Interspeech 2018
2-6 September 2018, Hyderabad, India

A comparison of input types to a deep neural network-based forced aligner

Matthew C. Kelley[1], *Benjamin V. Tucker*[1]

[1]Department of Linguistics, University of Alberta, Edmonton, AB, Canada

mckelley@ualberta.ca, bvtucker@ualberta.ca

Abstract

The present paper investigates the effect of different inputs on the accuracy of a forced alignment tool built using deep neural networks. Both raw audio samples and Mel-frequency cepstral coefficients were compared as network inputs. A set of experiments were performed using the TIMIT speech corpus as training data and its accompanying test data set. The networks consisted of a series of convolutional layers followed by a series of bidirectional long short-term memory (LSTM) layers. The convolutional layers were trained first to act as feature detectors, after which their weights were frozen. Then, the LSTM layers were trained to learn the temporal relations in the data. The current results indicate that networks using raw audio perform better than those using Mel-frequency cepstral coefficients and an off-the-shelf forced aligner. Possible explanations for why the raw audio networks perform better are discussed. We then lay out potential ways to improve the results of the networks and conclude with a comparison of human cognition to network architecture.

Index Terms: speech recognition, forced alignment, audio segmentation

1. Introduction

Forced alignment has become a common tool in the analysis of speech data [1]. A forced aligner takes in an audio file and a list of phones present in the audio signal. It then outputs the temporal boundaries for each phone. For example, in a recording of *cat*, a forced aligner would take the recording and list of [k], [æ], and [t], and then it would output the timestamps demarcating when each phone begins and ends. That is, it aligns the phone labels to their corresponding regions the audio. The present study is a preliminary exploration into different approaches to building an acoustic model using a deep neural network to create a forced aligner, with special attention paid to the effect of input type on the aligner performance.

A number of forced aligners already exist. Some rely on hidden Markov models, such as the Penn Forced Aligner [2], the ProsodyLab Aligner [3], MAUS [4], and the Montreal Forced Aligner [1], while Gentle [5] relies on neural networks. To date, there seems to be no forced alignment tool whose acoustic model is openly a deep neural network (it is uncertain how Gentle structures its neural networks, so it can't be said whether they use deep networks or not). It has previously been suggested that acoustic models using deep learning techniques can outperform models using hidden Markov models in speech recognition tasks [6]. As such, using deep neural networks as acoustic models for this task should be investigated.

Researchers in automatic speech recognition with neural networks have previously used Mel-frequency cepstral coefficients (MFCCs) as input to neural networks [7][6]. Others, however, have used raw audio samples [9][10][11]. MFCCs have also previously been found to perform approximately as well as raw audio for network inputs overall [9]. It is thus uncertain which of these two representations will perform better for the purposes of forced alignment. It's been noted in that the convolutional layers in some networks seem to be learning some sort of filter-bank [9], which bears some relation to using filter-banks when calculating MFCCs.

The goal of the present study is to assess what kind of neural network architecture and input work best for forced alignment and compare the results with already available packages for forced alignment. Based on the history of deep neural networks outperforming hidden Markov model-based models in speech recognition tasks, we predict that the deep neural networks will perform better than hidden Markov model-based forced aligners. A forced aligner needs to have high precision to be useful to researchers in phonetics, since some speech cues of interest happen at the scale of milliseconds. As such, some attention is paid to the level of precision a particular forced alignment tool affords, in addition to more typical accuracy measures used in automatic speech recognition.

2. Experiment details

Because a comparison between raw audio samples and MFCCs has yet to be performed for forced alignment, two different neural networks are trained and compared to each other. The first network is trained on raw audio samples, and the second, on MFCCs. To allow for a more even comparison between the input options, the network architecture is kept as similar as possible to control for factors related to differing architectures. As in [9], the network begins with a section of convolutional layers to serve as feature detectors. Once trained, this convolutional section's weights are frozen, and the section is then connected to another section with bidirectional long short-term memory (LSTM) layers. This section should learn temporal relationships in the data. The LSTM layers were chosen to be bidirectional because it has been found that bidirectionality can help a deep neural network perform accurately in speech recognition [7]. The overall architecture of combining convolutional layers with LSTM layers was inspired in part by remarks that there seems to have been just one attested attempt at this architecture before [12]. Further motivation is found in [13], where it's stated this kind of architecture allows for faster training than a deep recurrent neural network with an acceptable accuracy trade-off because the convolutional layers are light and don't require as many mathematical operations to train.

The TIMIT speech corpus was used as training data [14]. We selected the TIMIT corpus because it contains detailed phonetically time-aligned transcriptions, and it is used as a benchmark test for a number of different experiments in automatic speech recognition (such as in [7], [8], [9], and [11]).

10.21437/Interspeech.2018-1115

2.1. Data preparation

To prepare the data for the raw audio network, each audio file in the TIMIT corpus (including SA records) was pre-processed into frames containing 25 ms of samples (for a total of 400 samples, due to the 16 kHz sampling rate). The frames were processed at an interval of 1 ms (so a new frame would start 1 ms after the previous frame started). The label for the frame was determined based on what category the majority of the middle 1 ms of samples belonged to. In the case where this middle 1 ms was evenly distributed between two labels, the frame was assigned the earlier label to favor the duration of a phone over transition to a different one. Additionally, the audio files were zero-padded with 12 ms at the start and end so that every 1 ms frame could be assigned a label. Processing the frames in this way casts the problem as recognizing the middle 1 ms of the frame with 12 ms of context on each side of the target 1 ms. These frames serve as the raw audio input to the network.

To prepare the MFCC input to the network, each audio file in the TIMIT corpus was pre-processed into frames of 13 MFCCs, 13 delta coefficients, and 13 delta delta coefficients. The MFCCs were calculated using the Python Speech Features library [15]. As is the default in the library, the first MFCC was replaced with the log-energy of the entire frame. As before, the frame size was 25 ms, and each new frame started 1 ms after the previous one. The library padded zeros to the end of each signal so there was a whole number of frames. These MFCCs can be thought of as a spectrum of a spectrum of the energy in the frame. The delta coefficients represent the change of the MFCCs over time, and the delta-delta coefficients represent the change of the delta coefficients over time.

When determining the frame labels, we used phone collapsings following [16] and mentioned in [7] as standard for full speech recognition, with one exception. The phone [ɾ] was not collapsed into a label for silence as suggested because the flap in connected American English speech does not always create a period of silence in the acoustic signal [17], so a label for silence would describe this phone poorly and introduce noise into the silence category. Effectively, this meant that the 61 possible phones in TIMIT were collapsed into 40 different possibilities for classification, on the basis of acoustic similarity. For example, the closure periods for the oral stops [p], [t], [k], [b], [d], and [g] were all collapsed into one label, "sil" to indicate silence, since the periods of occlusion that precede them are nearly identical, acoustically. While, there may be voicing during the closure for the voiced stops [b], [d], and [g], we opted to keep the simplified phone structure as much as possible.

Finally, 184 utterances (5% of the training data) were randomly selected to be a validation set used to calculate validation accuracy during training. This 5% remained consistent throughout the training process.

2.2. Network architectures

Different configurations of the network architecture were implemented and trained using the Keras Python library [18] with the TensorFlow backend [19]. The hyperparameters for the networks were found by random search, as suggested in [20].

Each tested architecture began with a series of 1-dimensional convolutional layers that had rectified linear unit (ReLU) activation functions. There were no pooling layers because they were found to decrease the overall accuracy of the network. It's also not necessarily desirable for the results of the convolutions be time-independent, since the locus of certain frequency peaks in the raw audio can be a discriminative cue as to what phone class a given input belongs to, so max-pooling, for example, might not be helpful.

Connected to this series of convolutional layers were 2 bidirectional LSTM layers, which also had ReLU activation functions. Finally, the network ended in a fully-connected layer with a softmax activation function. It had 61 neurons, which represents the number of phone classes that the network was predicting between (though only 40 categories were used as labels), as is standard in such experiments [7]. The best architecture of those that were tested is described in detail below.

2.3. Decoding the network output

As currently designed, the neural nets perform frame classification, but not forced alignment directly. However, it's possible to use a network's frame classifications for a recording to perform alignment if the output is decoded. For the present study, the network output was decoded by the process of finding the most probable path through the output that would yield the specified phone sequence.

When an audio file is done being processed by the neural network, there is a resultant output matrix O that has 61 rows to represent each possible phone class and T columns, which are the timesteps in the network, from 1 up to and including T. From this matrix, the most probable sequence of frame labels that produce the desired final phone sequence can be determined with dynamic programming. This decoding process is given in pseudocode in Algorithm 1. Recall that the desired phone label sequence, which we will refer to as s, is provided by the user when running the forced aligner. Assume that the rows of O can be indexed by their associated phone labels, as well as their integer index. Computing boundaries between each phone from the resultant label sequence becomes straightforward because each label represents a 1 millisecond time step.

Algorithm 1 Decoding neural network output.

function DECODE(O, s)
  Define an n by T zero matrix M, where n = length(s).
  $M[1,1] \leftarrow O[s[1],1]$
  for j from 2 to T **do**
    if $j < (T - n + 1)$ **then**
      $M[1,j] \leftarrow O[s[1],j] * M[1,j-1]$
    end if
  end for
  for i from 2 to n **do**
    for j from 1 to T **do**
      if $j = i$ **then**
        $M[i,j] \leftarrow M[i-1,j-1] * O[s[i],j]$
      else if $i < j < (T - n + i)$ **then**
        $p \leftarrow max(M[i-1,j-1], M[i,j-1])$
        $M[i,j] \leftarrow p * O[s[i],j]$
      end if
    end for
  end for
  Call $argmax(\cdot)$ on each column of M and store in a.
  for j from $T - 1$ to 1 **do**
    if $\neg(a[j+1] - 1 \leq a[j] \leq a[j+1])$ **then**
      $a[j] \leftarrow a[j+1]$
    end if
  end for
  return the phone labels corresponding to the rows of O indicated by each element in a
end function

2.4. Assessing the forced alignment performance

A network's accuracy in the forced alignment task can be calculated by comparing how many of the frame labels from decoding the output match the labels in the test set. We refer to this measure as the path accuracy, denoted acc_{path}. Formally,

$$\text{acc}_{\text{path}} = \frac{n_c}{n_t}\,, \tag{1}$$

where n_c is the number of frames correctly identified, and n_t is the total number of frames.

2.5. Training the networks

The network hyperparameters were initially found for the raw audio input. The training began with finding the hyperparameters for the convolutional layers, which were connected to a 61-neuron fully-connected layer with softmax activation. The order in which the files would be trained on was shuffled at each training epoch, and each file's frame order was shuffled each time it was fed into the network. The network was tasked with predicting the phone label that corresponds to the frame that was fed into it. It was trained using categorical cross entropy as a loss metric, stochastic gradient descent with Keras' default values as the optimizer, and a batch size of 256. Each frame was shaped so that each millisecond of samples (16 samples) was its own row in the tensor, resulting in a 25 row by 16 column tensor. Each configuration was monitored with a callback in Keras that would check the validation accuracy of the network after each epoch and save the best performing configuration.

Once the convolutional layers were trained, their weights were frozen, and a new hyperparameter search began. The output layer was removed, the two bidirectional LSTM layers were added, and then another fully-connected layer with 61 neurons with softmax activation was added as an output layer. As with the previous training step, this network used stochastic gradient descent as its optimizer with Keras' default values, and its loss function was categorical cross entropy. The batch size varied a bit more here due to memory constraints on the network. Each utterance was split into 256 different batches of equal size, using zero-padding at the end if necessary to attain a whole number of batches. The same Keras callback from before was used to save each trial's most accurate network based on validation accuracy. The best configuration overall was found to be 5 layers of convolutions, with 98, 89, 66, 73, and 47 filters in each layer, respectively; kernel sizes of 1, 2, 1, 1, and 2 in each layer, respectively; stride lengths of 1, 1, 1, 2, and 1 in each layer, respectively; 2 epochs of training before the LSTM layers were attached leading to a validation accuracy of 51.7%; and 1 epoch of training once the LSTM section was attached. The LSTM layers had 64 and 156 units in each layer, respectively.

The MFCC network's architecture was identical, and the network was trained in the same manner as the raw audio network with the same hyperparameters, just with MFCCs as input instead of audio samples. Based on the validation accuracy, the convolutional section was trained for 1 epoch, and the LSTM section was trained for 7. We realize that this architecture may not be ideal for MFCC data and plan other future comparisons.

3. Results

For both of the final networks, acc_{path}, test accuracy on the TIMIT test set, and the median absolute error on the endpoints of each segment compared to the TIMIT transcriptions were calculated. We chose the median absolute error instead of the mean absolute error because we are interested in how the networks perform when they are performing as expected; the mean absolute error would be affected by outliers and not give as good of an idea of the performance. Additionally, we chose the Montreal Forced Aligner to compare the trained networks against a recently released and fairly accurate forced alignment tool [1]. Its acc_{path} was reverse engineered from the boundaries it determined, and its median absolute error were also calculated from its resulting alignment. Because the Montreal Forced Aligner's transcription scheme differed from that of TIMIT, it was run using the CMU Pronouncing Dictionary to create phone sequences from the TIMIT orthographic transcriptions. The calculated measures may be seen in Table 1.

Table 1: *Comparison measures for each aligner. Montreal Forced Aligner is represented as "MFA."*

Aligner	**acc_{path} value**	**Test acc.**	**MAE (s)**
Raw audio	74.7%	53.6%	0.008
MFCC	22.0%	0.6%	1.55
MFA	72.1%	—	0.1

The acc_{path} is highest for the raw audio network, followed closely by the Montreal Forced Aligner, and the MFCC network has by far the lowest value. Given the way that the measure was calculated, this speaks to how well the aligners did in identifying each millisecond of audio. In terms of precision of the output boundaries, the median absolute error is remarkably lower for the raw audio network than the Montreal Forced aligner, indicating that its boundaries are tighter to the ground truth in comparison to the Montreal Forced Aligner. In practice, this would suggest that the median difference between the raw audio network's boundaries and the ground truth would be 8 ms, while it would be 100 ms for the Montreal Forced Aligner. Given that the acc_{path} value for the raw audio network is as low as it is suggests that there may be some cases where its boundaries are far off the mark, since acc_{path} as defined here is susceptible to outliers. The same may be true for the Montreal Forced Aligner as well. In terms of the hypotheses that each neural net will perform better than the Montreal Forced Aligner, it was borne out that the raw audio network performed marginally better, however the MFCC network did not.

Test accuracies for our networks are not competitive with state-of-the-art systems that perform framewise phone identification, but we provide ours for comparison. Test accuracy is not calculable for the Montreal Forced Aligner since its frame identification cannot be readily accessed.

An example of the produced alignments may be seen in Figure 1. Both the raw audio network and the Montreal Forced Aligner are producing boundaries that are close to the ground truth of the TIMIT transcriptions. However, the output of the MFCC network seems to have made silence the most probable path for each frame, which will inevitably be right some of the time, as evidenced by the acc_{path} value.

4. Discussion and conclusion

Given the results, the network trained on raw audio shows promise for the forced alignment process, surpassing even the Montreal Forced Aligner. Its markedly better performance in terms of the median absolute error is indicative of the raw audio network's promise in this application. The tighter boundaries are of use to phoneticians and other researchers in need of aligned speech data. Since its median absolute error is so com-

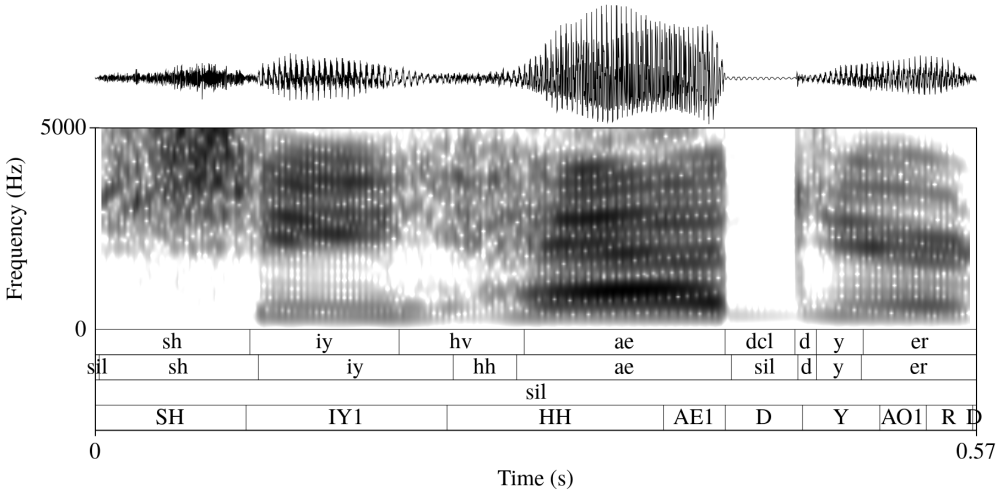

Figure 1: *Sample phone alignment from aligners for "she had your." From top to bottom: ground truth, raw audio network, MFCC network, Montreal Forced Aligner.*

paratively low, it further suggests that when the raw audio network is accurately recognizing the frames, it performs well. Its performance could be further improved by increasing its frame identification accuracy. It should be noted, however, that the Montreal Forced aligner was not trained on speech from the TIMIT speech corpus, so its results may have suffered slightly in comparison to the deep networks trained here. It also used a different transcription scheme, and though we tried to account for this when determining if it identified a frame correctly, it may still have affected accuracy score. Future work will address different transcription schemes more holistically or re-train the Montreal Forced Aligner on TIMIT for a more fair comparison.

The drastic difference between the raw audio network's performance and the MFCC network's suggests as well that the network may have learned a better representation in the convolutional layers for the raw audio than for the MFCCs or that the MFCC features the convolutional section learned weren't very applicable to the LSTM section. What the performance difference also means is that the MFCC network needs different hyperparameters and/or a different architecture than the raw audio one does. It is indeed peculiar that the MFCC model did not come close to matching the performance on the similar task of phone identification shown in [9] (where a raw audio network had a 69.47% frame identification accuracy on the TIMIT test set, and an MFCC model had a 71.80% identification accuracy on the TIMIT test set, both using a conditional random fields technique for decoding). More experimentation may reveal a better architecture and/or hyperparameters for the MFCCs.

On another note, the relative performance of the raw audio model during the different sections of its training seems to parallel some phenomena that are known about human speech perception and auditory cognition. Specifically, it's known that the process of human speech perception is dynamic and exploits temporal information and patterns in the acoustic signal [21], with even some models of speech perception taking this into account [22][23][24]. The raw audio model's performance increased once the LSTM layers were added onto it, affording it the ability to move beyond decontextualized frame classification to exploiting the temporal context of the acoustic data to inform its predictions. This similarity suggests that there is some relation between how the neural network is mathematically learning to separate the phone classes and human speech perception. Ultimately, deep neural networks may prove to be a useful tool for modeling human predictions about speech; the information being processed is similar, and the networks seem to benefit from access to similar information that humans use.

Future research on the connections between neural networks and human cognition could perhaps take the form of repeating speech perception experiments like testing a /da/-/ta/ voice onset time continuum as in [25]. The network's responses could be compared with human responses to the same stimuli. If similar results are found between the network's and humans' responses, it could be concluded that deep neural networks use perceptual information in a way similar to humans. It would further suggest that investigating what neural networks learn could provide further insight into what cues humans might be using in the process of speech perception.

In sum, forced alignment tools have yet to truly be brought into the age of deep learning. The present results are a step in that direction. The performance of the deep networks in the alignment task show promise, but require more refinement. Though the results presented here are restricted to only the TIMIT speech corpus, our modeling process is ongoing, and we are hopeful that new architecture configurations will provide even better results. After additional training and testing on other speech data sets and improving computational efficiency, we intend to release the resultant system as an open-source command-line utility.

5. Acknowledgements

This research was funded in part by a SSHRC grant to the second author and the Kule Institute for Advanced Study through the Deep Learning for Sound Recognition group at the University of Alberta. We also acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research. Additionally, the authors would like to thank the participants in the course on deep learning for sound and behavior at the University of Alberta, and Vadim Bulitko in particular. Any errors or misunderstandings are, of course, our own.

6. References

[1] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal Forced Aligner: Trainable text-speech alignment using Kaldi," in *Interspeech 2017*, 2017, pp. 498–502.

[2] J. Yuan and M. Liberman, "Speaker identification on the SCOTUS corpus," *Journal of the Acoustical Society of America*, vol. 123, no. 5, p. 3878, 2008.

[3] K. Gorman, J. Howell, and M. Wagner, "Prosodylab-aligner: A tool for forced alignment of laboratory speech," *Canadian Acoustics*, vol. 39, no. 3, pp. 192–193, 2011.

[4] T. Kisler, F. Schiel, and H. Sloetjes, "Signal processing via web services: The use case WebMAUS," in *Digital Humanities Conference 2012*, 2012, pp. 30–34.

[5] R. M. Ochshorn and M. Hawkins, "Gentle," 2017. [Online]. Available: https://lowerquality.com/gentle/

[6] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[7] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[8] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6645–6649.

[9] D. Palaz, R. Collobert, and M. Magimai-Doss, "Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks," *INTERSPEECH-2013*, pp. 1766–1770, 2013. [Online]. Available: http://arxiv.org/abs/1304.1018

[10] Z. Tüske, G. Pavel, S. Ralf, and N. Hermann, "Acoustic modeling with deep neural networks using raw time signal for LVCSR," *INTERSPEECH-2014*, pp. 890–894, 2014.

[11] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *arXiv:1609.03499 [cs]*, 2016. [Online]. Available: http://arxiv.org/abs/1609.03499

[12] Y. Zhang, M. Pezeshki, P. Brakel, S. Zhang, C. L. Y. Bengio, and A. Courville, "Towards end-to-end speech recognition with deep convolutional neural networks," *arXiv preprint arXiv:1701.02720*, 2017.

[13] F. Chollet, *Deep Learning with Python*. Shelter Island, NY: Manning Publications, 2018.

[14] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, "Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1," *NASA STI/Recon technical report n*, vol. 93, 1993.

[15] J. Lyons, "Python Speech Features," https://github.com/jameslyons/python_speech_features, 2017.

[16] T. Robinson, "Several improvements to a recurrent error propagation network phone recognition system," University of Cambridge, Department of Engineering, Tech. Rep. CUED/F-INFENG/TR82, 1991.

[17] N. Warner, A. Fountain, and B. V. Tucker, "Cues to perception of reduced flaps," *The Journal of the Acoustical Society of America*, vol. 125, no. 5, pp. 3317–3327, 2009.

[18] F. Chollet *et al.*, "Keras version 2.1.5," https://github.com/fchollet/keras, 2018.

[19] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," https://tensorflow.org, 2015. [Online]. Available: https://www.tensorflow.org/

[20] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.

[21] S. Rosen, "Temporal information in speech: Acoustic, auditory and linguistic aspects," *Phil. Trans. R. Soc. Lond. B*, vol. 336, no. 1278, pp. 367–373, 1992.

[22] K. N. Stevens, "Toward a model for lexical access based on acoustic landmarks and distinctive features," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1872–1891, 2002.

[23] D. Norris and J. M. McQueen, "Shortlist B: A Bayesian model of continuous speech recognition." *Psychological Review*, vol. 115, no. 2, p. 357, 2008.

[24] L. Ten Bosch, L. Boves, and M. Ernestus, "Diana, an end-to-end computational model of human word comprehension," in *18th International Congress of Phonetic Sciences (ICPhS 2015)*. University of Glasgow, 2015.

[25] A. M. Liberman, P. C. Delattre, and F. S. Cooper, "Some cues for the distinction between voiced and voiceless stops in initial position," *Language and Speech*, vol. 1, no. 3, pp. 153–167, 1958.