropriate to use the compressed MP3 extracted from the MPEG-4 video file for a number of reasons. Although using the uncompressed wav form would present a more faithful reproduction of the audio, and possibly provide more data to train on, the additional complexity far outweighed any possible gains. The MFCC extraction is very sensitive to the [something] and testing on the wav file proved prohibitively long. The space requirements were far larger and approached the file size of the compressed video and audio combined. Furthermore, since the aim is to identify human speech, the lossy mpeg compression regime is explicitly designed to mimic the constraints on human hearing using psychoacoustic modelling, whereby imperceptible frequency ranges, signals with subthreshold amplitudes and frequencies masked by other more dominant frequencies are excluded (Perceptual Coding of Digital Audio), among many other filtering techniques. This process is well-suited to feature extraction for ASR as human speech is naturally adapted to be comprehended by human ears, and vice versa.

The bandwidth of human speech is 80-260hz (2 references from wikipedia) (Titze, I.R. (1994). Principles of Voice Production, Prentice Hall (currently published by NCVS.org) (pp. 188), ISBN 978-0-13-717893-3. Baken, R. J. (1987). Clinical Measurement of Speech and Voice. London: Taylor and Francis Ltd. (pp. 177), ISBN 1-5659-3869-0.) The sample rate was chosen to be 16kHz in order to avoid aliasing of the signal due to sample rate being below the Nyquist rate, although the nyquist rate is actually 520hz... telephones sample at 8kHz, so above this is better. (http://www.dspguide.com/ch22/3.htm). The nyquist criterion defines the sample rate necessary to avoid aliasing, where the sample rate frequency is insufficient to capture the full details of the waveform, to be: fs > 2B (sampling frequency greater than twice the bandwidth).

2.2 Mel Frequency Cepstral Coefficients

In order to extract the features most relevant to speech, MFCC's were chosen due to their robustness to noise compared to another popular feature extraction technique, Linear Predictive Cepstral Coefficient [1]. Others have found that it performs less well without some preprocessing but is relatively fast to compute [10]. MFCC's are considered the baseline, reliable method and there is a python library (python_speech_features) available for computing these, and so this was the method chosen. This technique also applies pschoacoustic modelling to the signal by applying a series of transformations.

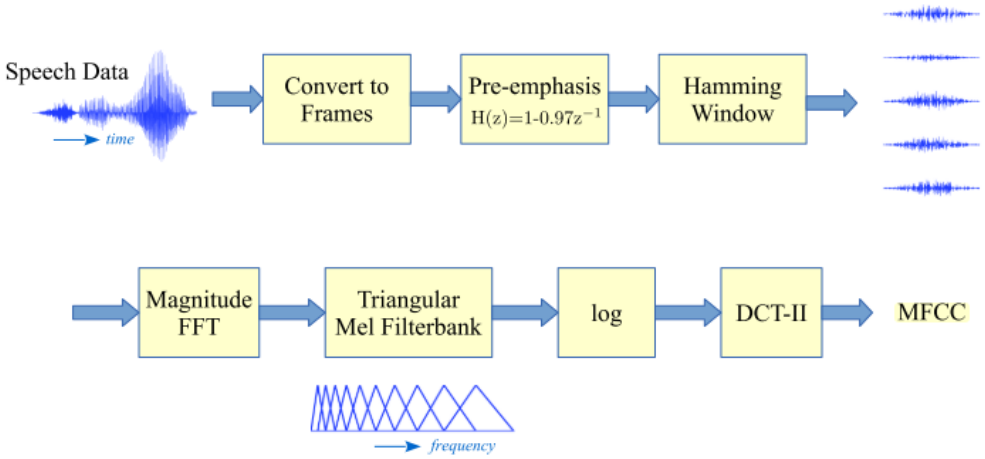

FIGURE 2.1: Steps of MFCC Feature Extraction [7]

2.2.1 Frame signal into short frames

] The signal is framed into overlapping frames in order for the frequency analysis to be performed. This is assumes that audio spectra are effectively constant over short time frames (typically 20-30ms). The windows are overlapping so that continual frequency features are captured...

2.2.2 Pre-emphasis

There is a preemphasis applied, where for each value in the frequency domain a high-pass filter is applied:

$$s_2(n) = s(n) - a * s(n-1) \tag{2.1}$$

where n is the sample number, s is the signal in the frequency domain and s2 is the transformed signal. This has 3 effects http://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html: (1) balance the frequency spectrum since high frequencies usually have smaller magnitudes compared to lower frequencies, (2) avoid numerical problems during the Fourier transform operation and (3) may also improve the Signal-to-Noise Ratio (SNR).

2.2.3 Periodogram

For each frame the periodogram is calculated, which encapsulates the relative energy in different frequency bands of the signal. This is done using the Discrete Fourier Transform (discrete due to the digital nature of the signal):

$$S_i(k) = \sum_{n=1}^{N} s_i(n)h(n)e^{-j2\pi kn/N} \text{for } 1 \leq k \leq K \tag{2.2}$$

where $h(n)$ is an n sample long analysis window (e.g. hamming window), and K is the length of the FFT. For a 16kHz signal in 25ms frames, N equates to 400 samples. The periodogram-based power spectral estimate for the speech frame is then given

by:

$$P_i(k) = \frac{1}{N}|S_k|2 \tag{2.3}$$

i.e. we take the absolute value of the complex fourier transform of each coeffecient, and square the result.

2.2.4 Mel Filterbank

This step applies a series of filters (26 commonly) to the periodogram, composed of vectors of length 26 with nonzero values at different sections along the vector. The filters are nonuniform, increasing in width with frequency. This emulates the nature of human hearing to have lower sensitivity to changes in frequency as frequency increases. This is based on the Mel scale which models this nonlinearity and describes exactly how to space the bins. The spectrum $P(f)$ is warped along its frequency axis f (in hertz) into the mel-frequency axis as $P(M)$, where M is the mel-frequency, using:

$$M(f) = 2595 \log((1 + f/700) \tag{2.4}$$

A non-linear frequency scale transformation is applied based on the observation that human hearing does not perceive frequencies linearly but, like amplitude, on a logarithmic scale:

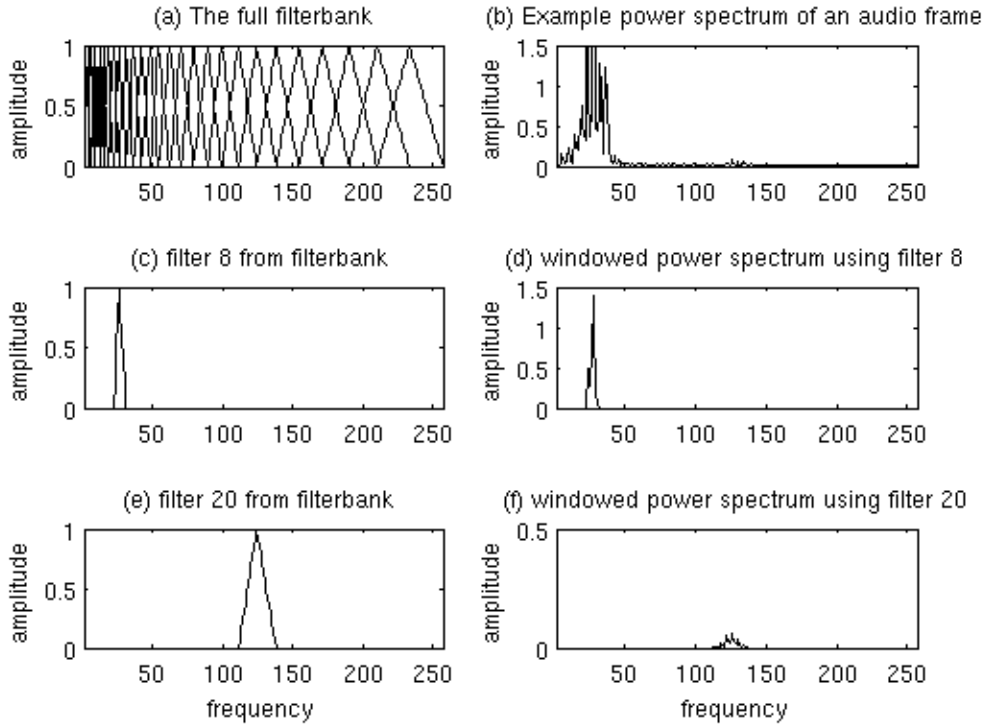

FIGURE 2.2: Filtering with Mel Filterbanks [4]

2.2.5 Log and DCT

The log of the energies in each filterbank is taken, and a Discrete Cosine Transformation (DCT) is applied to extract 13 new coefficients of this new spectrum. The

DCT has several benefits over another transformation like the Fourier transform in that it does not interpret components as infinite waves, and so is better suited where signals are more likely to be interrupted, as the framing process does. Furthermore only real numbers are output which is more readily dealt with by learners such as CNN's.
Generally the largest 13 coefficients are taken that represent the most important information. https://dsp.stackexchange.com/questions/31/how-do-i-interpret-the-dct-step-in-the-mfcc-extraction-process probably need better source...

2.2.6 Output

Hence MFCC's function as a feature selection technique that reduces the 400 samples each with 256 values (8 bits) of amplitude representation to this lower feature space specifically designed to emulate how humans hear, which are themselves well placed to interpret speech signals. However, Figure 2.3 indicates that the relation between MFCC's and speech presence is not immediately obvious. MFCC2 appears to have the most distinctive correlation with the coefficients doubling in size from ∼30 to ∼60 in the absence of speech, but other features have similar values, and there is significant intraclass variation.

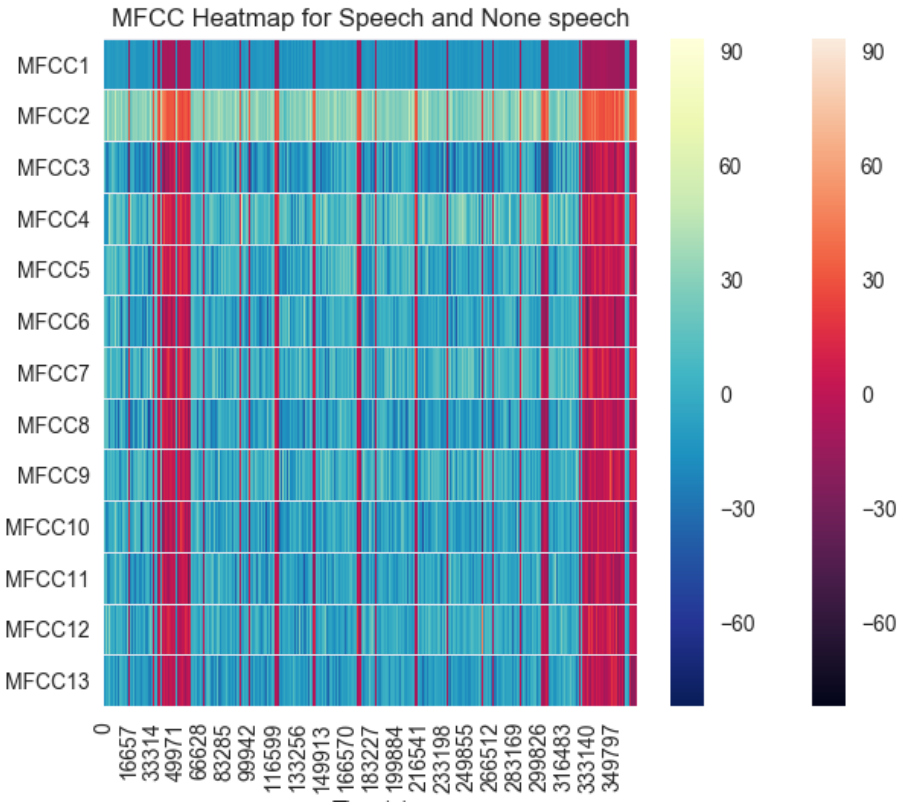

FIGURE 2.3: The MFCCs extracted from a Game of Thrones episode. Bluish shades indicate subtitles are absent, pinkish shades indicate they are present. Feature window index is used as a proxy for time.

2.3 Subtitles

The MFCC's provide the descriptive variables upon which to make predictions, and the objective variable is defined to be the presence of subtitles. This setup was chosen to enable the model to provide based solely on the srt file: since this subtitle synchroniser is aimed principally at cinemagoers, it is assumed that the original audio/video is unavailable, but it is assumed that an srt file is accessible. These are readily available on the internet for most films and are small (essentially text) files and so networking and device requirements would be minimal: they contain a series of entries made up of a start time, stop time, and a string to be presented on the screen at these times. In order to generate the subtitle array:

- Times are converted into seconds from start.
- The time at which the last subtitle disappears from the screen is used to determine the length in seconds that subtitles are present. Subtitles need only be synchronised when there are actually subtitles present.
- The number of frames required is determined by dividing the length in seconds by the frame step (10ms) (the frame length is 25ms, but these are overlapping).
- An empty array pb_array is generated with columns for an index, start time and binary subtitle value to indicate presence of subtitles. It is of equal length to the number of frames required.
- The 2 arrays, pb_array and subs_array, are parsed concurrently, with the frame times of pb_array checked to see if they lie within a times entry.

2.3.1 Pseudocode

Algorithm 1 pb_array_fill

```
1:  procedure MyProcedure
2:      i ← 0
3:      j ← 0
4:      m ← pb_array_length
5:      n ← subs_array_length
6:      while True do
7:          if i > m  then break
8:          if j > n  then break
9:          if pb_array[i] start time ≥ subs[j] start time then
10:             if pb_array[i] end time < subs[j] end time then
11:                 pb_array[i] ← 1
12:                 i ← i + 1
13:         else
14:             j ← j + 1
15:
16:
```

This outputs a vector with length equal to number of MFCC frames, filled with binary values indicating presence of subtitles. This provides the objective feature to predict.

Chapter 3

Learning

3.1 Learner Architecture

The network architecture implemented by Sabater was used as a starting point. This used convolutions in 1d with kernel size 3, where the filter is initially applied over the MFCC features. This process is identical to the more traditional 2d convolutions in computer vision, except the kernel is a vector rather than a square/array. 3 filters were used.

This architecture associates adjacent MFCC's with each other in a hierarchical fashion. It could be hypothesised that during speech the changes in pitch/frequency occur in a consecutive fashion, with sudden changes from different parts of the spectrum unlikely, and so this information can be compressed/summarised... No padding was used, and although this means central coefficients are over represented in relation to those at the edges, it could also be hypothesised that these occur less often and, being at the edges of the frequency spectrum, are less critical to comprehension, thus features can be reduced further. As in 2d convolutions, the exact nature of the filters are left to be learned during the optimisation (back propogation?) process. 12 filters in 2 convolution layers gives 24 parameters to be learned. The 1D convolutions output tensors of shape (V-(F-1),NF), V is the feature vector size, NF is number of filters. To produce the correct shape for the dense layer, the tensors are flattened out to feature vectors again. In the original implementation by Sabater, he applies the ReLu activation function at this point, before applying batch normalisation and dropout. These layers are then repeated until the output neuron with a sigmoid activation function.

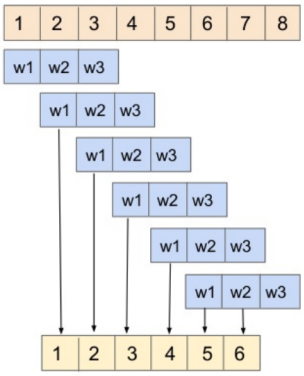

FIGURE 3.1: Convolutions in 1d [3]

3.1.1 Activation Functions

The activation functions provide the mathematical function that emulates the non-linear, biphasal response in the brain. For all the internal layers the ReLU function was used as this is less computationally expensive than the sigmoid where all neurons are guaranteed to fire at all times, and the use of a dropout layer removes these dying neurons to give a sparser output. The output activation function was a sigmoid, which is well suited for classification and ensures that all outputs from the final dense layer will be evaluated.

3.1.2 Batch Normalisation

I didn't normalise the features directly since this relies on having the full dataset and when deployed it would not be possible to do this since we are only granted access to the dataset incrementally. In order to learn, it is most effective when the training and test data have the same distribution. However, at each layer the outputs retain this relation only weakly due to the non-linearity of the activation functions, a process known as "internal covariance shift", and so at each layer the new distribution must be relearnt. In addition, due to the flattening of the sigmoid function at larger values, the optimisation process slows as the gradient varies less over fixed learning rates. This can be compensated for by renormalising after every layer, but In order to normalise, the mean and variance is computed across the whole training set, but this is expensive and "not differentiable everywhere". Instead, batchnorm stochastically computes the mean and variance over much smaller overlapping batches. This can be seen as having a regularising effect as the outputs through each layer are now dependent on what other examples were present in that batch via the mean and variation with which it is normalised rather than being processed independently. Since this process can be done several times, a single example can be related to many other examples. At test time, the learned features of the gaussian are fixed and applied to the data to give deterministic responses. In order to compensate for this, the outputs can be normalised to a gaussian with zero mean and unit variance across each of the dimensions, output xhat. In order to ensure that the activation functions are thus not constrained to this gaussian distribution, a further transformation is made: $y = \gamma\hat{x} + \beta$, and incorporate these new parameters into the learning process to be identified via backpropogation, hence different distributions can be inferred directly.

3.1.3 Dropout

Dropout is one method of limiting overtraining and the interdependence of neurons within layers. Nodes are defined to be active in a stochastic function as determined by a probability parameter. Hence when training, every time a sample is propagated through the network a random subset, in proportion to the probability parameter, of the neurons will not produce any output. Hence, later layers cannot depend on the presence of all inputs and reduces the tendency of neurons to co-adapt, activating in similar ways to other neurons and hence not providing additional information. In this way, a much larger set of network configurations is randomly sampled, and hence can be viewed as operating as an ensemble method for 1 hidden layer, different for more than 1.. All architectures share weights – whenever we use a hidden unit its got the same weights as it has in other architectures. Extreme form of bagging – very large number of architectures trained on just 1 sample. Sharing of weights means that every model is very strongly regularised – the weights learned from 1 architecture are stored before the architecture is sampled again. A dropout value of 0.2 was applied.

3.2 Tuning

Once the general neural network architecture was decided, there were still a range of hyperparameters to tune. In order to evaluate the effect of different optimisers, batch size and number of epochs, the hyperas package was used. This enables models with different parameters to be evaluated and the optimal parameters identified

using grid search cross validation and an optimisation algorithm, Tree of Parzen Estimators (TPE) was used here. http://hyperopt.github.io/hyperopt/. The range of parameters tested was: batch_size=[32, 64, 128], epochs=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 50, 100], optimizer=['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax', 'Nadam'], In so doing, the best performing parameters were identified to be: optimiser = Nadam, batch size = 32. [However, due to the random initialisation process this produced different results every time...]

3.3 Training Set Size

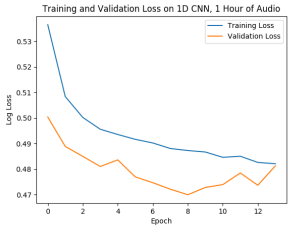

FIGURE 3.3: Validated on a separate episode of The Walking Dead

In order to identify the size of training set required, the optimised learner was run on a separate validation set and the loss evaluated with epoch to see how training accuracy would improve with more data. What was thought to be a small dataset consisting of 1 hour of audio from The Walking Dead in fact was enough for the log-loss plot to indicate that the best results achievable had already been done so, and no more data would be required to replicate the results on data with a similar distribution. Interestingly the validation loss is less than the training loss.. The validation results gave [0.81, 0.50] [accuracy, loss], which emulated closely the training results. This indicated that an hour of audio should be sufficient, but to ensure success, a full feature film was used for training.

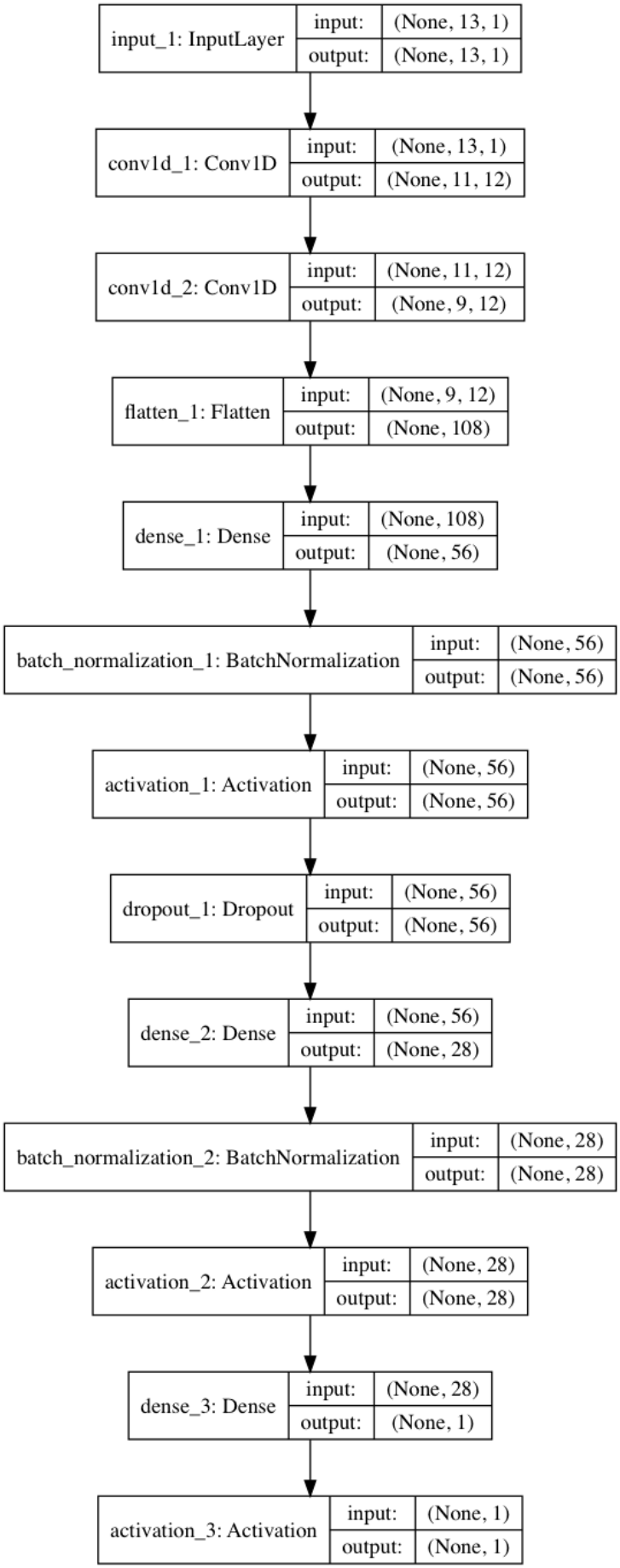

FIGURE 3.2: Model Architecture

Chapter 4

Synchronising

4.1 Array Matching

With a sufficiently accurate subtitle predictor, the next challenge was to match the predictions to the true array. Sabater initially took the whole prediction array and computed the log loss over different positions of the subtitles array in order to find the minimum which would equate to the best match. However, this approach assumes full access to both arrays: in order to match subtitles in real time, the probability array would be accessed incrementally as new audio is recorded and features extracted. Initially, it was attempted to record audio of fixed duration in order to match (size 50 initially, equivalent to 500ms), whilst noting the time for the algorithm to run, and applying a delay to the subtitles once the match had been found. The length of the audio was to be optimised, and it was assumed that live audio data would commence acquisition after the beginning of the film so that the algorithm does not attempt to match audio features generated before the film has actually begun, perhaps due to background noise or advertisments. To test this independently of the subtitles predictor, the array match was given a sample of the subtitles array N steps after the beginning of the full subtitles array in order to evaluate the behaviour. Interestingly, this approach consistently returned index 0 as the best position. On investigation, it was revealed that there were no subtitles at the beginning and so the sample array was made up of entirely zeros, and so the function returned this as the optimal spot: there were in fact multiple optimum positions (generally all at the start) where no subtitles were present and it simply returned the first position which fulfilled the minimisation condition. Whilst this seemed like an obstacle initially, it was realised that the duration of time in which no subtitles occurs can be considered a distinctive feature in itself, especially when combined with the assumption that synchronisation has commenced not long after the beginning of the film: it is unlikely that viewing would commence after the majority of the film has already finished. In addition, new data can be continuously acquired by simply appending new predictions based on new features extracted from the incoming audio. This foreknowledge, a product of the specific nature of this problem, can be utilised to shrink the search space significantly by defining the first match to be obtimal, whilst continuously expanding the number of examples to match against the base truth. In practice this is implemented by recording data and extracting the features, and if no speech is found, continue to acquire new data until a subtitle is predicted. This subtitle would correspond to the first subtitle present in the film and would provide the distinctive feature to search for when matching. This could be done in 2 ways, either the array generated, of a long sequence of zeros followed by a 1, is matched using the log loss, or in an even greater simplification, the subtitles are simply commenced as soon as this is detected. Alternatively, if speech commences from the beginning, the recorded features should provide enough variation that a match can be found

when searching from the beginning. – the longer the duration of time in which no subtitles occurs, the longer the sequence of zeros

Appendix A

Frequently Asked Questions

A.1 How do I change the colors of links?

The color of links can be changed to your liking using:

`\hypersetup{urlcolor=red}`, or
`\hypersetup{citecolor=green}`, or
`\hypersetup{allcolor=blue}`.

If you want to completely hide the links, you can use:

`\hypersetup{allcolors=.}`, or even better:
`\hypersetup{hidelinks}`.

If you want to have obvious links in the PDF but not the printed text, use:

`\hypersetup{colorlinks=false}`.

Bibliography

[1] Utpal Bhattacharjee. "A comparative study of LPCC and MFCC features for the recognition of Assamese phonemes". In: *International Journal of Engineering Research & Technology* 2.3 (2013), pp. 1–6. ISSN: 2153-1234. DOI: 10.4236/jis.2012.34041. URL: https://pdfs.semanticscholar.org/7c8f/8a9d5ba85788b569bc04ca9f07d6ce689pdf.

[2] N Campbell. "Autolabelling Japanese ToBI". In: *ICSLP 96. Fourth International Congress on Conference on Language Processing Proceedings* 4 (1996), pp. 2399 – 2402. DOI: 10.1109/ICSLP.1996.607292. URL: http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=607292.

[3] Universitat Politècnica de Catalunya. *Recurrent Neural Networks II (D2L3 Deep Learning for Speech and Langu...* 2017. URL: https://www.slideshare.net/xavigiro/recurrent-neural-networks-2-d2l3-deep-learning-for-speech-and-language-upc-2017.

[4] *Crypto*. URL: http://www.practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/.

[5] *Dolby CaptiView*. URL: https://www.dolby.com/us/en/professional/cinema/products/captiview.html.

[6] *HOYTS Cinemas*. URL: https://www.hoyts.co.nz/experiences/cc.

[7] D. S. Pavan Kumar. "Feature Normalisation for Robust Speech Recognition". In: *CoRR* abs/1507.04019 (2015). arXiv: 1507.04019. URL: http://arxiv.org/abs/1507.04019.

[8] Alfonso Ortega et al. "Real-time live broadcast news subtitling system for Spanish". In: *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* (2009), pp. 2095–2098. ISSN: 19909772. URL: http://vivolab.es/demos/subtitle/subtitling.pdf.

[9] Alberto Sabater. *Automatic Subtitle Synchronization – Machine Learnings*. 2017. URL: https://machinelearnings.co/automatic-subtitle-synchronization-e188a9275617.

[10] Urmila Shrawankar and V M Thakare. "Techniques for Feature Extraction In Speech Recognition System : A Comparative Study". In: *International Journal Of Computer Applications In Engineering, Technology and Sciences (IJCAETS),ISSN 0974-3596* (2013), pp. 412–418. arXiv: 1305.1145. URL: http://arxiv.org/abs/1305.1145.

[11] *Subtitled Cinema Showings in London*. URL: http://yourlocalcinema.com/london.central.html.

[12] Debara L. Tucci, Michael H. Merson, and Blake S. Wilson. "A summary of the literature on global hearing impairment: Current status and priorities for action". In: *Otology and Neurotology* (2010). ISSN: 15317129. DOI: 10.1097/MAO.0b013e3181c0eaec.