INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Shallow-Fusion End-to-End Contextual Biasing

Ding Zhao, Tara N. Sainath, David Rybach, Pat Rondon, Deepti Bhatia, Bo Li, Ruoming Pang

Google, Inc., USA

{dingzhao,tsainath,rybach,rondon,bhatiad,boboli,rpang}@google.com

Abstract

Contextual biasing to a specific domain, including a user's song names, app names and contact names, is an important component of any production-level automatic speech recognition (ASR) system. Contextual biasing is particularly challenging in end-to-end models because these models keep a small list of candidates during beam search, and also do poorly on proper nouns, which is the main source of biasing phrases. In this paper, we present various algorithmic and training improvements to shallow-fusion-based biasing for end-to-end models. We will show that the proposed approach obtains better performance than a state-of-the-art conventional model across a variety of tasks, the first time this has been demonstrated.

1. Introduction

Incorporating contextual information is critical to improving the performance of ASR systems [1]. Contextual information could include a user's favorite songs, contacts, apps or current location. Typically, this information is represented by an independently-trained contextual n-gram language model, represented as a weighted finite state transducer (FST). Traditional ASR systems, consisting of separate acoustic, pronunciation, and language models (AM, PM, LM), bias the recognition process towards a specific context by merging the contextual and ASR LMs [1].

End-to-end (E2E) systems combine the AM, PM, and LM as a single neural network. Incorporating contextual knowledge in E2E models is challenging for a variety of reasons. First, E2E models use far less text data during training compared to conventional LMs, which decreases their exposure to context-specific data. Thus, we find E2E models making more errors in rare, context-dependent words and phrases, like proper nouns, compared to conventional models. Second, for efficient decoding E2E models must prune to a small number of candidates ($\sim 4-10$) at each step of the beam search. Hence rare words and phrases, like context-dependent n-grams, are likely to fall off the beam.

Past work explored incorporating an independent contextual n-gram LM into the E2E framework [2] for contextual modeling, also known as *shallow fusion* [3]. However, their methods do poorly on proper nouns. With their methods, proper nouns are usually pruned during beam search before biasing can be applied, as biasing happens at the end of a word (rather than the grapheme/wordpiece units the E2E model predicts).

[4] showed that it is more effective to perform biasing within the E2E model, in keeping with the theme of all-neural optimization. However, one of the concerns with all-neural biasing is that word error rate (WER) degrades when scaling up to a large number of n-grams. Another concern is that because contextual biasing may always be active, the ASR performance should not degrade on utterances we do not want to bias, which we will refer to as "anti-context". Anti-context results were not shown in [4].

In this work, we explore various improvements to shallow fusion to address some of the challenges in E2E modeling and concerns with all-neural biasing. First, to address early pruning of contextual n-grams, we explore biasing at the sub-word unit level (grapheme, wordpiece) rather than the word-level. Second, we explore applying the contextual FST before beam pruning rather than after. Third, because contextual n-grams are typically used with a common set of prefixes ("call", "text"), we investigate incorporating these prefixes into shallow fusion, similar to conventional models [2]. We find this helps tremendously to avoid degradation for anti-context while maintaining biasing gains. Fourth, to help with modeling proper nouns, we explore various techniques to leverage a larger corpus of text-only data. Specifically, we compare performance by (1) creating a large number of proper noun text-only queries, and synthesizing corresponding speech (2) leveraging a large amount of unsupervised audio-text data, filtered to keep data with proper nouns and (3) fuzzing training data transcripts to create more proper nouns.

We report results across four different contextual test sets. We find our proposed changes to the contextual FST construction lead to significant improvements in shallow-fusion based biasing compared to past work [2, 4]. In addition, we observe that, by improving proper noun modeling by training with a large amount of unsupervised data, we can improve performance further. Overall, we are able to develop an end-to-end biasing solution that outperforms a conventional context-independent phoneme based CTC model with PM and LM [5] by 20-40% relative across almost all test sets.

2. End-to-End Biasing Improvements

2.1. Shallow Fusion Biasing

Traditional ASR systems (with separate AM, PM, and LMs), perform contextual biasing by representing a list of biasing phrases as an n-gram finite state transducer (FST) and composing it with the ASR LM during decoding [6]. This helps increase the probability of the n-grams in the contextual FST, and reduce WER in certain scenarios. We will use a similar technique to build a contextual FST, and then incorporate it into the E2E decoding framework.

Given a set of acoustic observations $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_K)$, E2E models provide posterior probabilities for a set of sub-word units $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_L)$ given these observations, that is, $P(\mathbf{y}|\mathbf{x})$. Shallow fusion interpolates the score from the E2E model with an external contextual LM during beam-search decoding, given by (1).

$$\mathbf{y}^* = \arg\max_{\mathbf{y}} \log P(\mathbf{y}|\mathbf{x}) + \lambda \log P_C(\mathbf{y}) \qquad (1)$$

Here, $P_C(\mathbf{y})$ is the score from the contextual LM and λ is a tunable hyperparameter controlling how much the contextual LM influences the overall model score during beam search.

To construct the contextual LM for E2E models, we assume

http://dx.doi.org/10.21437/Interspeech.2019-1209

that a set of word-level biasing phrases are known ahead of time, and compile them into an n-gram weighted finite state transducer (WFST) [7]. This word-level WFST, G, is then left-composed with a "speller" FST, S, which transduces a sequence of graphemes/wordpieces into the corresponding word. Similar to [3], we obtain the subword FST as the contextual LM, $min(det(S \circ G))$.

2.2. Previous Work

Biasing of E2E models with shallow fusion has been explored previously in [2], in which the contextual LM was applied only at word boundaries, similar to biasing with traditional systems [6]. This approach was not found to be effective when the list of contextual phrases contains proper nouns (e.g., song names or contacts). Because E2E models predict sub-word unit labels $\mathbf{y}$ (graphemes, wordpieces) during the beam search, applying a contextual LM at word boundaries will not work if the words to be biased are not present in the beam.

To address this issue, [4] looked at pushing the weights of the subword FST to each subword unit. To make the subword FST deterministic, the authors use the same weight for every subword unit. To avoid artificially giving weight to the candidates that match the prefixes but not the entire phrase, they also included failure arcs, which we show for illustrative purposes in Figure 1. Unlike the n-gram FST, whose failure arcs do not carry weights [6], the subword FST contains failure arcs that negates the weights that have been added before reaching the current state. Biasing per subword unit was found to be more effective than biasing at the end of each word, as it helps keeping the biasing candidates in the beam. However, the authors only explored the idea using grapheme subword units. In addition, the authors never explored results with "anti-context", to ensure that biasing does not affect recognition quality if all biasing phrases are not relevant. The following sections introduce improvements to shallow-fusion E2E biasing to address theses concerns.

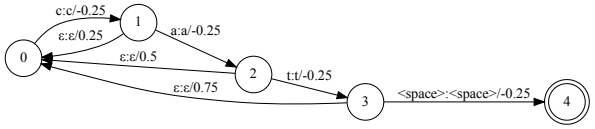

Figure 1: *Contextual FST for the word "cat", represented at the subword unit level with backoff arcs.*

2.3. Shallow Fusion Improvements

2.3.1. Biasing Before Beam Pruning

All previous biasing work, for both conventional and E2E models [2, 4, 6], combines scores from the contextual LM and the base model (ASR LM or E2E model) on the word or subword lattice, a concept known as on-the-fly lattice rescoring [8]. E2E models are decoded with tight beam thresholds resulting in a much smaller number of path hypotheses compared to conventional models. Although the contribution of the contextual model impacts future decoding process, there is a high chance that it does not affect immediate pruning of a particular biased candidate, as hypotheses may already been pruned before the contextual model is applied. In this work, we explore applying the contextual model directly to the E2E model score *before* the general beam pruning.

2.3.2. Grapheme Model vs Wordpiece Model

A concern with biasing graphemes, as in [2, 4], is that we can flood the beam with lots of unnecessary words that have a partial grapheme match with the contextual FST. For example, imagine we want to bias the word "cat", and thus a contextual FST is constructed such that we bias 'c', 'a', and 't', as shown in Figure 1. When we bias towards the grapheme 'c', we might bring not only "cat" but also "car" into the beam as we continue decoding.

However, if we bias at the wordpiece level, there is a sparser match of relevant subwords, and therefore more relevant words come onto the beam. Following the previous example, if the wordpiece to bias is "cat", then "car" would never come into the beam. In this work, we explore using a 4,096-wordpiece vocabulary, as we have found experimentally this to give reasonable results.

2.3.3. Prefixes

Shallow fusion biasing manipulates the posterior probabilities. We found that shallow fusion biasing described above hurts recognition quality on utterances that do not contain any biasing phrase, a setting known as anti-context. Similar to conventional models [9], we explore only activating a biasing phrase if it is proceeded by a commonly used set of prefixes. For example, a contact request typically has the prefix "call", "text", or "message", while a song request often uses the prefix "play".

In this work, we construct the contextual FST with prefixes mined from data. We extract all prefixes appearing more than 50 times that precede biasing phrases. Overall we find 292 phrases for contact requests, 11 phrases for song requests and 66 phrases for app requests. We construct an unweighted prefix FST and concatenate it with the contextual FST. We also allow an empty prefix option to skip the prefix. That alone, however, would cancel the intended constraining effect of the activation prefix. Thus we use a smaller biasing weight λ if the phrase is preceded with an empty prefix to bias against it.

2.4. Proper Noun Knowledge

2.4.1. Unsupervised Data

One idea to improve proper noun coverage is to train with a large amount of unsupervised data. The unsupervised data is collected by mining anonymized utterances from voice search traffic. These utterances are then decoded by a state-of-the-art conventional ASR model [10] and only utterances with a high confidence are kept. Finally, to ensure that we select utterances mainly focused on proper nouns, we run a proper noun tagger [11, 12, 13] on the transcripts and only keep utterances with proper nouns.

We train the model on roughly 100 million unsupervised utterances created using above method, along with the 35 million supervised utterances. To ensure that training is not dominated by the larger amount of unsupervised data, during each step of training, we choose to fill the batch with supervised data 80% of the time, and unsupervised data 20% of the time, as this was found empirically to give the best performance [14].

2.4.2. Synthesizing Proper Nouns

An issue with unsupervised data is that the transcripts could be wrong. It also limits the alternative spellings of names, such as Eric or Erik or Erick, to the ones predicted by the conventional model. As an alternate, we created synthetic training datasets by generating sentences with a variety of proper nouns and then synthesizing this data, using a concatenative TTS approach with only 1 voice [15].

We mined the web for various contextual biasing categories, namely communication, media and apps. Overall we extracted roughly 580K contact names, 42K song names and 70K app

names. Next, we mined the logs for various prefixes used with each category e.g. `call John mobile` gives us the prefix `call` for the communication category. The phrases are k-anonymized, i.e., they are selected only if used by at least 50 different users during a week. Then we combine the category-specific prefixes and proper nouns to generate the utterance text and use a synthetic speech data generator to create training sets with roughly 1 million utterances for each category.

Similar to our supervised training set [10], we add noises to synthesized utterances using a room simulator, as described in Section 3.1. During training, each batch is filled with supervised or synthetic data 90% and 10% of the time respectively.

2.4.3. Fuzzing Transcripts

Finally, we explore if we can introduce more proper nouns into the supervised training data itself. Specifically, we run a proper noun tagger on each utterance. For each proper noun, we get the phonetic representation of the proper noun. For example "Caitlin" is represented by the phonemes `k eI t l @ n`. Next, we look at alternative words in the lexicon with the same phoneme sequence, for example "Kaitlyn". Given the ground truth and alternative words, we randomly sample one of these words during training. This gives the model more proper nouns during training. Our intuition is that if the model can spell more names during training, it will be more confident to spell these names when a contextual FST is used during decoding, and words will not fall off the beam.

3. Experimental Details

3.1. Data Sets

The supervised training set used for experiments consists of 35 million English utterances ($\sim 27,500$ hours). The training utterances are anonymized and hand-transcribed, and are representative of Google's voice search traffic. This data set is created by artificially corrupting clean utterances using a room simulator, adding varying degrees of noise and reverberation such that the overall SNR is between 0dB and 30dB, with an average SNR of 12dB [16]. The noise sources are from YouTube and noisy environmental recordings.

We evaluate biasing on a number of test sets, detailed in Table 1. *VS* contains short voice search queries extracted from Google traffic and anonymized. While this set is not used for biasing, we use it to report "anti-context" performance.

Table 1: *Description of evaluated test sets.* `B-OOV` *is the oov rate in context-phrases only (i.e., rihanna, John, etc.).*

Test Set	# Utts	Ave. # Biasing Phrases	`B-OOV`, Sup.	`B-OOV`, Sup. + Unsup.	Audio Source
VS	14k	-	-	-	Real
Songs	15k	303	1.4%	1.0%	TTS
Cnt-TTS	15k	75	6.2%	0.2%	TTS
Cnt-Real	5k	197	6.9%	4.7%	Real
Apps	16k	13	1.5%	0.5%	TTS

As described in [4], the *Songs*, *Cnt-TTS*, and *Apps* sets are created by synthesizing sentences in each of these categories using a parallel-wavenet approach [17] with the same voice used to generate synthetic training data in in Section 2.4.2. During recognition, the ground truth proper noun in the utterance, along with some other proper nouns in the same category, are provided as a set of biasing phrases. The *Songs* test set contains media requests (e.g. `play rihanna music`) with biasing phrases containing popular songs and artist names in US-English. The *Cnt-TTS* test set contains communication requests (e.g. `call John mobile`) with biasing phrases containing popular US-English names. Finally, the *Apps* test set contains requests to interact with an app (e.g. `open trivia game`) with biasing phrases containing popular app names. Noise is artificially added to the synthetic data, similar to [16]. Note that none of the synthetic test sets have phrases that appear in the TTS synthesized training data.

Finally, the *Cnt-Real* set contains anonymized and hand-transcribed utterances extracted from Google traffic. Only utterances with an intent to communicate with a contact are included in the test set. The contact names are tagged in the transcript, and also included as part of the biasing set along with other popular US-English names. The *Cnt-Real* dataset covers the harder utterances where the user issues a typed query following the voice action, which likely means that the recognition result was not correct. It also acts as a good sanity test to ensure the model does not over-fit to synthetic data.

3.2. Modeling

All experiments use 80-dimensional log-Mel features, computed with a 25ms window and shifted every 10ms. Similar to [10], at the current frame, t, these features are stacked with 3 frames to the left and downsampled to 30ms frame rate. All experiments use the Recurrent Neural Network Transducer (RNN-T) model [18]. Specifically, the encoder network architecture consists of a time reduction layer [19], followed by 8 long short-term memory LSTMs [20], where each layer has 2,000 hidden units followed by a 600-dimensional projection layer. The decoder consists of 2 LSTM layers with 2,000 hidden units and a 600-dimensional projection layer. The encoder and decoder are fed to a joint-network that has 600 hidden units. The joint network is fed to a softmax layer, with either 96 units (for graphemes) or 4,096 units (for wordpieces [21]). All RNN-T models are trained in Tensorflow [22] on 8×8 Tensor Processing Units (TPU) slices with a batch size of 4,096.

During inference, each utterance is associated with a set of biasing phrases used to construct a contextual FST. Each arc in this FST has the same weight, as shown in Figure 1. This weight is tuned independently for each category (songs, contacts, etc.) [9] to optimize performance on the above test sets.

4. Results

4.1. Shallow Fusion

Table 2 shows the proposed algorithmic improvements, as discussed in Section 2.3. Experiments $E0$ and $E1$ are grapheme and wordpiece baselines without biasing. $E2$ shows grapheme biasing results without any of the proposed improvements. As noted in [4] and also shown in $E3$, using a subtractive cost to prevent keeping bad candidates on the beam gives improvements across all sets. Switching from grapheme to wordpiece biasing ($E4$), such that we bias at longer units, helps to keep more relevant candidates on the beam, and also improves performance. Finally, applying the biasing FST before beam search pruning, denoted as *early biasing* below, helps to ensure that good candidates remain on the beam early on, and leads to additional improvements ($E5$). Overall, our best shallow fusion setup is to bias at the wordpiece level with subtractive cost and early biasing.

Table 2: *Shallow Fusion Results.*

Exp ID	Model	*Songs*	*Cnt-Real*	*Cnt-TTS*	*Apps*
E0	Grapheme, No Biasing	19.6	18.6	28.3	13.0
E1	WPM, No Biasing	22.7	15.8	37.0	13.3
E2	Grapheme, Biasing	14.8	12.5	24.2	8.7
E3	+ subtractive cost	8.1	10.0	13.2	5.8
E4	+ WPM	6.2	8.3	10.0	3.6
E5	+ early biasing	5.3	7.5	7.3	2.7

4.2. Anti-Context

Since contextual biasing may be always active, we must make sure the recognizer does not degrade performance when all biasing phrases are not present in an utterance. To test this, we explored biasing the *VS* test set with a biasing FST constructed from 200 biasing phrases randomly selected from the biasing contexts of the *Cnt-TTS* test sets. The rest of the test sets use the same biasing contexts as the previous section.

Table 3 shows anti-context results. $E1$ is the baseline no-biasing wordpiece model. Biasing this model ($E5$) gives a large degradation in performance on *VS*. As discussed in Section 2.3.3, conventional models address this issue by including a prefix in the biasing FST. If we apply biasing only after seeing any of the non-empty prefixes ($E6$), we can improve results on *VS* but degrade quality on the biasing sets. However, in $E7$, if we allow one of the prefixes to be empty, we got similar results as not having the prefixes. Our solution is to use a smaller weight on the context phrase if preceeded by an empty prefix ($E8$). With this approach, we observe very little degradation in *VS*, and also improved performance on the biasing test sets ($E8$).

Table 3: *Anti-Context Results.*

Exp ID	Model	*VS*	*Songs*	*Cnt-Real*	*Cnt-TTS*	*Apps*
E1	WPM, No Biasing	6.9	22.7	15.8	37.0	13.3
E5	+ Biasing, no prefix	12.5	5.3	7.5	7.3	2.7
E6	+ non-empty prefix	7.0	10.1	6.9	7.6	2.7
E7	+ empty prefix	12.5	5.3	7.9	7.4	2.4
E8	+ empty prefix, lower weight	7.3	5.3	6.7	6.8	2.4

4.3. Proper Noun Knowledge

In this section, we explore how we can improve biasing quality by improving the model's knowledge of proper nouns. Our baseline here is $E8$, the RNN-T wordpiece model trained on 35M supervised voice search utterances. Experiment $E9$ shows improvements across all biasing test sets when training with unsupervised data. Training with TTS data ($E10$) gives larger improvements on the TTS test sets compared to unsupervised data ($E9$), but results in a larger degradation on a real test set (*Cnt-Real*). This indicates that the improvements in TTS biasing sets are primarly coming from matched audio conditions between training and test data, rather than learning a richer vocabulary of proper nouns. Finally, fuzzing transcripts ($E11$) shows a quality degradation on all sets, with reasons still unclear to us.

We now analyze $E9$ (unsupervised data). Table 5 shows the percentage of incorrectly recognized biasing phrases in each test set (ERR), and the same metric on biasing phrases with at least one OOV word (OOV), i.e., a word not in the training data. First, the table shows the model is very likely to misrecognize OOVs without biasing. Training with unsupervised data largely improves the ERR metric without biasing, mainly because it reduces OOV rate as shown in Table 1. Second, we see that training with unsupervised data also improves the WER on OOV utterances with biasing, as it provides the model with more similar words during training, giving the model more confidence to output the correct word. Thus, models trained with unsupervised data give better results in all test sets with and without biasing.

Table 4: *Proper Noun Results.*

Exp ID	Model	*Songs*	*Cnt-Real*	*Cnt-TTS*	*Apps*
E8	Sup Data	22.7	15.8	37.0	13.3
	+ Biasing	5.3	6.7	6.8	2.4
E9	Sup + Unsup Data	14.7	15.4	25.0	9.6
	+ Biasing	3.0	5.8	5.4	1.9
E10	TTS + Biasing	4.3	7.1	1.8	1.0
E11	Fuzzing + Biasing	7.2	11.0	14.0	4.4

Table 5: *Error rate in biasing phrases, and errors due to OOV.*

Model	*Songs*		*Cnt-Real*		*Cnt-TTS*		*Apps*	
	Err	OOV	Err	OOV	Err	OOV	Err	OOV
E8	21.1	94.7	23.0	91.9	62.9	99.0	16.2	97.9
+Bias	4.5	64.2	8.0	29.4	7.0	24.4	2.0	26.1
E9	15.5	90.8	22.3	94.6	55.6	100.0	12.1	100
+Bias	3.8	60.2	5.6	19.0	4.6	20.8	1.8	15.4

4.4. Comparison To Conventional Model

Table 6 compares the biasing performance of RNN-T to a conventional model of comparable size (130MB), consisting of a CTC AM trained with context-independent phoneme targets, along with a separate PM and LM [5]. The RNN-T model outperforms the conventional model by 20%-40% relative on all categories but songs, perhaps due to the prefix set for songs not being comprehensive enough.

Table 6: *E2E vs. Conventional Model Biasing*

Model	*VS*	*Songs*	*Cnt-Real*	*Cnt-TTS*	*Apps*
RNN-T	**6.7**	3.0	**5.8**	**5.4**	**1.9**
Conventional	9.3	**2.4**	6.8	5.7	2.4

5. Conclusions

In this paper, we present various algorithmic and data improvements to shallow-fusion E2E biasing. We find that RNN-T shallow-fusion-based biasing shows competitive performance to an on-device conventional model across a variety of contextual-biasing tasks.

6. Acknowledgements

The authors would like to thank Brian Roark, Justin Scheiner, Yanzhang (Ryan) He, Golan Pundak, Ben Haynor, Rohit Prabhavalkar, Ian McGraw and Trevor Strohman for useful discussions.

7. References

[1] P. Aleksic, M. Ghodsi, A. Michaely, C. Allauzen, K. Hall, B. Roark, D. Rybach, and P. Moreno, "Bringing contextual information to Google speech recognition," in *Interspeech*, 2015.

[2] I. Williams, A. Kannan, P. Aleksic, D. Rybach, and T.N. Sainath, "Contextual speech recognition in end-to-end neural network systems using beam search," in *Interspeech*, 2018.

[3] A. Kannan, Y. Wu, P. Nguyen, T. N. Sainath, Z. Chen, and R. Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," in *ICASSP*, 2018.

[4] G. Pundak, T.N. Sainath, R. Prabhavalkar, A. Kannan, and D. Zhao, "Deep context: End-to-end contextual speech recognition," in *to appear in Proc. of SLT*, 2018.

[5] I. McGraw, R. Prabhabalkar, R. Alvarez, M. Gonzalez, K. Rao, D. Rybach, O. Alsharif, H. Sak, A. Gruenstein, F. Beaufays, and C. Parada, "Personalized speech recognition on mobile devices," in *ICASSP*, 2016.

[6] K. Hall, E. Cho, C. Allauzen, F. Beaufays, N. Coccaro, K. Nakajima, M. Riley, B. Roark, D. Rybach, and L. Zhang, "Composition-based on-the-fly rescoring for salient n-gram biasing," in *Interspeech*, 2015.

[7] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[8] D. Rybach, J. Schalkwyk, and M. Riley, "On lattice generation for large vocabulary speech recognition," in *ASRU*, 2017.

[9] L. Vasserman, B. Haynor, and P. Aleksic, "Contextual Language Model Adaptation using Dynamic Classes," *SLT*, 2016.

[10] G. Pundak and T. N. Sainath, "Lower frame rate neural network acoustic models," in *Proc. Interspeech*, 2016.

[11] Google, "Cloud natural language," `https://cloud.google.com/natural-language/`, 2018, [Online].

[12] Z. Huang and W. Xu and K. Yu, "Bidirectional LSTM-CRF Models for Sequence Tagging," *CoRR*, vol. abs/1508.01991, 2015.

[13] D. Andor, C. Alberti, D. Weiss, A. Severyn, A. Presta, K. Ganchev, S. Petrov, and M. Collins, "Globally Normalized Transition-Based Neural Networks," *CoRR*, vol. abs/1603.06042, 2016.

[14] B. Li, T.N. Sainath, R. Pang, and Z. Wu, "Semi-supervised Training for End-to-End Models Via Weak Distillation," in *Proc. ICASSP*, 2019.

[15] X. Gonzalvo, S. Tazari, C. Chan, M. Becker, A. Gutkin, and H. Silen, "Recent Advances in Google Real-time HMM-driven Unit Selection Synthesizer," in *Interspeech*, 2016.

[16] C. Kim, A. Misra, K. Chin, T. Hughes, A. Narayanan, T. N. Sainath, and M. Bacchiani, "Generated of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home," in *Proc. Interspeech*, 2017.

[17] A. van den Oord, Y. Li, and I. Babuschkin et. al., "Parallel wavenet: Fast high-fidelity speech synthesis," Tech. Rep., Google Deepmind, 2017.

[18] A. Graves, "Sequence transduction with recurrent neural networks," *CoRR*, vol. abs/1211.3711, 2012.

[19] H. Soltau, H. Liao, and H. Sak, "Reducing the Computational Complexity for Whole Word Models," in *ASRU*, 2017.

[20] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov 1997.

[21] M. Schuster and K. Nakajima, "Japanese and Korean voice search," *2012 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2012.

[22] M. Abadi et al., "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," Available online: http://download.tensorflow.org/paper/whitepaper2015.pdf, 2015.