Speaker Identification with Neural Networks

Sparsh Agarwal (CS Major, Junior) [1] **Wayne Chew (CS Major, Senior)** [1]
Keshav Sharma (CS Major, Sophomore) [1]
Newton Wolfe (CS Major, Sophomore) [1]

Abstract

Voice is a highly important method of human interaction, and a major part of peoples' identities. Despite this, few implementations of speaker recognition have been performed at large scale. We train a neural network based on narration from audio books as input, to learn voiceprints and determine the speaker of an arbitrary utterance. Experiments were been performed in various frameworks and focus was on ability to categorize using minimal computation. Using only a quarter second input, high accuracy ($> 75\%$) recognition was possible among many users, with even higher accuracies with fewer users.

1. Introduction

It has long been science fiction for every device to be able to interact with humans using voice control, and with the advent of cloud computing and large-scale machine learning, voice based interfaces are increasingly able to understand and act upon human speech. With devices like Amazon Echo and Google Home available to the consumer market, the use of vocal interfaces to computers is becoming common. Many households have multiple individuals, and it is desirable for a voice-only device like Amazon Echo to provide personal context. This requires that the device is able to seamlessly distinguish between their voices. Furthermore, as voice-based interfaces become an integral part of consumers' lives and might have private data, it is imperative that it reject instructions from outsiders. This requires speaker verification and speaker recognition both to ensure proper categorization of authenticated users and rejection of others. For this paper, we draw an important but subtle distinction between speaker verification and speaker recognition. Speaker verification, confirms the identity that one claims to have, while speaker recognition is when the computer is expected to recognize and label an unknown person. Speaker verification is a 1:1 problem whereas, speaker recognition is a 1:N problem. However, many of these interfaces are not specific to certain speakers, even in situations where that would be desirable. It is important for the device to be able to understand which of the multiple users is speaking to it and respond in context, as a human would.

Voiceprints are like thumbprints in that they are unique for each individual. As such, they can be used to identify an individual. This is how our brain interprets and distinguishes between different people. Perhaps the most familiar method of recognizing speakers is by

[1]University of Wisconsin-Madison, Madison, Wisconsin, USA. Correspondence to: Newton Wolfe <nwolfe@cs.wisc.edu>.

CS 638 Class Project, Spring 2017.

recognizing the pattern in which they speak. However, we will not be using this pattern to classify; instead, we will be using another information rich feature of human voice, which is the amount of energy a person exerts in various ranges of frequencies while speaking. (Geitgey, 2016) This is just one such feature and there exist many such variabilities in voice which can be used, but our experiments found this particular category to work well.

2. Problem Definition and Algorithms

Our problem is to determine the speaker in a recording, based on previous recordings of the speaker. To do this, we collate recordings from several different speakers, then process these recordings via the Fast Fourier Transform into discrete samples which represented the energy intensities at different frequencies across a given sampling period. Audio processing was done via ffmpeg (Ffmpeg, 2010) and the librosa (McFee et al., 2017) audio library. This approach was inspired by the approach of Geitgey (2016), which was originally to distinguish between different sounds from the same speaker.

These samples were then uses as inputs to our neural networks, with one input node per frequency, and one output node per speaker. This structure was chosen because of the difficulty of encoding multiple feature relationships into a neural network; while there is a natural ordering of the frequencies that may offer additional information for a neural network, our representation was much simpler. A one-hot output encoding was used due to our experience using such an encoding in labs 2 and 3, and because of the lack of a natural relationship among the speakers.

We chose not to use time-series data for multiple reasons. First, it restricts our neural network to learning inherent characteristics of a speakers' voice, like the overtones produced in the speech pathway, rather than other characteristics of speech like speech rate that may vary depending upon the situation. Furthermore, it allows our network to be used in more applications; fast recognition of a speaker is desirable in many settings. Finally, in early tests of our neural networks with 6 speakers we were able to achieve 98% classification accuracy of different speakers, suggesting that time series data is not necessary to produce high-accuracy results, and simpler architectures allow for much less training data to be used.

Our primary neural network's structure is depicted in Figure 1. We used 2 hidden layers that were both fully connected; while convolutional neural networks like the ones we implemented in Lab 3 are useful in data with 2 dimensional structure; they are of limited use with single-dimensional data, like ours. As such, we use fully connected layers, and use 2 to trade off between accuracy and computational performance. We used ReLU for all activation functions, and the cross-entropy error function for classification.

Training of our models was done via the ADAM optimizer with the parameters described in the paper when it debuted. (Kingma & Ba, 2014) Additionally, early stopping was used to prevent overfitting.

3. Methods

3.1. Dataset

For our recordings, we downloaded high quality recordings of different speakers doing audiobook narration from LibriVox, via the LibriSpeech repository. (Panayotov et al., 2015) Audiobooks were chosen as our preferred audio source because they offer consistent but high quality recordings; this allows for minimal preprocessing of the audio while avoiding our neural networks learning to distinguish non-speaker features like background noise level. We tried to extract various different features

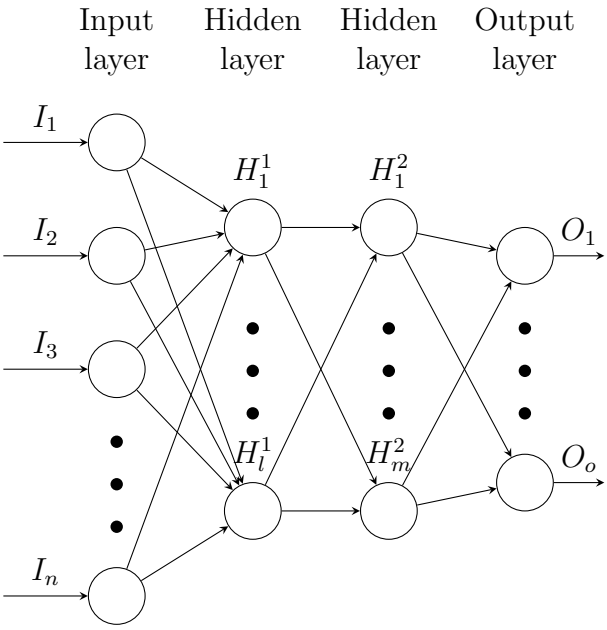

Figure 1. A fully connected neural network with two hidden layers; we used a neural network with this architecture.

from the audio book and in differnt formats to determine which ones fits nest. Based on our understanding that the pattern would be in energy vs frequency, we focused on various forms of energy in a voice form and the formats in which it can be represented. SOme of the examples are loudness per frequency per sencond interms of amplitude(magnitude) or decibles. Other example that we used was a combination of phase difference and amplitude variation over frequency. This results in a spectrogram which was our initial level of understanding for pattern recognition in various voice prints. Figure 2 displays a sample spectrogram of one of our audio files; this spectrogram shows the intensity of the sound at various frequencies across time. For the purposes of our training, we only use the intensities at a single time, rather than time-series data, which would be a single column of the spectrogram.

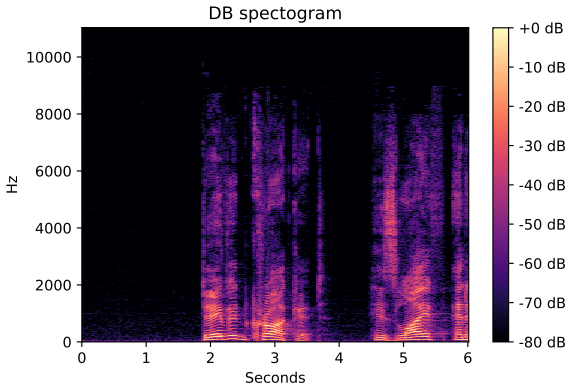

Figure 2. A spectrogram of the first 6 seconds of one of our recordings

3.2. Experimental Details

In each experiment, we trained the neural network on samples derived from several sentences from a speaker across multiple different audiobooks, and then tested on samples derived from sentences from different audiobooks. This structure was chosen to minimize the possibility of learning non-speaker features of the audio, like background noise. We chose accuracy as our preferred metric because in non-security sensitive applications (like using Amazon Echo or Google Home) a false positive and a false negative are both adverse outcomes, and neither are substantially worse than the other.

We tested several different independent variables. The first variable we tested was the parameters to the Fast Fourier Transform. The first parameter, `nfft`, refers to the number of "buckets" the frequency is divided into, with a large number meaning there is higher precision for the frequency space. The second parameter, `hop_length`, refers to the length of each sample in units of the native audio samples; lower numbers mean shorter samples are fed to the neural network. We tuned these parameters using k-nearest neighbor and support vector machine classifiers; the former is

a natural way of testing whether the parameters we chose were sufficiently precise, while the latter performed remarkably well at differentiating between speakers when used with a radial basis function kernel, comparing one speaker to all others.

The second independent variable we focused on was the number of samples we trained on per speaker. The Google Home uses only two instances of the hot phrase "Okay, Google" to train voice recognition, but it limits its recognition to only that hot phrase. Our hypothesis was that a larger number of samples would be correlated with higher accuracy predicting the speaker.

The third independent variable we tested was the number of speakers; once we had established good parameters for the FFT and an appropriate number of training samples, it was a logical step to test the accuracy as a function of number of speakers the network was trained on. We hypothesized that as the number of speakers went up, the accuracy would fall exponentially. This hypothesis was based in the fact that the network could make pairwise errors, and the number of pairs grows exponentially in the number of speakers.

The final variable we tested was our network structure. We started with a simple one-layer architecture and tested several different sizes for the hidden layer, then moved on to testing more complex network architectures. We intended to determine how large a network was "large enough" to encode the differences between speakers but as small as possible within that size, for optimum computational performance. We also trained a few multi-layer networks to see if a deep representation offered accuracy benefits over a single hidden layer.

4. Results

4.1. Data Representation

Our first focus was on determining a data representation that encoded the information that we desired. We performed the majority of these experiments with k-nearest neighbor classifiers and support vector machines so that we could parallelize along with working on our neural network programs.

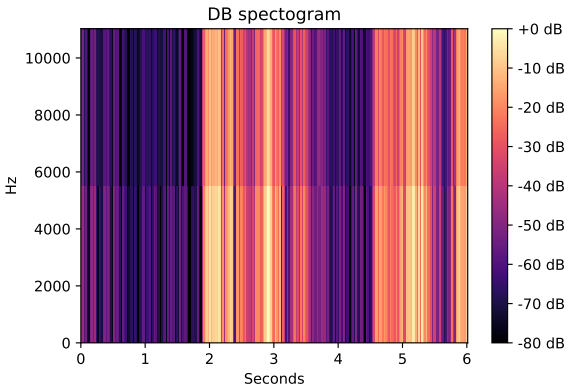

Figure 3. A spectrogram with a low `nfft`, the number of bins for the FFT

In Figure 3 we see the spectrogram of an audio sample with a very low number of FFT bins, so the granularity on the frequency is low. By trying different values for this value, we converged on 2048 bins (or approximately one bin for every 10 Hz) being an appropriate value, based on the results of using SVM on the dataset.

On the other hand, in Figure 4 we see a spectrogram with a `nfft` of 2048 but a relatively high `hop_length`, corresponding to a relatively long sample for the FFT. Our tests with SVMs indicated that approximately one quarter of a second was an appropriate value for this parameter, as any lower (see Figure 2) and the patterns in the frequencies were lessened, while any higher took longer to process (and left a smaller training set size) for no accuracy

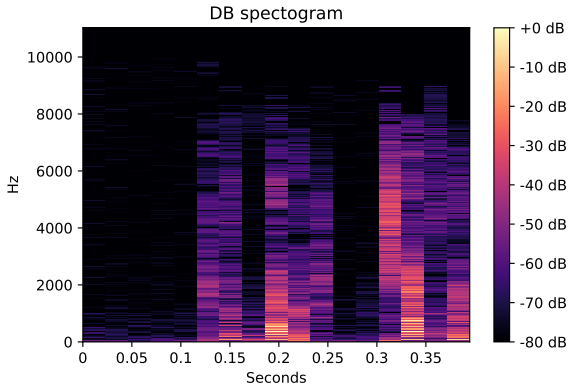

Figure 4. A spectrogram with a relatively high `hop_length`

gain in SVM.

We used a preliminary data set to benchmark the performance of SVM, k-NN, and a one hidden layer neural network. This preliminary data set was from LibriVox audiobooks but had some methodological issues, namely that the training set and testing set were interspersed heavily, as we randomly selected samples from a pool rather than differentiating them based on audiobooks.[1] With 6 speakers and approximately 20 minutes of training data per speaker (corresponding to roughly 3000 training samples), a k-NN approach was able to reach 79% accuracy, an SVM approach 96% accuracy, and the neural network approach 98.5% accuracy. This reassured us in our choices of parameters for the FFT, as well as providing us a basis for expected performance on our real dataset.

As we swtiched over to the neural network, we found that using decibels for the sound level gives bad accuracy. This is probably because decibels use a logarithmic scale where a double in sound pressure correspond to a 6dB increase in decibels. Hence, we switch to using the magnitude of the frequencies by removing the phase from the results of the FFT. Using magnitude as the feature increased the accuracy of our neural network.

[1]We would like to acknowledge Professor Shavlik for explaining why this was a problematic setup, and recommending we change it.

4.2. Samples Per Speaker

We trained various sizes of taraining data sets for a particular set of speakers. As shoud be the case, the accuracy of cross validation as well as training set increased as the number of training exambles increased. This was because the network was getting better as we trained it on more training data.

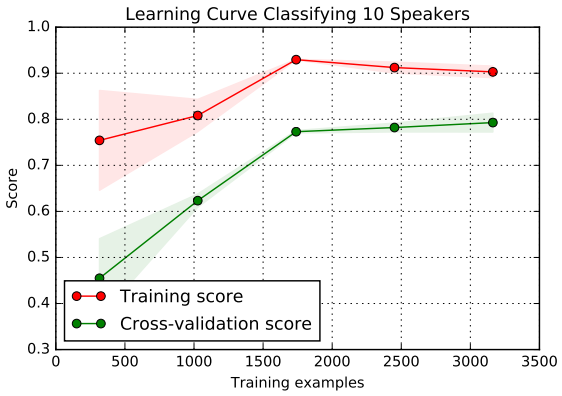

Figure 5. The accuracy learned with 10 speakers, across different numbers of training examples

Also it can be noticed that variation in accuracy of cross validation iss because the network is training itself in the initial stages. But as it gets better on its weights, the variation reduces and the growth in accuracy becomes more stable.

Figure 6 shows the same increase in the accuracy as the sample size increases. But it can ben seen from the 2 graphs together that the neural network gets trained around sample size of 1700 for 10 speakers in Figure 5, and the accuracy does not increase anyfurther by providing larger sample size. Whereas in Figure 6 where we have 20 speakers, the accuracy keeps on increasing as we provide more samples. Thought the accuracy iis not same

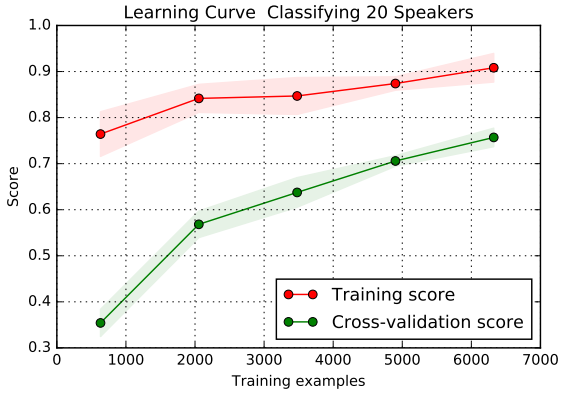

Figure 6. The accuracy learned with 20 speakers, across different numbers of training examples

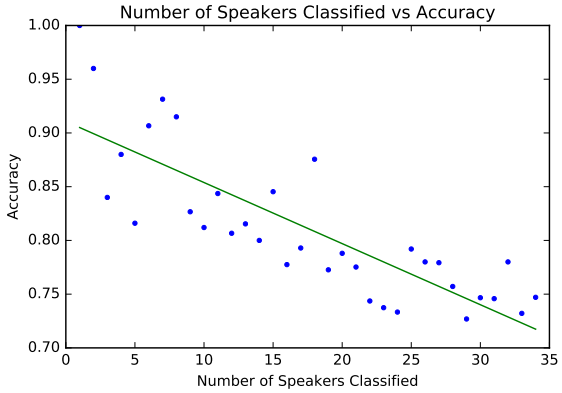

Figure 7. A plot comparing the number of speakers the network is trained to distinguish with the accuracy of its predictions

at equal sample size, the final accuracy increases for 20 speakers. This implies that as the number of speakers increase, we will need more data to distinguish between them and each individual voice print becomes more different from other in larger data set, the final accuracy increases.

4.3. Number of Speakers

We also tried varing the number of speakers while keeping the factors determing neural network and sample size same. But the results were not quite useful.

The result depicted in Figure 7 does not agree with our hypothesis. The accuracy decreases as the number of speakers the neural network tries to classify increases. The decrease is linear with gradient = -0.0057 and y-intercept = 0.91, while we had expected the decrease to be exponential. Further discussion of this point will be made in section 5.

4.4. Network Structure

The neural network structure that we selected in the end is a fully-connected 2 hidden layer neural network. We found that increasing the number hidden layers does not contribute much to the accuracy. After trying out the neural network with different number of hidden units for each layer, we concluded that 256 hidden units for the first hidden layer and 128 hidden units for the second hidden layer gives the best accuracy. This set-up gives the best accuracy for its running time as it takes longer to train when the number of hidden units increases.

We tried other activation functions such as logistic sigmoid and hyperbolic tan but the rectified linear unit function works the best. We also used the "adam" solver with a L2 penalty parameter of $10^{-}6$ after testing it out with different penalty parameter.

Figure 8 shows the accuracy of a one hidden layer neural network with the number of hidden units as the independent variable. Note that the x-axis is plotted using a logarithmic scale. It can be seen that the accuracy of the neural network increases exponentially until it hits 16 hidden units. Then, the growth of the accuracy slows down from 32 hidden units to 1024 hidden units with a mean training set accuracy of 0.9 and a tuning set accuracy/cross-validation score 0.81.

A 2 hidden layer neural network performs slightly better than a 1 hidden layer neural

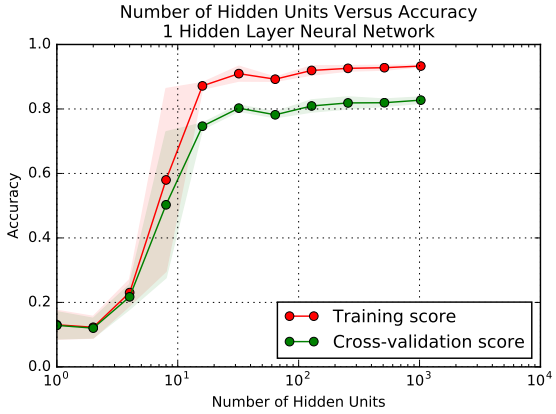

Figure 8. A plot comparing the number of hidden units in the network with the accuracy of its predictions

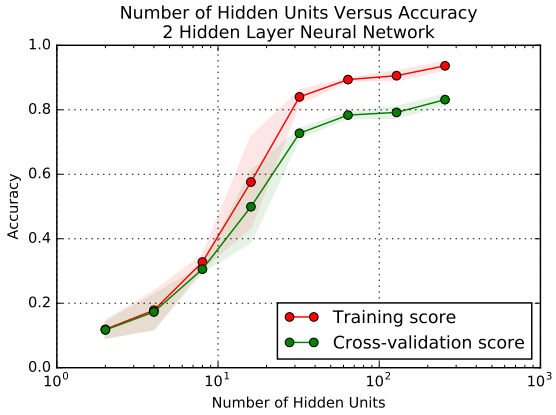

Figure 9. A plot comparing the number of hidden units in a two-layer network with the accuracy of its predictions

network. This is shown in Figure 9. The x-axis is also scaled logarithmatically and it represents the number of hidden units for the first hidden layer. The number of hidden units in the second hidden layer is half the number of hidden units in the first hidden layer. The increase in accuracy also slows down at 16 hidden units. The performance increases until 256 hidden units with a 0.93 training set accuracy and 0.83 tuning set accuracy/cross-validation score.

5. Discussion

Surprisingly, speaker recognition turned out to be a much more tractable machine learning problem than we had anticipated; even with a tiny amount of information by human standards, machine learning algorithms were able to achieve extremely high accuracy at recognition among a small number of speakers. We expected that to get high performance we would need to implement much more complex neural network structures, particularly LSTM architectures or other recurrent structures; almost all recent high-performing speech recognition software is using such recurrent neural networks. (Amodei et al., 2016) However, we saw very high performance even with a single layer of hidden units, suggesting that unique properties of a voice are encoded directly in the frequencies produced in speech, and not reliant upon time-series data.

Our hypotheses were all shown to be correct, with the notable exception of our hypothesis regarding how the number of speakers would affect the accuracy of our predictions. While we had expected that the accuracy would drop off exponentially with the number of speakers due to pairwise errors made by the network, instead the dropoff was only linear. This suggests that error rates were not primarily due to pairwise confusion.

6. Individual Contributions

6.1. Wayne Chew

Extracted data from audio files and convert them to features that can be used by the neural network. Run experiments to figure out the best set up for the neural network. Plotted out the data obtained from the experiments.

6.2. Newton Wolfe

Researched existing speaker recognition methods and how neural networks are being used

for speech to text translation. Prototyped methods for running the FFT on our dataset to extract features for the neural network. Ran early neural network experiments to determine hyperparameters and optimization details. Formalized experimental plans to develop a cohesive understanding of the neural network's performance.

6.3. Sparsh Agarwal

Worked on determining which features of voice can be useful for our categorization. We tried on using different features like magnitude, decible and various other features to determine which one gives the best unique pattern for each individual. Further helped in performing various experiments to test our network over different factors.

6.4. Keshav Sharma

Worked on experimenting with data on k-NN and SVM. Experimented with altering feature values for getting different spectograms. Futher worked with convolutional neural network and found that it would not be the right fit for the project.

7. Related Work

The vast majority of the published work using neural networks on speech is to do speech-to-text recognition and transcription. Early approaches used feedforward models, with human-engineered feature extraction followed by multilayer perceptron recognition. (Bourlard & Morgan, 1994; Renals et al., 1994) This progressed to using recurrent and convolutional neural networks, including long short term memory (LSTM) architectures. (Sak et al., 2014) In very recent times, networks that do end-to-end processing from sound to text have been developed, using a combination of recurrent and convolutional techniques, that are able to learn multiple languages from large corpora. (Amodei et al., 2016)

There has also been a small amount of work on voiceprinting using neural networks, most notably Li et al. (2015). This paper translates voiceprinting into a texture recognition problem, opening it up to various methods developed for textural processing. However, this paper is similarly restricted to known-content voiceprinting, whereas our method does not rely upon a known utterance.

8. Future Work

The experiments in this paper were performed on audio sampled from audiobooks, which lacked any disturbance in the background; while this is beneficial for training, it is not representative of the real world. As such, future work would include evaluating the neural network over a noisy data sample with background disturbances. An interesting further extension would be to concurrent-speaker-recognition, in which 2 or more speakers talk at the same time and the network attempts to identify and differentiate their voices. Finally, sentiment analysis has been a popular topic in frequent years, but generally from text sources. (Liu, 2010) A possible extension would be to enhance the neural networks to be able to analyze sentiment from short segments of speech, as intonation may encode further information than the content of a speaker's utterance.

9. Conclusions

We present a high-performance neural network for speaker recognition, which is able to differentiate between 20 audiobook readers with upwards of 75% accuracy, and performs substantially better with fewer speakers. Unlike previous published works, our network does not rely upon a fixed utterance, instead allowing any utterance to be used for authentication. This performance relies upon less than 2 min-

utes of audio per speaker, and allows for prediction based on only 250 milliseconds of an utterance, allowing for extremely fast recognition in an application.

References

Amodei, Dario, Anubhai, Rishita, Battenberg, Eric, Case, Carl, Casper, Jared, Catanzaro, Bryan, Chen, JingDong, Chrzanowski, Mike, Coates, Adam, Diamos, Greg, Elsen, Erich, Engel, Jesse, Fan, Linxi, Fougner, Christopher, Hannun, Awni, Jun, Billy, Han, Tony, LeGresley, Patrick, Li, Xiangang, Lin, Libby, Narang, Sharan, Ng, Andrew, Ozair, Sherjil, Prenger, Ryan, Qian, Sheng, Raiman, Jonathan, Satheesh, Sanjeev, Seetapun, David, Sengupta, Shubho, Wang, Chong, Wang, Yi, Wang, Zhiqian, Xiao, Bo, Xie, Yan, Yogatama, Dani, Zhan, Jun, and Zhu, Zhenyao. Deep speech 2 : End-to-end speech recognition in english and mandarin. In Balcan, Maria Florina and Weinberger, Kilian Q. (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 173–182, New York, New York, USA, 20–22 Jun 2016. PMLR. URL http://proceedings.mlr.press/v48/amodei16.html.

Bourlard, Hervé A and Morgan, Nelson. *Connectionist Speech Recognition: A Hybrid Approach*, volume 247. Springer Science & Business Media, 1994.

Ffmpeg. Ffmpeg, 2010. URL http://www.ffmpeg.org.

Geitgey, Adam. Part 6: How to do speech recognition with deep learning, 2016. URL https://medium.com/@ageitgey/machine-learning-is-fun-part-6-how-to-do-speech-recognition-with-deep-learning-28293c162f7a.

Kingma, Diederik and Ba, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Li, Penghua, Chen, Minglong, Hu, Fangchao, and Xu, Yang. A spectrogram-based voiceprint recognition using deep neural network. In *Control and Decision Conference (CCDC), 2015 27th Chinese*, pp. 2923–2927. IEEE, 2015.

Liu, Bing. Sentiment analysis and subjectivity. In *Handbook of Natural Language Processing, Second Edition*, pp. 627–666. Chapman and Hall/CRC, 2010.

McFee, Brian, McVicar, Matt, Nieto, Oriol, Balke, Stefan, Thome, Carl, Liang, Dawen, Battenberg, Eric, Moore, Josh, Bittner, Rachel, Yamamoto, Ryuichi, Ellis, Dan, Stoter, Fabian-Robert, Repetto, Douglas, Waloschek, Simon, Carr, CJ, Kranzler, Seth, Choi, Keunwoo, Viktorin, Petr, Santos, Joao Felipe, Holovaty, Adrian, Pimenta, Waldir, and Lee, Hojin. librosa 0.5.0, February 2017. URL https://doi.org/10.5281/zenodo.293021.

Panayotov, Vassil, Chen, Guoguo, Povey, Daniel, and Khudanpur, Sanjeev. Librispeech: an asr corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 5206–5210. IEEE, 2015.

Renals, Steve, Morgan, Nelson, Bourlard, Hervé, Cohen, Michael, and Franco, Horacio. Connectionist probability estimators in hmm speech recognition. *IEEE Transactions on Speech and Audio Processing*, 2(1): 161–174, 1994.

Sak, Haşim, Senior, Andrew, and Beaufays, Françoise. Long short-term memory recurrent neural network architectures for large

scale acoustic modeling. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.