NEUFA: NEURAL NETWORK BASED END-TO-END FORCED ALIGNMENT WITH BIDIRECTIONAL ATTENTION MECHANISM

Jingbei Li[1], *Yi Meng*[1], *Zhiyong Wu*[1,2], *Helen Meng*[2], *Qiao Tian*[3], *Yuping Wang*[3], *Yuxuan Wang*[3]

[1] Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
[2] The Chinese University of Hong Kong, Hong Kong SAR, China
[3] ByteDance, Shanghai, China

{lijb19, my20}@mails.tsinghua.edu.cn, {zywu, hmmeng}@se.cuhk.edu.hk, {tianqiao.wave, wangyuping, wangyuxuan.11}@bytedance.com

ABSTRACT

Although deep learning and end-to-end models have been widely used and shown their superiority in automatic speech recognition (ASR) and text-to-speech (TTS) synthesis, state-of-the-art forced alignment (FA) models are still based on hidden Markov model (HMM). HMM has limited view of contextual information and is developed with long pipelines, leading to error accumulation and unsatisfactory performance. Inspired by the capability of attention mechanism in capturing long term contextual information and learning alignments in ASR and TTS, we propose a neural network based end-to-end forced aligner called NeuFA, in which a novel bidirectional attention mechanism plays an essential role. NeuFA integrates the alignment learning of both ASR and TTS tasks in a unified framework by learning bidirectional alignment information from a shared attention matrix in the proposed bidirectional attention mechanism. Alignments are extracted from the learnt attention weights and optimized by the ASR, TTS and FA tasks in a multi-task learning manner. Experimental results demonstrate the effectiveness of our proposed model, with mean absolute error (MAE) on test set drops from 25.8 ms to 23.7 ms at word level, and from 18.0 ms to 15.7 ms at phoneme level compared with state-of-the-art HMM based model.

Index Terms— bidirectional attention mechanism, forced alignment, end-to-end model, multi-task learning

1. INTRODUCTION

Forced alignment (FA), which produces the bidirectional mapping between the given text and speech sequences, has been widely used in speech processing tasks over decades. Such bidirectional mapping information provides the possibility of recognizing fine-grained prosody patterns of speakers, from which high-level analysis like sociolinguistical, phonetical and psycholinguistical analysis could be built [1, 2, 3, 4, 5].

Conventionally, FA models are based on hidden Markov model (HMM) [6], such as Prosodylab-Aligner [7] and Montreal forced aligner (MFA) [8]. HMM based models only have limited views of contextual information. Moreover, these models need to be trained in long pipelines which not only are resource-consuming, but also lead to error accumulation and unsatisfactory results.

With the development of deep learning and end-to-end technology, many neural network based models are proposed for automatic speech recognition (ASR) [9, 10] and text-to-speech (TTS) synthesis [11, 12, 13] and outperform conventional methods. These models commonly employ the attention mechanism [14] to capture a long view of contextual information. Besides, the attention mechanism used in ASR and TTS has also shown its capability of learning alignment information [9, 11, 12] and improved model interpretability. Then it is of possibility to build a neural network model for FA with the alignment information learnt by attention mechanism. FastSpeech [13] introduces a primary way to convert the alignment information to duration by summing the weights for each phoneme. Moreover, further improvement could be made if the alignments could be optimized by both ASR and TTS with multi-task learning.

In this paper, to improve FA with deep learning and end-to-end technologies, we propose a neural network based end-to-end forced aligner named NeuFA, in which the most essential algorithm is a novel bidirectional attention mechanism. The bidirectional attention mechanism are extended from the original attention mechanism to learn bidirectional relation mapping between two sets of key-value pairs. In the neural network based FA task, the input includes both text and speech sequences, and the bidirectional alignment information between text and speech can be learned by two separately but related tasks from text to speech (TTS) and speech to text (ASR). In this way, NeuFA integrates the alignment learning of both ASR and TTS tasks in a united framework by learning the bidirectional alignment information from the shared attention matrix in bidirectional attention mechanism and optimized by the ASR, TTS and FA tasks with multi-task learning. Then the attention weights for ASR and TTS tasks are obtained from the shared attention matrix and converted to boundaries at word or phoneme level by a boundary detector. Experimental results demonstrate the effectiveness of our proposed model. The source code of NeuFA is also available at https://github.com/thuhcsi/NeuFA.

2. METHODOLOGY

2.1. Bidirectional attention mechanism

To achieve bidirectional relation learning for parallel data, we extend conventional attention mechanism [14] to bidirectional attention mechanism to learn the bidirectional relation mapping between two sets of key-value pairs, as shown in Figure 1. Each set of keys serves as the queries for the other set of key-value pairs. Then two weighted sums of the values are produced based on a shared compatibility function between the two sets of keys. It is also formulated as:

$$A = f(K_1, K_2) \tag{1}$$

$$W_{12}, W_{21} = softmax(A, A^T) \tag{2}$$

$$O_1, O_2 = W_{12}^T V_1, W_{21}^T V_2 \tag{3}$$

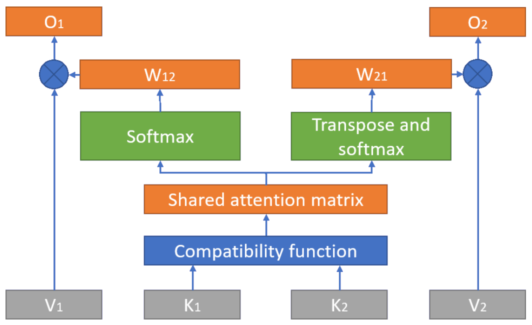

Fig. 1. Bidirectional attention mechanism

where $K_1 \in R^{n_1 \times d_{k1}}$, $V_1 \in R^{n_1 \times d_{v1}}$ and $K_2 \in R^{n_2 \times d_{k2}}$, $V_2 \in R^{n_2 \times d_{v2}}$ are the two sets of key-value pairs, n_1 and n_2 are the numbers of the key-value pairs, d_{k1}, d_{v1}, d_{k2} and d_{v2} are feature dimensions, f is the compatibility function, $A \in R^{n_1 \times n_2}$ is the shared attention matrix, $W_{12} \in R^{n_1 \times n_2}$ and $W_{21} \in R^{n_2 \times n_1}$ are the attention weights for two directions respectively, $O_1 \in R^{n_2 \times d_{v1}}$ is the weighted sum of V_1 for each key in K_2 and $O_2 \in R^{n_1 \times d_{v2}}$ is the weighted sum of V_2 for each key in K_1.

Depending on how the compatibility function is implemented, we propose a multiplicative form and an additive form for bidirectional attention mechanism. For the multiplicative form (i.e. the bidirectional multiplicative attention), the compatibility function are implemented as:

$$A = f_1(K_1) \times f_2(K_2)^T \tag{4}$$

where $f_1 : R^{n_1 \times d_{k1}} \rightarrow R^{n_1 \times d_a}$ and $f_2 : R^{n_2 \times d_{k2}} \rightarrow R^{n_2 \times d_a}$ are linear projections, d_a is the feature dimension of the attention hidden space. The additive form or the bidirectional additive attention can also be deduced from additive attention [15], which is formulated as:

$$A = f_a(dup_1(f_1(K_1)) + dup_2(f_2(K_2))) \tag{5}$$

where $dup_1 : R^{n_1 \times d_a} \rightarrow R^{n_1 \times n_2 \times d_a}$ and $dup_2 : R^{n_2 \times d_a} \rightarrow R^{n_1 \times n_2 \times d_a}$ duplicate the outputs of $f_1(K_1)$ and $f_2(K_2)$ to a same shape, $f_a : R^{n_1 \times n_2 \times d_a} \rightarrow R^{n_1 \times n_2}$ is a linear projection.

2.2. Neural network based forced alignment

In this section we will introduce our proposed neural network based FA model, NeuFA. As shown in Figure 2, NeuFA has a text encoder and a speech decoder for the TTS task, a speech encoder and a text decoder for the ASR task. These two encoder-decoder modules share a same attention module implemented with the bidirectional attention mechanism introduced in Section 2.1. Then a boundary detector is employed to extract the boundaries from the attention weights in both ASR and TTS tasks.

2.2.1. *Text encoder and speech encoder*

The text encoder and speech encoder convert the input text and speech sequences into text and speech encodings. The encoder of Tacotron 2 [12] is employed as the text encoder in NeuFA since it has already shown its ability in modeling text information in TTS. The speech encoder is designed based on the content encoder [16] proposed in voice conversion to capture the textual information in speeches. The speech encoder consists of 3 convolutional layers each containing 512 filters with shape 17×1 and batch normalization, and two 256 dimensional bidirectional GRU [17] layers.

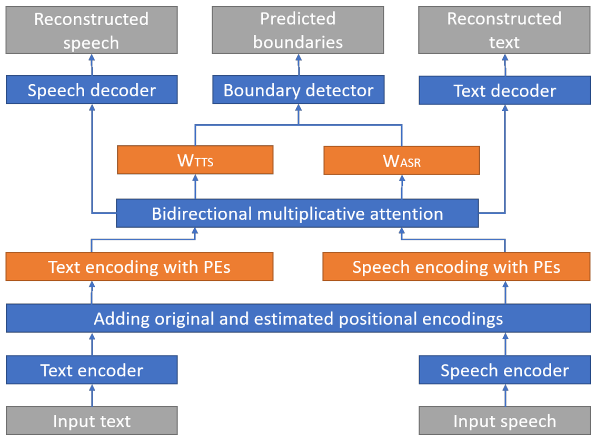

Fig. 2. Neural network based FA with bidirectional attention

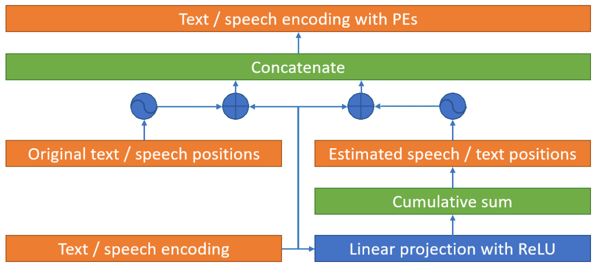
Text / speech encoding with PEs
Concatenate
Original text / speech positions
Estimated speech / text positions
Cumulative sum
Text / speech encoding
Linear projection with ReLU

Fig. 3. Adding original and estimated positional encodings

2.2.2. *Estimated positional encodings*

As shown in Figure 3, original and estimated positional encodings (PEs) are added to the text encodings and speech encodings before they are processed by the attention mechanism.

We employ the positional encodings in Transformer [14] as the original positional encodings for texts and speeches:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d}) \tag{6}$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d}) \tag{7}$$

where *pos* is the position, i is the index, and d is the number of the feature dimensions.

However, the positions (i.e. *pos*) are of different meanings in the original positional encodings for texts and speeches. For texts, the positions are the indices of the input phonemes; while for speeches, they are the indices of the frames of the input speeches. Such mismatch in the positions may confuse the model to learn clear alignments. Therefore, we propose estimated positional encodings to alleviate this mismatch by estimating the possible speech or text positions from its corresponding text or speech encodings.

The estimated positional encodings are also computed with Equation 6 and 7, except the positions are estimated positions rather than the indices of the input sequences. For texts, estimated speech positions are computed from the text encodings. And for speeches, estimated text positions are computed from the speech encodings. In computing the positions, the input encodings are first converted into estimated lengths by linear projection with $ReLU$ activation. Then the estimated lengths are cumulatively summed into monotonic estimated positions. The procedure can be formulated as:

$$pos'_s = cumsum(ReLU(f_t(E_t))) \tag{8}$$

$$pos'_t = cumsum(ReLU(f_s(E_s))) \tag{9}$$

where E_t, E_s are the input text and speech encodings, f_t, f_s are linear projections, pos'_s, pos'_t are the estimated cumulatively summed speech and text positions for the input text and speech sequences.

The original and estimated positional encodings are separately added to two copies of the input text and speech encodings, and then concatenated together:

$$E'_t = [E_t + PE_t; E_t + PE'_s] \tag{10}$$

$$E'_s = [E_s + PE'_t; E_s + PE_s] \tag{11}$$

where PE_t, PE_s are the original text and speech positional encodings, PE'_t, PE'_s are the estimated text and speech positional encodings, E'_t, E'_s are the output text and speech encodings with positional encodings added.

By introducing the estimated positional encodings to NeuFA, two additional losses called relative estimated text and speech length losses are used in back-propagation. These two losses ensure that the estimated positions are compatible with the index based positions, which can be formulated as:

$$loss_l^t = MSE(1, last(pos'_t)/L_t) \tag{12}$$

$$loss_l^s = MSE(1, last(pos'_s)/L_s) \tag{13}$$

where L_t, L_s are the ground-truth lengths of the input text and speech sequences, $last$ returns the last elements of pos'_t and pos'_s which correspond to the total estimated text and speech lengths, MSE is mean squared error.

In practise, we find that the estimated text positional encodings are significantly helpful to learn clear alignments than just original positional encodings. Moreover, these positional encodings are vital to distinguish those phonemes with the same pronunciation in sentences.

2.2.3. *Bidirectional attention mechanism*

As shown in Figure 2, the bidirectional attention takes the text and speech encodings with PEs as inputs, and respectively summarizes textual and acoustic information for the TTS and ASR tasks. We use an 128 dimensional bidirectional multiplicative attention for NeuFA since it is more efficient in memory usage.

2.2.4. *Text decoder and speech decoder*

The text decoder and speech decoder respectively reconstruct the input text and speech sequences from the summarized acoustic and textual information. The text decoder consists of a stack of two 128 dimensional bidirectional LSTM layers and a linear projection with softmax activation. The speech decoder has a similar structure, consisting of a stack of two 256 dimensional bidirectional LSTM layers and a linear projection. Cross entropy and MSE are used as the loss functions for the reconstructed text and speech sequences:

$$loss_t = CrossEntropy(T', T) \tag{14}$$

$$loss_s = MSE(S', S) \tag{15}$$

where T and S are the input text and speech sequences, T' and S' are the reconstructed text and speech sequences.

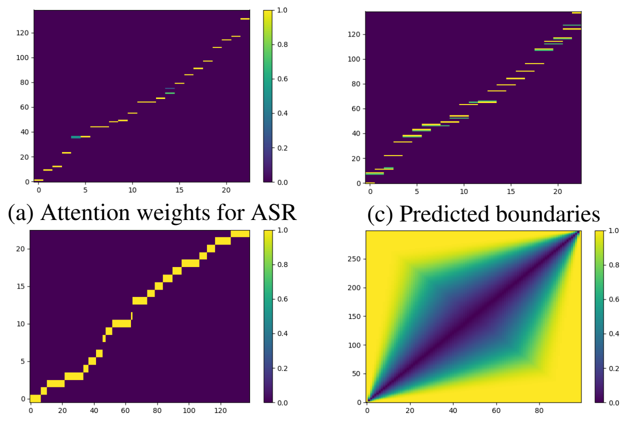
(a) Attention weights for ASR (c) Predicted boundaries (b) Attention weights for TTS (d) Diagonal constraint matrix

Fig. 4. (a) Learnt attention weights for ASR. (b) Learnt attention weights for TTS. (c) Illustration of predicted (green) and ground-truth (yellow) phoneme boundaries. (d) Illustration of a diagonal constraint matrix used in the diagonal attention loss.

2.2.5. *Boundary detector*

Instead of directly predicting the boundaries, the boundary detector takes the attention weights from both ASR and TTS directions as inputs to predict the left and right boundary signals for each phoneme. The boundary signals are inspired by the stop token in Tacotron [11] and Tacotron 2 [12], which can provide more fine-grained gradients for each position than just the values of boundaries. Ground-truth boundary signals are generated from the annotated boundaries, in which the elements are set to 0 if they are before the boundaries and 1 otherwise. Therefore the predicted boundary signals are also monotonic signals ranged from 0 to 1. During inference, the positions of the first elements whose boundary signals are greater than 0.5 are used as the predicted boundaries.

A 6 channel feature matrix is generated from the input attention weights, including the original attention weights and their cumulative sums in both forward and backward directions:

$$\begin{aligned} F = [&W_{TTS}, cumsum(W_{TTS}), r(cumsum(r(W_{TTS}))), \\ &W_{ASR}^T, cumsum(W_{ASR}^T), r(cumsum(r(W_{ASR}^T)))] \end{aligned} \tag{16}$$

where W_{TTS} and W_{ASR} are the attention weights for the TTS and ASR directions, r reverses its input along the temporal dimension and F is the output 6 channel feature matrix. W_{ASR} is transposed due to the Equation 2 in bidirectional attention mechanism.

The feature matrix F is then processed by a stack of 3 convolutional layers each containing 32 filters with shape 17×17 and a linear projection followed by the $sigmoid$ activation to detect the left and right boundaries. The outputs are cumulatively summed and converted to monotonic boundary signals ranged from 0 to 1 by the $tanh$ activation for each left and right boundaries.

$$B' = tanh(cumsum(sigmoid(f(convs(F))))) \tag{17}$$

where $convs$ is the stacked convolutions, f is linear projection and B' is the predicted boundary signals.

MAE of the predicted and ground-truth boundary signals is used as the loss function for boundary detector:

$$loss_b = MAE(B', B) \tag{18}$$

where B and B' are the ground-truth and predicted boundary signals respectively.

2.2.6. *Diagonal attention loss*

Diagonal attention loss is proposed to help the model learn alignments by multiplying the elements in attention weight matrices with diagonal constraints. The diagonal attention loss is defined as:

$$loss_a = mean\left((W_{TTS} + W_{ASR}^T) \odot D\right) \quad (19)$$

$$D_{i,j} = tanh\left(\frac{1}{2}max\left(\frac{p}{q}, \frac{q}{p}, \frac{1-p}{1-q}, \frac{1-q}{1-p}\right)\right) \quad (20)$$

where $\odot$ is element-wise multiplication and D is dynamically generated with Equation 20 where i, j are the indices, $p = \frac{i}{n_1}$, $q = \frac{j}{n_2}$ are the relative position of i, j in their corresponding dimensions. An illustration of D is also shown in Figure 4 (d).

The final loss function for training NeuFA is then defined as:

$$loss = \alpha loss_t + \beta loss_s + \gamma loss_l^t + \delta loss_l^s + \epsilon loss_a + \zeta loss_b \quad (21)$$

where α to ζ are loss weights.

3. EXPERIMENTS

3.1. Training setups

We follow the training setups of the MFA [8] to train NeuFA except the MFCCs are extracted with librosa [18] rather than Kaldi [19], and graphme-to-phoneme conversions are made by a pretrained model [20]. We train an NeuFA for word level as if only orthographic transcription in Buckeye [21] were known, and another NeuFA for phoneme level with the phoneme sequences given in Buckeye.

Each NeuFA model is firstly trained on the full set of the LibriSpeech [22] corpus for 120,000 steps with a batch size of 16. The learning rate is fixed to 10^{-4}. Loss weights α to δ are simply set to 0.1, 1, 10, 10 to balance the losses to a same level. ϵ is set to 1,000 to help the model learning alignments. ζ is set to 0 since there is no boundary annotation for the LibriSpeech corpus.

Then each model is trained on the data of 36 speakers in the Buckeye corpus for 220,000 steps with a batch size of 16 to train the boundary detector. The learning rate is also fixed to 10^{-4}. Loss weights α to δ are same to those in the previous stage. ζ is now set to 100 and ϵ is set to 0 since the boundary loss can provide more accurate alignment information than the diagonal attention loss.

The experiments are implemented with PyTorch [23] on an NVIDIA Tesla V100 GPU.

3.2. Experimental results

The data of the rest 4 unseen speakers in the Buckeye corpus are used as the test set. The models are evaluated on the MAE and medians of absolute errors of the predicted left and right boundaries at word and phoneme levels. Then the comparisons are made with the model shipped in the MFA package as the baseline.

As shown in Table 1, NeuFA outperforms MFA at both word and phoneme levels. The MAE is reduced to 23.7 ms and 15.7 ms respectively. And the medians drops to 9.0 ms and 9.1 ms, which means that NeuFA always predicts more accurate boundaries than MFA. This can also be demostrated by the accuracies at different tolerances (percentage below a cutoff) evaluated and shown in Table 2. The accuracies are improved by 0.14, 0.04, 0.01 at 10, 25 and 50 ms tolerances for word level, and 0.05, 0.03, 0.01 at corresponding tolerances for phoneme level. At 100 ms tolerance, the performance of NeuFA is on par with MFA.

Table 1. Performances of the baseline and proposed approaches

	Word level		**Phoneme level**	
Approach	**mean**	**median**	**mean**	**median**
MFA	25.8 ms	12.3 ms	18.0 ms	10.0 ms
NeuFA	**23.7** ms	**9.0** ms	**15.7** ms	**9.1** ms
- w/o EPEs	32.1 ms	11.5 ms	19.5 ms	9.9 ms
- w/o TPEs	33.8 ms	12.0 ms	20.7 ms	10.0 ms
- w/o SPEs	24.2 ms	9.1 ms	16.8 ms	9.2 ms
- w/o ASR	37.8 ms	14.0 ms	24.6 ms	10.8 ms
- w/o TTS	50.7 ms	18.7 ms	33.5 ms	15.8 ms
- w/o DAL	26.6 ms	10.0 ms	18.7 ms	10.1 ms

Table 2. Accuracies at different tolerances for different approaches

Approach	**10 ms**	**25 ms**	**50 ms**	**100 ms**
MFA (word)	0.41	0.78	0.91	0.96
NeuFA (word)	**0.55**	**0.82**	**0.92**	**0.96**
MFA (phoneme)	0.50	0.84	0.94	0.98
NeuFA (phoneme)	**0.55**	**0.87**	**0.95**	**0.98**

3.3. Ablation studies

In ablation studies, the performance drops a lot if we remove the estimated postional encodings (EPEs) and only use the original positional encodings to train the model. This illustrates the mismatch between original text and speech position encodings.

The performance also drops if we remove the original and estimated text positional encodings (TPEs) and only use the original and estimated speech positional encodings (SPEs) to train the model. However, it barely drops if we do the opposite. This is reasonable since estimating the number of phonemes from speech is more easier and accurate than estimating the duration of speech from phonemes.

Significant performance loss will occur if the ASR or TTS task is removed by setting the corresponding loss weight to 0. This shows the necessity of both the ASR and TTS tasks for FA, and also the necessity of the proposed bidirectional attention mechanism.

No alignment is learnt on the LibriSpeech corpus if we remove the diagnol attention loss (DAL) by setting ϵ to 0. The training on the Buckeye corpus will then be similar to the training from random initialization. Although the model can still learn alignments with the full architecture of NeuFA, it suffers from the absence of fine-trained bidirectional text-speech mapping information.

4. CONCLUSIONS

To improve FA with deep learning and end-to-end technologies, we propose a neural network based end-to-end forced aligner named NeuFA based on the bidirectional attention mechanism which is also proposed in this paper. NeuFA integrates the alignment learning of both ASR and TTS in a united framework by the proposed bidirectional attention mechanism. Attention weights for two tasks are obtained from the shared attention matrix in bidirectional attention mechanism, and converted to boundaries at word or phoneme level by a boundary detector. The effectiveness of NeuFA has been demonstrated by experimental results and ablation studies.

Acknowledgment: This work is partially supported by National Natural Science Foundation of China (NSFC) (62076144) and National Social Science Foundation of China (NSSF) (13&ZD189).

5. REFERENCES

[1] Martine Adda-Decker and Natalie D Snoeren, "Quantifying temporal speech reduction in french using forced speech alignment," *Journal of Phonetics*, vol. 39, no. 3, pp. 261–270, 2011.

[2] Christian DiCanio, Hosung Nam, Douglas H Whalen, H Timothy Bunnell, Jonathan D Amith, and Rey Castillo García, "Using automatic alignment to analyze endangered language data: Testing the viability of untrained alignment," *The Journal of the Acoustical Society of America*, vol. 134, no. 3, pp. 2235–2246, 2013.

[3] William Labov, Ingrid Rosenfelder, and Josef Fruehwald, "One hundred years of sound change in philadelphia: Linear incrementation, reversal, and reanalysis," *Language*, pp. 30–65, 2013.

[4] Barbara Schuppler, Mirjam Ernestus, Odette Scharenborg, and Lou Boves, "Acoustic reduction in conversational dutch: A quantitative analysis based on automatically generated segmental transcriptions," *Journal of Phonetics*, vol. 39, no. 1, pp. 96–109, 2011.

[5] Jiahong Yuan, Mark Liberman, and Christopher Cieri, "Towards an integrated understanding of speaking rate in conversation," in *Ninth International Conference on Spoken Language Processing*, 2006.

[6] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *IEEE ASSP Magazine*, vol. 3, no. 1, pp. 4–16, Jan. 1986, Conference Name: IEEE ASSP Magazine.

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

[8] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger, "Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi," *Proc. Interspeech 2017*, pp. 498–502, 2017.

[9] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2016, pp. 4960–4964.

[10] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang, "Conformer: Convolution-augmented Transformer for Speech Recognition," *arXiv:2005.08100 [cs, eess]*, May 2020, arXiv: 2005.08100.

[11] Yuxuan Wang, R. J. Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc Le, Yannis Agiomyrgiannakis, Rob Clark, and Rif A. Saurous, "Tacotron: Towards End-to-End Speech Synthesis," *arXiv:1703.10135 [cs]*, Mar. 2017, arXiv: 1703.10135.

[12] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, R. J. Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, and Yonghui Wu, "Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions," *arXiv:1712.05884 [cs]*, Dec. 2017, arXiv: 1712.05884.

[13] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu, "FastSpeech: fast, robust and controllable text to speech," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 3171–3180. Curran Associates Inc., Red Hook, NY, USA, Dec. 2019.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., pp. 5998–6008. Curran Associates, Inc., 2017.

[15] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *arXiv:1409.0473 [cs, stat]*, Sept. 2014, arXiv: 1409.0473.

[16] Kaizhi Qian, Yang Zhang, Shiyu Chang, Mark Hasegawa-Johnson, and David Cox, "Unsupervised Speech Decomposition via Triple Information Bottleneck," in *Proceedings of the 37th International Conference on Machine Learning*. Nov. 2020, pp. 7836–7846, PMLR, ISSN: 2640-3498.

[17] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," *arXiv:1409.1259 [cs, stat]*, Sept. 2014, arXiv: 1409.1259.

[18] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*. Citeseer, 2015, vol. 8, pp. 18–25.

[19] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, and Petr Schwarz, "The Kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. 2011, IEEE Signal Processing Society.

[20] Jingbei Li, "Pre-trained Grapheme-to-Phoneme (G2P) models," Aug. 2021, https://github.com/petronny/g2p.

[21] Mark A. Pitt, Keith Johnson, Elizabeth Hume, Scott Kiesling, and William Raymond, "The Buckeye corpus of conversational speech: labeling conventions and a test of transcriber reliability," *Speech Communication*, vol. 45, no. 1, pp. 89–95, Jan. 2005.

[22] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2015, pp. 5206–5210, ISSN: 2379-190X.

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, Eds., pp. 8024–8035. Curran Associates, Inc., 2019.