Interspeech 2018
2-6 September 2018, Hyderabad

# Contextual Speech Recognition in End-to-End Neural Network Systems using Beam Search

*Ian Williams, Anjuli Kannan, Petar Aleksic, David Rybach, Tara N. Sainath*

Google, Inc.

{iantw, anjuli, apetar, rybach, tsainath}@google.com

## Abstract

Recent work has shown that end-to-end (E2E) speech recognition architectures such as Listen Attend and Spell (LAS) can achieve state-of-the-art quality results in LVCSR tasks. One benefit of this architecture is that it does not require a separately trained pronunciation model, language model, and acoustic model. However, this property also introduces a drawback: it is not possible to adjust language model contributions separately from the system as a whole. As a result, inclusion of dynamic, contextual information (such as nearby restaurants or upcoming events) into recognition requires a different approach from what has been applied in conventional systems.

We introduce a technique to adapt the inference process to take advantage of contextual signals by adjusting the output likelihoods of the neural network at each step in the beam search. We apply the proposed method to a LAS E2E model and show its effectiveness in experiments on a voice search task with both artificial and real contextual information. Given optimal context, our system reduces WER from 9.2% to 3.8%. The results show that this technique is effective at incorporating context into the prediction of an E2E system.

**Index Terms**: speech recognition, end-to-end, contextual speech recognition, neural network

## 1. Introduction

A contextual automatic speech recognition (ASR) system uses real-time contextual signals to dynamically adjust the priors in a pre-trained speech recognition system [1]. Contextual signals can include: a user's location, the device being used, or personalization information such as a user's favorite songs and calendar events (Figure 1). Including this information has been shown to improve recognition results [2]. Our contextual ASR system was previously built on a conventional architecture, and in this paper we propose a design to bring similar improvements to E2E architectures.


Figure 1: *Contextual ASR system and example contexts.*

Conventional ASR systems break down the recognition problem into subproblems which are independently modeled and trained, then jointly executed during recognition. For example, an acoustic model will associate raw acoustic features to phonetic units such as context dependent phonemes, a pronunciation model will map those acoustic units to words, and a language model will assign likelihoods to word sequences. Afterwards, a text normalization component may transform the spoken form of the word sequence to a written form (e.g. one pm vs. 1:00 PM or 13:00).

This relative independence of modules has the benefit of adjustability. Conventional contextual systems rely on being able to inspect and modify individual components of modular systems in order to function. For example, a standalone language model can support dynamic population of classes [3], and a standalone pronunciation model allows dynamic injection of pronunciations [4]. One drawback of this is that information consumed within one piece of the modeling may be useful elsewhere; acoustic signals could inform a language model or text normalizer.

An E2E system refers to a system in which a single component learns to associate raw acoustic data with written language without the need for independently trained components. Within the last few years, E2E (also known as sequence-to-sequence) models implemented using neural networks have become competetive with conventional systems [5]. Among these E2E models are connectionist temporal classification approaches such as RNN-T [6] or attention-based approaches, such as Listen Attend Spell [7].

In this work we focus on bringing on-the-fly rescoring [8] into the LAS implementation of an E2E system. We do not have the same amount of adjustability in the E2E paradigm, but the beam search portion of the system provides a place to implement rescoring. The beam search maintains a set of partial sequence hypotheses and decides which previous outputs to feed back into the LAS decoder to continue generating outputs [7]. Previous work has explored fusing a language model into an E2E system (as in Cold Fusion and Deep Fusion [9]). Recently, shallow fusion demonstrated using a swappable LM that can be changed per task [10]. This work is related but differs in that we do not fuse a complete LM into the system, instead we adjust the network outputs with a partial LM containing only contextual n-grams.

We created a system which takes contextual phrases for an utterance along with an amount by which to increase their likelihoods. The phrases are split into n-grams and compiled into a weighted finite state transducer (WFST), which allows fast and efficient search [8]. During beam search the WFST is traversed along with the outputs from the LAS network, and contextual rescoring is performed when a match is found.

The rest of the paper is organized as follows. Section 2 provides background on contextual speech recognition. Section 3 gives an overview of the system designs. Section 4 describes the implementation of LAS and contextual modeling in LAS.



10.21437/Interspeech.2018-2416

In section 5 we present our experimental results, and in 6 we conclude.

## 2. Contextual Speech Recognition

There are many speech recognition applications where on-the-fly adjustment is indispensable. In the voice search task, we have previously shown quality improvements by introducing n-gram weight adjustments for salient n-grams from personal contexts and geographic information [2] as well as improved recognition of contact names using context [11]. Products such as the Google Assistant use context for all types of personal entities (e.g. songs, artists), among many other applications.

Powering this functionality is a context module, which is responsible for fetching data from databases, the recognition request, and other on-line services. All of these contextual sources must respond quickly, as this process takes place as the user begins speaking. Once the contexts are collected, the context module transforms them into WFSTs which can be fed into the recognition system. In the conventional system, these can be used in several ways. In this paper we are concerned only with on-the-fly rescoring.

### 2.1. On-the-fly Rescoring

In on-the-fly rescoring, we compile the contexts into a set of n-grams, $\mathcal{B}$, for which we wish to increase likelihood. These n-grams are compiled into a WFST as described in [8].

During recognition, whenever a word boundary is reached for a word $w$, the recognition component presents the score for that word $S_w$ along with its word history $w_H$ to a rescoring function $F_R$. This function returns a rescored score $R_w$, which is used as the recognition proceeds.

$$R_w = \begin{cases} F_R(S_w, w|w_H) & \text{if } w|w_H \in \mathcal{B} \\ S_w & \text{else} \end{cases} \quad (1)$$

In previous work [2], we have used two approaches for $F_R$. The first is a log-linear interpolation between the contextual WFST score $C_w$ and base score $S_w$, using an interpolation weight $\alpha$.

$$F_R(S_w, w|w_H) = (1-\alpha)S_w + \alpha C_w(w|w_H) \quad (2)$$

The second is a likelihood "boost", obtained by applying the contextual WFST boost $C_w$ to the hypothesis score in log-space, moderated by a weight $\alpha$.

$$F_R(S_w, w|w_H) = S_w - \alpha C_w(w|w_H) \quad (3)$$

## 3. System Design

### 3.1. Conventional ASR

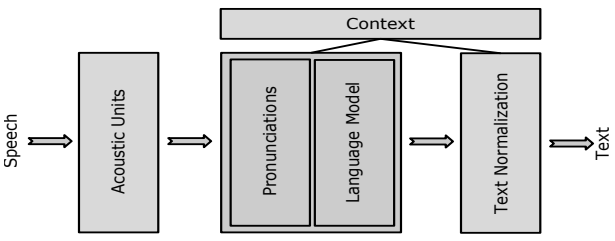


Figure 2: *Conventional contextual ASR system.*

In the conventional system, the decoder computes $F_R$ whenever a word boundary is reached in the joint pronounciation/language model graph (Figure 2). This implementation acts upon thousands of competing paths during exploration of the graph. In this implementation, on-the-fly rescoring has the broad ability to affect the search of the graph [12].

In addition to on-the-fly rescoring, we are able to dynamically modify the joint pronounciation/language model graph by adding new paths at runtime. This allows us to insert dynamic pronunciations or language model classes [3, 4]. We do not yet have a direct replacement for this additional functionality in LAS.

### 3.2. LAS Rescoring system

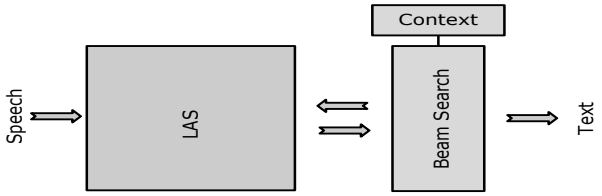


Figure 3: *LAS-based contextual ASR system.*

The LAS system is rescored in a simpler fashion. The context system still provides the WFSTs containing the n-grams of contextual relevance, however in this case they are provided to the beam search module (Figure 3). This module iteratively interacts with the LAS neural network to build hypotheses one unit at a time.

In this arragement, rescoring can only be applied on the significantly smaller number of competing hypotheses maintained in the search (typically 8).

## 4. Rescoring LAS with Context

### 4.1. The LAS Model

There are several attention-based E2E models that have been studied recently [13]. Our experiments are conducted on the LAS architecture [7], though we posit that the method should be applicable to this whole family of models.

As the name suggests, LAS consists of three components that are analogous to the three major parts of a conventional speech system. First, given a sequence of $d$-dimensional feature vectors $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T)$, where $\mathbf{x}_t \in \mathbb{R}^d$, the *encoder* produces a sequence of fixed-size feature representations, denoted $\mathbf{h}_1^{\text{enc}}, \cdots, \mathbf{h}_T^{\text{enc}}$. Typically an encoder consists of a stack of long short-term memory (LSTM) [14] layers. This "listening" component is akin to an acoustic model.

Next, at each decoding step $u$, the model "attends" to various parts of the $T$ feature representations to compose a context vector $\mathbf{c_u}$, which is similar in function to dynamic time warping.

Finally, conditional on the context vector output by the attention mechanism, as well as the previous output $y_{u-1}$, the *decoder* "spells", meaning it outputs a probability distribution $P(y_u|y_{u-1}, \mathbf{c_u})$ over output symbols. Output symbols can be graphemes, phonemes, or wordpieces; in our experiments we use graphemes.

Although the three components of the LAS model have similar roles to the parts of a conventional speech system, a key difference is that they are trained jointly as a single neural network. Typically they are trained to minimize cross entropy between the output probability distribution and the ground truth grapheme labels. The basic LAS model is shown in the dotted line box in Figure 4.

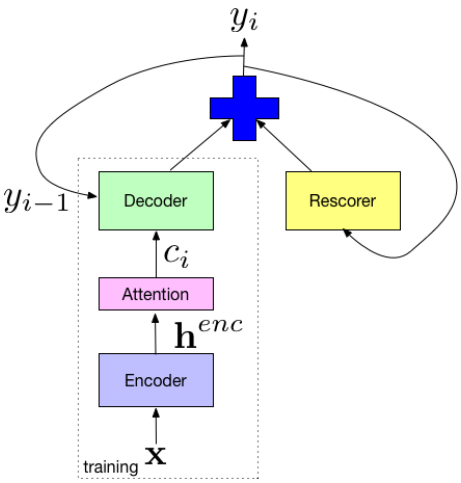


Figure 4: *LAS model with on-the-fly rescoring.*

### 4.2. Decoding LAS with Beam Search

At inference time, we must decode the model to generate a sequence of output labels. Ideally we would like to find the output sequence with maximum likelihood according to the output probability distribution, i.e.

$$\mathbf{y}^* = \underset{y}{\operatorname{argmax}}\, p(\mathbf{y}|\mathbf{x}) \tag{4}$$

The model, however, gives only step-wise distributions $P(y_u|y_{u-1}, \ldots, y_0, \mathbf{x})$. Exhaustively computing the probability of every possible output sequence would be intractable, so it is common to approximate the arg max using a beam search.

Beam search is conducted as follows. At the first decoding step, we feed a special start-of-sentence token $\langle sos \rangle$ to the decoder. The decoder then outputs a distribution $P(y_1|\langle \text{sos} \rangle, \mathbf{c_0})$ over the $G$ graphemes that comprise the output vocabulary. For each of the top $k$ most likely graphemes, we create a partial transcript and add it to our beam. In the next decoding step, we extend each of these $k$ partial transcripts by each of the $G$ graphemes, for a total of $kG$ candidates. Then we once again retain only the $k$ most likely partial transcripts from this set of extensions. If the special end-of-sentence token $\langle eos \rangle$ is encountered, the transcript is considered complete and removed from the beam. This repeats until there are no more partial transcripts.

For attention-based end-to-end speech models, good results can be achieved with a relatively small $k$, such as 8 or 16, with little benefit for higher values [7].

### 4.3. Beam Search Adaptation

To incorporate context into the LAS model, we modify the objective of the beam search. Rather than optimizing the criterion from Equation 4, we optimize the following:

$$\mathbf{y}^* = \arg\max_y \frac{p(\mathbf{y}|\mathbf{x})}{p_C(\mathbf{y})^\alpha} \tag{5}$$

where $p_C$ is a prior distribution over output sequences based on some context $C$ and $\alpha$ is a tunable parameter controlling the amount of influence of the context. In other words, we apply a small boost in likelihood for output sequences which are more likely given the context. This is similar to shallow fusion [10], except that here we selectively apply a small boost instead of including an LM score in all hypotheses. The following section will explain more about how $p_C$ is generated.

In practice, this means that we modify the score used to evaluate partial transcripts in the beam search. At every step of the beam search, we consider the top $k$ partial transcripts from the previous step. We extend each transcript by each of the $L$ graphemes, then compute the score

$$s(\mathbf{y}) = \log p(\mathbf{y}|\mathbf{x}) - \alpha \log p_C(\mathbf{y}) \tag{6}$$

for each of the $kL$ candidates. Notice this is equation (3) applied to LAS. Then we once again retain only the $k$ partial transcripts with the highest scores. $\alpha$ can be tuned on a dev set to minimize WER.

### 4.4. Encoding Context

In the previous section we assumed that we had access to a distribution $p_C$. Now we turn to the question of how this is constructed and applied in practice.

As in conventional on-the-fly rescoring, the context $n$-grams are first compiled into a WFST which we denote $G$. This allows different $n$-grams to have different weights. Note that $n$-grams are words, but the LAS model is decoded at the grapheme level. Therefore, we next compose the word-level WFST with a "speller" which spells the graphemes of each word. Following the procedure used by [15] for a general language model, we compose $G$ with a speller FST $S$ to produce $C = min(det(S \circ G))$. $C$ can then be accessed to output a log probabilty of any prefix during the beam search. Importantly, unlike [15] we do not perform weight pushing. This is because we only want to apply score boosts at word boundaries.

## 5. Experiments

We ran two sets of experiments to evaluate the relative performance of the contextual systems. In all of our tests, the data has been anonymized and does not contain any personally identifiable information. In all of our experiments we show word error rate (WER), and for some we provide the sentence accuracy (SACC).

In the first set of experiments, we test the theoretical lower limit of WER this system can achieve when it is provided optimal context. We used the n-grams from the exact transcript of the utterance as context. We refer to this experiment as the transcript truth experiment. Another variant of this experiment provides varying numbers of distractors: n-grams which do not appear in the transcipt and have nothing to do with it. When tuning a contextual system for real-world use, it is important to use rescoring weights which give good results when given good context and yet do not deteriorate in quality when given bad context.

The second set of experiments tests the system on a real contextual task. We chose utterances belonging to a dialog state which expects a very small number of possible transcripts: "yes", "no", "cancel", "send". We refer to this as the Yes/No/Cancel (YNC) test.

The baseline in our experiments is our production conventional system. It uses an LSTM to transform acoustic features into probability distributions over context-dependent phonemes, which are then matched using a WFST-based decoder to word sequences. More details can be found in our previous work on contextual systems [2, 12, 1].

The architecture used for LAS in all experiments is as follows [5]. The encoder consists of five layers of 1400 unidirec-

tional LSTM cells. The attention mechanism is a multi-headed additive attention [16] using 4 heads, with a 1,024 dimensional query vector. The decoder consists of 2 layers of 1,024 RNN cells. In the beam search we use a beam size of 8.

The network is trained to output graphemes. This includes all lowercase letters, the numbers 0-9, a space symbol, punctuation and syntactic characters (e.g. '.', '&', '%', '?'). The network outputs a special end-of-sentence token when it believes it has reached the end ("$\langle eos \rangle$").

All experiments use the same acoustic features. The encoder is provided with 80-dimensional log-mel acoustic features. These are computed over a 25ms window with a frame shift of 10ms. As discussed in [13], three consecutive frames are stacked and every third stack is given to the encoder.

### 5.1. Transcript Truth Experiments

The test set used for this experiment consists of 1,000 utterances selected from voice search traffic. When distractors are present, we used randomly selected transcripts from other utterances in the test set.

| Case | WER [%] | SACC [%] |
|---|---|---|
| Baseline | 9.6 | 76.32 |
| Baseline biased | 4.4 | 89.19 |
| LAS | 9.2 | 73.30 |
| LAS biased | 3.8 | 90.40 |

Table 1: *WER and SACC observed on the transcipt truth test set for the baseline and the LAS system, both with and without rescoring. No distractors are present.*

This table gives the best results we achieved in the no-distractor case for both systems. The baseline improves from 9.6% WER and 76.32% SACC to 4.4% WER and 89.19% SACC with rescoring. The LAS system improves from 9.2% WER and 73.3% SACC to 3.8% and 90.4% with rescoring.

In figure 5, we show how the system performs when provided distracting context. Each line corresponds to a particular configuration of rescoring parameters. The key shows what WER that configuration achieves when provided the transcript truth.

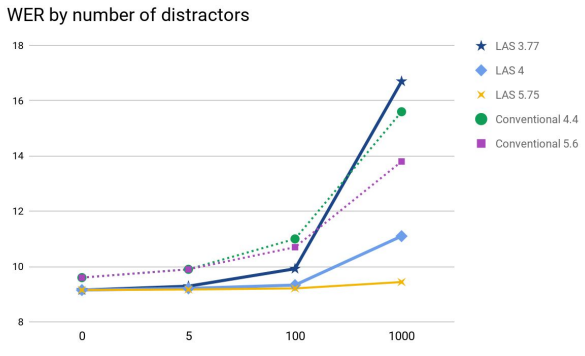


Figure 5: *Performance comparison for various configurations in LAS and conventional contextual systems.*

As expected, biasing the transcipt truth highly tends to decrease the WER quickly as the number of distractors increases. The most aggressive tune of LAS, which achieves a WER of 3.77 with transcript truth, decreases the WER to 16.7 with 1,000 distracting transcripts. However, with slightly less strong tuning the LAS system outperforms the conventional system at all points.

### 5.2. YNC Experiments

The test set used for this experiment consists of 976 real utterances where the speaker's device was in a YNC dialog state. The context provided to the rescorer is only the set of unigrams "yes", "no", "cancel", and "send". While nearly every utterance in the set contains one of these words, many of them contain a variant such as "send now" or "cancel cancel". These are very difficult-to-recognize utterances with significant noise and multiple speakers.

One challange that arose during this experiment is that the model we use is trained only on utterances consisting of more than three words. Therefore, the model has difficulty correctly terminating the short transcripts found in this dataset. This is a problem that affects the LAS system more than a conventional system because the LAS system learns its inherent LM from the training utterances, whereas a conventional system can learn its LM from any available data. To address this issue, we introduced a word insertion penalty (WIP) which adds a linear penalty to the transcript score based on length.

The following table shows that the WER for the LAS system is initially very high due to the difficulty of this test set. Our rescoring method reduced WER by 23% relative. While the WIP did little to impact the LAS system alone, it was critical to achieving results with rescoring. Even though the LAS model does worse than the baseline system, rescoring achieved a better relative WER improvement.

| Case | WER [%] | SACC [%] |
|---|---|---|
| Baseline | 12.0 | 90.1 |
| Baseline + rescore | 9.7 | 92.5 |
| LAS | 20.9 | 84.0 |
| LAS + WIP | 20.5 | 84.1 |
| LAS + WIP + bias | 15.7 | 86.1 |

Table 2: *WER and SACC observed on the YNC test set for new LAS system, both with and without rescoring. No distractors are present.*

## 6. Conclusion

We have described a technique to build a contextual speech recognition system using E2E ASR systems. We have shown that the existing work in contextual modeling can be applied very similarly in this new paradigm. The experiments show that the technique works very well and is suitable for use in the next generation of multi-modal speech recognizers; it achieves even better results than the previous system on tests designed to measure the ability to adapt to context and robustness to bad context.

In the future, we plan to introduce other contextual adaptation techniques from conventional speech recognizers to E2E models. We also plan to use this work in similar families of models, such as RNN-T.

## 7. References

[1] P. S. Aleksic, M. Ghodsi, A. Michaely, C. Allauzen, K. B. Hall, B. Roark, D. Rybach, and P. J. Moreno, "Bringing contextual information to google speech recognition." in *Proc. Interspeech*, 2015, pp. 468–472.

[2] J. Scheiner, I. Williams, and P. Aleksic, "Voice search language model adaptation using contextual information," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 253–257.

[3] L. Vasserman, B. Haynor, and P. Aleksic, "Contextual language model adaptation using dynamic classes," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 441–446.

[4] A. Patel, D. Li, E. Cho, and P. Aleksic, "Cross-lingual phoneme mapping for language robust contextual speech recognition," in *accepted to Proc. ICASSP*, 2018.

[5] C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, and M. Bacchiani, "State-of-the-art Speech Recognition With Sequence-to-Sequence Models," in *Proc. ICASSP*, 2018.

[6] A. Graves, "Sequence transduction with recurrent neural networks," in *Proc. ICASSP*, 2012.

[7] 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)*, March 2016, pp. 4960–4964.

[8] K. B. 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*, 2015.

[9] A. Sriram, H. Jun, S. Satheesh, and A. Coates, "Cold fusion: Training seq2seq models together with language models," *CoRR*, vol. abs/1708.06426, 2017. [Online]. Available: http://arxiv.org/abs/1708.06426

[10] 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 *accepted to Proc. ICASSP*, 2018.

[11] P. Aleksic, C. Allauzen, D. Elson, A. K. D. M. Casado, and P. J. Moreno, "Improved recognition of contact names in voice commands," in *ICASSP 2015*, 2015.

[12] I. Williams and P. Aleksic, "Rescoring-aware beam search for reduced search errors in contextual automatic speech recognition," in *Proc. Interspeech*, 08 2017, pp. 508–512.

[13] R. Prabhavalkar, K. Rao, B. Li, L. Johnson, and N. Jaitly, "A Comparison of Sequence-to-sequence Models for Speech Recognition," in *Proc. Interspeech*, 2017.

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

[15] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-End Attention-based Large Vocabulary Speech Recognition," in *Proc. ICASSP*, 2016.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. 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. Curran Associates, Inc., 2017, pp. 5998–6008. [Online]. Available: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf