Interspeech 2024
1-5 September 2024, Kos, Greece

Improving Neural Biasing for Contextual Speech Recognition by Early Context Injection and Text Perturbation

Ruizhe Huang[1], *Mahsa Yarmohammadi*[1], *Sanjeev Khudanpur*[1], *Daniel Povey*[2]

[1]Johns Hopkins University, USA
[2]Xiaomi Corp., China

{ruizhe,mahsa,khudanpur}@jhu.edu

Abstract

Existing research suggests that automatic speech recognition (ASR) models can benefit from additional contexts (e.g., contact lists, user specified vocabulary). Rare words and named entities can be better recognized with contexts. In this work, we propose two simple yet effective techniques to improve context-aware ASR models. First, we inject contexts into the encoders at an early stage instead of merely at their last layers. Second, to enforce the model to leverage the contexts during training, we perturb the reference transcription with alternative spellings so that the model learns to rely on the contexts to make correct predictions. On LibriSpeech, our techniques together reduce the rare word error rate by 60% and 25% relatively compared to no biasing and shallow fusion, making the new state-of-the-art performance. On SPGISpeech and a real-world dataset ConEC, our techniques also yield good improvements over the baselines.
Index Terms: speech recognition, contextual biasing, data augmentation

1. Introduction

Human speech recognition does not occur in isolation. In addition to acoustic cues, we often rely on various contextual resources, such as semantic or visual context or speaker's background knowledge, to aid in understanding and interpreting spoken content. In particular, these contextual cues play a significant role in recognizing rare words and named entities. End-to-end (E2E) automatic speech recognition (ASR) has emerged as the dominant solution of ASR, due to its simplicity of modeling and impressive performance. However, a conventional E2E ASR model takes merely acoustics features as input and outputs the corresponding text transcription.

Recently, various contextual biasing techniques (contextual ASR) have been proposed to improve standard E2E ASR models, including [1, 2, 3] for connectionist temporal classification (CTC) models, [4, 5, 6, 7] for attention-based encoder-decoder (LAS) models, [8, 9, 10, 11, 12, 13, 14, 15, 16, 17] for transducer models and more recently [18, 19, 20, 21] for (or with) large language models (LLMs). Following the majority of prior research, we define the context to be lists of biasing words, which are usually rare words in the model's training data. Other types of context, such as visual contexts [22], date-time and location [23] are out of scope of this paper.

In general, contextual ASR can be achieved either in a shallow way or deep way (or a hybrid of the two). In shallow biasing, the internal representations of the E2E models are unchanged. The contextual biasing most likely happens only during the decoding process, where the contexts are used to guide the beam search, e.g., shallow fusion [1, 24, 25, 26, 27], spelling correction [28]. Shallow fusion is considered as a simple yet robust baseline, as it can be easily integrated into beam search to provide moderate improvement in recognizing rare words. In deep (neural) biasing, the context is injected into the E2E model to edit the internal representations of the models and make a potentially new output distribution, e.g., with cross-attention over the biasing lists [4, 11]. Neural biasing has been reported outperforming shallow fusion [2, 4, 9, 10, 11, 12], due to the specialized parameters trained to accommodate contexts. However, many existing work [7, 13, 14, 15, 16, 17] do not directly compare their neural biasing approaches with shallow fusion. They only report the gains over non-contextual ASR baselines.

This paper proposes two techniques that can improve neural biasing across various ASR models. First, we inject contexts into earlier layers of the encoder as opposed to only the last layer. Although this idea has been explored in some existing work [29, 15], the specific layer and the number of layers to be integrated with contexts remain unclear. Other work [2, 14] use the outputs from intermediate encoder layers as the queries for contexts lookup, but the resulting contextual embedding is still integrated with the encoder's final output at the last layer. While some work [3, 9] raise concerns about runtime latency associated with neural biasing, we report decoding runtime in our experiments and find that the overhead of early context injection is negligible when the biasing lists are of size 500. For larger biasing lists, light-weight algorithms may be applied to shorten the biasing lists, which can be a future work.

Secondly, during training, we propose to perturb the reference transcription with alternative, similar-sounding spellings of the rare words. For example, we opt to replace the word "Klein" with a random alternative spelling "Klane" in both the transcription and contexts. Hence, the end-to-end trained model is forced to rely on the contexts to make correct predictions. Alternative spellings has been explored in [1, 3, 9, 10, 26, 30, 31, 32]. Among them, [1, 3, 26, 30, 31] use alternative spelling during decoding to improve the recall of rare or out-of-vocabulary words. Closely related to our work are [9, 10, 32] where the alternative spellings are used as data augmentation during training. However, their neural biasing architecture is very different from ours. [32] is based on CLAS architecture [4] and tries to better distinguish phonetically confusable phrases. [9, 10] use a contextual predictor ("PLM") which is implemented by a prefix tree, instead of cross attention mechanism, in their transducer model. Thus, their encoders are not context-aware, although transducers are "encoder-heavy" models. Note that [1, 3, 26, 30, 31, 32] also propose algorithms or models to generate alternative spellings, e.g., a grapheme-to-grapheme (G2G) model [26], which may further benefit our approach. In this work, we simply use less than 200 hand-crafted linguistic rules (e.g., "ein"↔"ane", "s"↔"z") that cover all 26 English letters and show this already goes a long way.

10.21437/Interspeech.2024-749

Despite the simplicity of our techniques, we achieve the new state-of-the-art results on LibriSpeech [33] in the contextual ASR setup [10]. Furthermore, we demonstrate promising improvements over shallow fusion with neural biasing on two public datasets, SPGISpeech [34] and ConEC [35], where the latter uses real-world contexts rather than synthesized contexts from the ground truth. Our implementation and experiment results are available in the ConEC repository[1].

2. Contextual ASR

In this section, we review conventional, non-contextual ASR models, taking transducers as an example. Then we describe the integration of neural contextual biasing by cross attention mechanism to the transducer models.

2.1. Transducer ASR Model

Transducer model is first proposed in [36] to learn the transformation between sequences, e.g., from speech to text transcription. Formally, it learns the probability $p(\mathbf{W}|\mathbf{X})$ of word (or word piece) sequence $\mathbf{W} = (w_1, w_2, ..., w_U)$ of length U, given a speech feature sequence $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)$ of length T. Transducer model has been widely used in ASR due to its effectiveness and streaming nature. It has three components.

Encoder: The encoder serves the role of an acoustic model. It takes the feature sequence $\mathbf{X}$ as input, and transforms it to a sequence $\mathbf{H}^{enc} = (\mathbf{h}_1^{enc}, \mathbf{h}_2^{enc}, ..., \mathbf{h}_T^{enc})$ of high-level representations of the input:

$$\mathbf{H}^{enc} = f^{enc}(\mathbf{X}) \tag{1}$$

There can be many layers of the transformation. The output from the i-th layer f_i^{enc} is denoted as $\mathbf{H}_i^{enc}$, and the final output is $\mathbf{H}^{enc}$. Note, there can be downsampling along the time axis, which is omitted here for simplicity. In practice, the encoder is implemented by a recurrent neural network (RNN) or more recently Conformer [37] architecture. Encoder can take up most parameters (e.g., more than 90%) in the transducer model.

Predictor. Given $\mathbf{H}^{enc}$ from the encoder, one may produce a frame-wise softmax distribution over the output word pieces vocabulary, which is the idea of a CTC model. The transducer model, on the other hand, explicitly imposes dependencies between the output word pieces in $\mathbf{W}$ by the predictor. It acts like a language model:

$$\mathbf{h}_{u-1}^{pred} = f^{pred}(w_1, w_2, ..., w_{u-1}) \tag{2}$$

Joiner. The joiner takes the embeddings $\mathbf{h}_t^{enc}$ and $\mathbf{h}_u^{pred}$ from both the encoder and the predictor to produce a softmax probability distribution over the output word pieces vocabulary:

$$p(w_{t,u}|\mathbf{X}) = \text{Softmax}(f^{join}(\mathbf{h}_t^{enc}, \mathbf{h}_{u-1}^{pred})) \tag{3}$$

Note that the output from the joiner is of the shape (T, U, V) for a single input sequence, where V is the size of the word pieces vocabulary. From this output, the probability of the ground-truth transcription $\bar{\mathbf{W}}$ can be computed efficiently via dynamic programming, which will be maximized during training.

2.2. Neural Biasing with Cross Attention

CLAS [4] was first proposed to bias the attention-based encoder-decoder (LAS) models with cross attention. Later, [8, 11, 12] applied cross attention to bias transducer models, which has become a popular solution recently. In general, contextual ASR models learn to predict the probability $p(\mathbf{W}|\mathbf{X}, \mathbf{C})$ of word sequences given both acoustic feature sequence $\mathbf{X}$ and some context $\mathbf{C}$. In this work, we consider context for each utterance as a list of biasing words or phrases $\mathbf{C} = \{\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N\}$. Note, $\mathbf{c}_0$ is a special entry corresponding to the *no-bias* option.

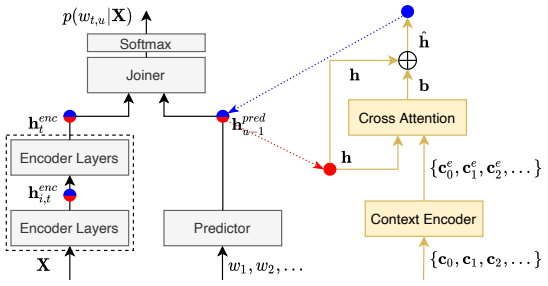

Figure 1: *The transducer model (left) and its contextual biasing module (right). The red/blue dots mark the locations where the contexts can be injected to the main model, by plugging in the contextual biasing module. During training, the gray modules can be frozen, while only the yellow modules are trained.*

Intuitively, when we predict the word piece at the t-th frame, we expect the relevant entries in the biasing list get sufficient attention. To do this, one can first independently represent each word or phrase (of various number of characters) $\mathbf{c}_j \in \mathbf{C}$ by a fixed-dimensional embedding $\mathbf{c}_j^e \in \mathbf{C}^e$ of D dimensions. An LSTM **context encoder** can be employed to accomplish this:

$$\mathbf{c}_j^e = \text{BiLSTM}(\mathbf{c}_j) \tag{4}$$

Note $\mathbf{c}_0^e$ is hard-wired to all zeros.

Then, for each frame $\mathbf{h}_t^{enc}$ from the encoder output — it can also be the output from the i-th encoder layer, we omit the subscript i for simplicity — or for each frame $\mathbf{h}_{u-1}^{pred}$ from the predictor's embedding, $\mathbf{h}_t^{enc}$ or $\mathbf{h}_{u-1}^{pred}$ is used as the query to attend to the embedded biasing list, whose keys and values are both $\mathbf{c}_j^e \in \mathbf{C}^e$. This makes the cross-attention **contextual biasing adapter**:

$$\mathbf{A}^{enc} = \text{MHA}(\text{q=}\mathbf{H}^{enc}, \text{k=}\mathbf{C}^e, \text{v=}\mathbf{C}^e) \tag{5}$$

where $\mathbf{A}^{enc}$, $\mathbf{H}^{enc}$, $\mathbf{C}^e$ are matrices of shape $T \times N$, $T \times D$ and $N \times D$ respectively (additional projection layers are omitted here). For the t-th frame, $\mathbf{a}_t^{enc} \in \mathbf{A}_t^{enc}$ defines an attention weights distribution over the N biasing words. Their weighted sum is the attention output: $\mathbf{b}_t^{enc} = \mathbf{a}_t^{enc\top} \cdot \mathbf{C}^e$.

Finally, the attention output $\mathbf{b}_t^{enc}$ is used to "edit" the encoder's (or decoder's) embedding. This can be implemented via element-wise addition:

$$\hat{\mathbf{h}}_t^{enc} = \mathbf{h}_t^{enc} + \mathbf{b}_t^{enc} \tag{6}$$

Thus, the internal representations of the transducer model become context-aware. Modified embeddings $\hat{\mathbf{h}}_t^{enc}$ and $\hat{\mathbf{h}}_{u-1}^{pred}$ are fed to the next encoder layers or the joiner to make context-aware predictions. This is illustrated in Figure 1.

The above context encoder and the cross-attention biasing adapter can be trained in an end-to-end fashion together with the transducer model. In [8, 11], all parameters are trained from scratch. In [12], the transducer's parameters are pretrained and frozen. Only a small number of the rest of the parameters (yellow modules in Fig. 1) are updated. The benefit is that the neural biasing maintains the performance of the pretrained model when no context is provided. We follow [12] in this paper.

[1] https://github.com/huangruizhe/ConEC

3. Proposed Approaches

3.1. Early Context Injection

In [2, 4, 5, 8, 11, 12, 14, 17], cross-attention neural biasing architectures are used as described in section 2.2. All of them inject contexts only into the encoder's *final* output. The drawback is that the contextualization of a frame $\hat{\mathbf{h}}_t^{enc}$ has no impact on the other frames, as they are edited essentially independently from one another. On the other hand, if contexts are injected to earlier encoder layers (e.g., at $\mathbf{h}_{i,t}^{enc}$ in Figure 1), it may have far-reaching impacts on the model's internal states via its self-attention mechanism.

One might argue that cross-attention context lookup can introduce significant computation overhead, hence they perform it only once at the last encoder layer. However, as we will show later in Section 4, this overhead is negligible. In fact, context injection can be viewed as simply adding an extra layer to the encoder. For ASR, the input or intermediate sequences usually have several hundred frames. Thus, each layer of the encoder, e.g., a conformer [37], needs to do self-attention over several hundred items. This is a comparable computation to contexts lookup if we have several hundred biasing words to attend to.

The predictor of the transducer is usually implemented by a simple LSTM network or even a stateless n-gram feed-forward network [38]. Thus, we only bias the predictor's final output.

3.2. Text Perturbation with Alternative Spellings

When transcribing unfamiliar names of people, locations or products, humans tend to choose the most familiar or phonetically similar option. Given a reference guide, they will likely refer to it for guidance. Moreover, when the contents in the reference guide change, they adapt accordingly. Text perturbation follows this idea to train the model to optimally use the contextual information. While acoustic data augmentation (e.g., [39]) is a common practice for ASR, text perturbation has not been widely adopted yet.

On the other hand, we observe that ASR models can overfit the training data. The word error rates on the training data is so low that the end-to-end training of neural biasing modules may not have enough chances to learn to attend to the contexts. On LibriSpeech and SPGISpeech (Section 4.1), the training data distribution and word error rates are listed in Table 1. Even without contexts, there is only 29.38% and 4.42% utterances in the training data containing at least one mis-recognized rare word (defined in Section 4.1). This implies that *only* these utterances may potentially benefit from the contexts containing ground-truth rare words during training. After random text perturbation (details can be found in Section 4.1), the percentage of such mis-recognized utterances containing rare words increases to 90.13% and 36.37% of all utterances.

Table 1: *Librispeech / SPGISpeech training data distributions*

	LibriSpeech	**SPGISpeech**
Duration	1000 hours	5000 hours
# Words	28,210,665	141,450,888
% Rare words (RW)	10.09%	6.12%
Training WER (common/rare)	5.77 % (5.12%/11.57%)	1.39% (1.27%/3.09%)
Avg utterance length	33.5 words	24.0 words
% Utterances:		
• with RW	91.24%	70.26%
• with mis-recog RW	29.38%	4.42%
• with mis-recog RW (after perturbation)	90.13%	36.37%

In our experiments, we apply hand-crafted linguistic rules[2] to obtain similar-sounding spelling alternatives. We replace the "maximal" matched pattern with its counterpart (e.g., "lee" → "li" although pattern "e"↔"a" is also a match). For some languages, e.g., Chinese, it is even easier to obtain spelling alternatives by pronunciation dictionary lookup for characters.

4. Experiments

4.1. Datasets

We use LibriSpeech [33] and SPGISpeech [34] to train contextual ASR models. The orthographically normalized version of SPGISpeech is used. Dataset statistics can be found in Table 1. We follow the same setup as in [10, 17] to generate artificial biasing lists during training. More specifically, we define rare words to be the words beyond top 5k/3k most frequent words for LibriSpeech/SPGISpeech. Due to the long tail nature, these words has poor ASR performance compared to the common words (Table 2 row 1). Thus, we provide the ASR model with such words, as well as some distractors, as the context, aiming to enhance the model's ability to recognize these words. For each utterance, we include all rare words from its reference transcription into its context. We also add 100 distractors sampled from all the rare words in the training vocabulary. We observe no gains when adding more distractors.

For evaluation, we use the test sets in LibriSpeech and SPGISpeech. The LibriSpeech contexts are predefined in [10], and we define the SPGISpeech contexts similary. We also use ConEC [35], which consists of earnings calls in Earnings-21 [40] and their real-world supplementary materials including presentation slides, earnings news release, a list of meeting participants' names and affiliations. We follow the evaluation metrics in [10, 17] to compute overall word error rate (WER) and the WER for common words and rare words (U-WER, B-WER). For ConEC, we also report the WERs for named entities.

For text perturbation, we randomly perturb the spellings of rare words with a given probability (0.2 for both datasets). For SPGISpeech, we also randomly (with probability of 0.8) discard utterances containing no rare words during training.

4.2. Model

We use stateless transducer [38] with Zipformer [41] encoder. The model has 15 encoder layers and 71.5M parameters in total. The transducers are trained on LibriSpeech or SPGISpeech. Then, we freeze their parameters. We use a BiLSTM context encoder of two layers and 128-dim hidden states, which is shared across all contextual biasing modules. We do not notice improvements using separate context encoders or using BERT [42] as the encoder. Each biasing adapter is implemented by a 4-head, 128-dim multi-head dot product attention layer, as well as necessary projection layers. Overall, the contextual biasing modules account for 3.7%–6.7% of the parameters compared to the transducer model. Our main contextual ASR model injects contexts at both the 9th and 15th (the last) encoder layers. The model is optimized with ScaledAdam [41] for 30 epochs.

Table 2: *Contextual ASR on LibriSpeech. Each cell is formatted as WER (U-WER / B-WER). N is the number of distractors added to the biasing words list. NO (no biasing), SF (shallow fusion), NB (neural biasing), TP (text perturbation), "@Layers: 9,15" means to which layers contexts are injected. †: This row shares the same results as there is no context for "no biasing", so N is irrelevant.*

	N=100		N=500		N=1000	
	test-clean	test-other	test-clean	test-other	test-clean	test-other
NO	†	†	2.17 (1.25/9.65)	5.22 (3.32/21.83)	†	†
SF	1.49 (1.18/3.98)	4.01 (3.27/10.50)	1.59 (1.26/4.22)	4.11 (3.32/11.05)	**1.63** (1.31/**4.27**)	4.26 (3.47/11.21)
PromptASR [15]	1.73 (- / -)	4.07 (- / -)	2.0 (- / -)	4.45 (- / -)	2.13 (- / -)	4.67 (- / -)
Guided attn [17]	2.2 (1.8/5.1)	5.4 (4.7/12.2)	n/a	n/a	2.4 (1.9/6.4)	6.0 (5.0/15.3)
NB	1.72 (1.17/6.03)	4.13 (3.18/12.47)	1.72 (**1.19**/6.06)	4.33 (3.24/13.85)	1.78 (**1.19**/6.60)	4.34 (**3.19**/14.41)
+ @Layers: 9,15	1.52 (1.13/4.65)	3.69 (**3.06**/9.16)	1.71 (1.21/5.75)	4.00 (**3.16**/11.44)	1.86 (1.28/6.58)	4.38 (3.35/13.46)
+ TP	**1.24** (**1.11/2.29**)	**3.32** (3.08/**5.46**)	**1.50** (1.24/**3.56**)	**3.69** (3.18/**8.19**)	1.74 (1.33/5.10)	**4.24** (3.49/**10.84**)

4.3. Results

The WER results for LibriSpeech are reported in Table 2. When there is no context or biasing, the rare word WER can be as high as 21.83%. Shallow fusion with biasing words provides significant improvement on this dataset, which is nearly 50% relative WER reduction. Note that shallow fusion is insensitive to the size N of the biasing words lists. The neural biasing results from two recently published papers [15, 17] and our vanilla neural biasing implementation do not outperform our shallow fusion baseline, even though their "no biasing" results (not shown here) are very close to ours. When contexts are injected to both the 9th and 15th layers, we see improvements over the vanilla neural biasing. When we further perturb the reference transcriptions, we achieve the best neural biasing results, with 60% and 25% B-WER reduction over no biasing and shallow fusion, without degrading U-WER. Also note that neural biasing is more sensitive to the biasing list size N compared to shallow fusion.

On SPGISpeech (Table 3), with a test set of 100 hours long, our techniques again outperform the baselines. On ConEC, our methods surpass the baseline except for one entity class.

Table 3: *Contextual ASR on SPGISpeech and ConEC*

SPGI	**N=100**			**N=500**	
NO	2.10 (1.81/6.60)				
SF	1.85 (1.73/3.57)			1.85 (1.74/3.61)	
NB	1.78 (1.68/3.24)			1.82 (1.69/3.76)	
+9,15,TP	**1.71 (1.65/2.56)**			**1.79 (1.68/3.35)**	
ConEC	**WER**	**PERSON**	**PRODUCT**	**ORG**	
NO	10.41 (8.71/26.02)	45.9	24.25	29.54	
SF	**10.29** (**8.70**/24.84)	39.82	**21.86**	26.09	
NB	10.66 (8.93/26.44)	41.38	24.85	27.46	
+9,15,TP	10.40 (8.76/**24.61**)	**35.72**	25.48	**25.70**	

Next, we explore which layers are optimal for integrating the contexts (Table 4). Text perturbation is disabled here. It appears that the combination of layers 9 and 15 yields the best performance. We also measure the wall-clock time to decode LibriSpeech *test-other* (of 5.3 hours) with a beam size of 4 for beam search. It takes about 3.5 minutes on one NVIDIA Tesla V100 GPU, even when we inject contexts to 4 encoder layers. As a reference, it takes 2.6 minutes for a non-contextual transducer model to decode the same data.

Finally, we search for the best probability for perturbing each rare word (Table 5). When the probability takes 0.2, the contextual ASR model has a balanced WER performance for common and rare words.

[2] https://gist.github.com/huangruizhe/dd75cf44bde12751500b8c43c73f3f22

Table 4: *Context injection to different encoder layers on LibriSpeech ($N = 500$)*

@Layers	**WER (test-other)**	**Runtime (min)**
NO	5.22 (3.32/21.83)	2.6
15 (the last)	4.33 (3.24/13.85)	3.2
9	4.25 (3.25/13.03)	3.3
6, 15	4.19 (3.24/12.55)	3.5
9, 15	**4.00 (3.16/11.44)**	3.4
11, 15	4.14 (3.24/12.09)	3.3
6, 9, 11, 15	4.19 (3.23/12.63)	3.6

Table 5: *The impact of the probability for text perturbation on LibriSpeech ($N = 500$, @Layers: 9,15)*

	test-clean	**test-other**
NO	2.17 (1.25/9.65)	5.22 (3.32/21.83)
0.1	1.60 (1.27/4.25)	3.89 (3.38/8.34)
0.2	**1.50 (1.24/3.56)**	**3.69** (**3.18**/8.19)
0.4	1.65 (1.40/3.66)	3.81 (3.45/**6.99**)

5. Conclusion

In this paper, we apply two techniques to improve cross attention based neural biasing for contextual ASR. First, we inject contexts into intermediate encoder layers in addition to the last layer. Second, during training, we replace the rare words with their similar-sounding alternative spellings in both the reference transcription and contexts. The techniques yield significant improvement in recognizing rare words and named entities on three datasets, including a real-world contextual ASR test set. Future work may explore using advanced alternative spellings generators, shortening the biasing lists or reducing the sensitivity of contextual ASR models to the distractors.

6. Acknowledgements

The authors would like to acknowledge the helpful discussion and support from Jing Liu, Mingzhi Yu, Grant Strimel, and Ariya Rastrow. This work was supported by a fellowship from JHU + Amazon Initiative for Interactive AI (AI2AI).

7. References

[1] J. D. Fox and N. Delworth, "Improving contextual recognition of rare words with an alternate spelling prediction model," *INTERSPEECH*, 2022.

[2] S. Dingliwal, M. Sunkara, S. Ronanki, J. J. Farris, K. Kirchhoff *et al.*, "Personalization of CTC speech recognition models," *SLT*, 2023.

[3] Z. Lei, E. Pusateri, S. Han, L. Liu, M. Xu *et al.*, "Personalization of CTC-based end-to-end speech recognition using pronunciation-driven subword tokenization," *ICASSP*, 2024.

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

[5] A. Bruguier, R. Prabhavalkar, G. Pundak, and T. N. Sainath, "Phoebe: Pronunciation-aware contextualization for end-to-end speech recognition," *ICASSP*, 2019.

[6] C. Huber, J. Hussain, S. Stüker, and A. H. Waibel, "Instant one-shot word-learning for context-specific neural sequence-to-sequence speech recognition," *ASRU*, 2021.

[7] Z. Zhang and P. Zhou, "End-to-end contextual asr based on posterior distribution adaptation for hybrid ctc/attention system," *ArXiv*, vol. abs/2202.09003, 2022.

[8] M. Jain, G. Keren, J. Mahadeokar, and Y. Saraf, "Contextual RNN-T for open domain ASR," *INTERSPEECH*, 2020.

[9] D. Le, G. Keren, J. Chan, J. Mahadeokar, C. Fuegen *et al.*, "Deep shallow fusion for RNN-T personalization," *SLT*, 2020.

[10] D. Le, M. Jain, G. Keren, S. Kim, Y. Shi *et al.*, "Contextualized streaming end-to-end speech recognition with trie-based deep biasing and shallow fusion," *INTERSPEECH*, 2021.

[11] F.-J. Chang, J. Liu, M. H. Radfar, A. Mouchtaris, M. Omologo *et al.*, "Context-aware transformer transducer for speech recognition," *ASRU*, 2021.

[12] K. M. Sathyendra, T. Muniyappa, F.-J. Chang, J. Liu, J. Su *et al.*, "Contextual adapters for personalized speech recognition in neural transducers," *ICASSP*, 2022.

[13] G. Sun, C. Zhang, and P. C. Woodland, "Minimising biasing word errors for contextual ASR with the tree-constrained pointer generator," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, 2022.

[14] R. Pandey, R. Ren, Q. Luo, J. Liu, A. Rastrow *et al.*, "Procter: Pronunciation-aware contextual adapter for personalized speech recognition in neural transducers," *ICASSP*, 2023.

[15] X. Yang, W. Kang, Z. Yao, Y. Yang, L. Guo *et al.*, "PromptASR for contextualized ASR with controllable style," *ICASSP*, 2024.

[16] H. Futami, E. Tsunoo, Y. Kashiwagi, H. Ogawa, S. Arora *et al.*, "Phoneme-aware encoding for prefix-tree-based contextual ASR," *ICASSP*, 2024.

[17] J. Tang, K. Kim, S. Shon, F. Wu, P. Sridhar *et al.*, "Improving ASR contextual biasing with guided attention," *ICASSP*, 2024.

[18] Z. Chen, H. Huang, A. Andrusenko, O. Hrinchuk, K. C. Puvvada *et al.*, "SALM: Speech-augmented language model with in-context learning for speech recognition and translation," *ICASSP*, 2024.

[19] C. Sun, Z. Ahmed, Y. Ma, Z. Liu, Y. Pang *et al.*, "Contextual biasing of named-entities with large language models," *ICASSP*, 2024.

[20] E. Lakomkin, C. Wu, Y. Fathullah, O. Kalinli, M. L. Seltzer *et al.*, "End-to-end speech recognition contextualization with large language models," *ICASSP*, 2024.

[21] K. Everson, Y. Gu, C.-H. H. Yang, P. G. Shivakumar, G.-T. Lin *et al.*, "Towards ASR robust spoken language understanding through in-context learning with word confusion networks," *ICASSP*, 2024.

[22] P. Pramanick and C. Sarkar, "Can visual context improve automatic speech recognition for an embodied agent?" *EMNLP*, 2022.

[23] S. N. Ray, S. Mitra, R. Bilgi, and S. Garimella, "Improving RNN-T ASR performance with date-time and location awareness," *ArXiv*, vol. abs/2106.06183, 2021.

[24] D. Zhao, T. N. Sainath, D. Rybach, P. Rondon, D. Bhatia *et al.*, "Shallow-fusion end-to-end contextual biasing," *INTERSPEECH*, 2019.

[25] W. Wang, Z. Wu, D. Caseiro, T. Munkhdalai, K. C. Sim *et al.*, "Contextual biasing with the Knuth-Morris-Pratt matching algorithm," *ArXiv*, vol. abs/2310.00178, 2023.

[26] D. Le, T. Koehler, C. Fuegen, and M. L. Seltzer, "G2G: TTS-driven pronunciation learning for graphemic hybrid ASR," *ICASSP*, 2019.

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

[28] X. Wang, Y. Liu, J. Li, V. Miljanic, S. Zhao *et al.*, "Towards contextual spelling correction for customization of end-to-end speech recognition systems," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, 2022.

[29] Z. Wu, T. Munkhdalai, P. Rondon, G. Pundak, K. C. Sim *et al.*, "Dual-Mode NAM: Effective top-k context injection for end-to-end ASR," *INTERSPEECH*, 2023.

[30] R. Huang, O. Abdel-Hamid, X. Li, and G. Evermann, "Class LM and word mapping for contextual biasing in end-to-end ASR," in *INTERSPEECH*, 2020.

[31] G. Chen, O. Yilmaz, J. Trmal, D. Povey, and S. Khudanpur, "Using proxies for oov keywords in the keyword search task," *ASRU*, 2013.

[32] U. Alon, G. Pundak, and T. N. Sainath, "Contextual speech recognition with difficult negative training examples," *ICASSP*, 2019.

[33] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "LibriSpeech: An ASR corpus based on public domain audio books," *ICASSP*, 2015.

[34] P. K. O'Neill, V. Lavrukhin, S. Majumdar, V. Noroozi, Y. Zhang *et al.*, "SPGISpeech: 5,000 hours of transcribed financial audio for fully formatted end-to-end speech recognition," *INTERSPEECH*, 2021.

[35] R. Huang, M. Yarmohammadi, J. Trmal, J. Liu, D. Raj *et al.*, "ConEC: Earnings call dataset with real-world contexts for benchmarking contextual speech recognition," *LREC*, 2024.

[36] A. Graves, "Sequence transduction with recurrent neural networks," *ICML Workshop on Representation Learning*, 2012.

[37] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *INTERSPEECH*, 2020.

[38] M. R. Ghodsi, X. Liu, J. A. Apfel, R. Cabrera, and E. Weinstein, "Rnn-transducer with stateless prediction network," *ICASSP*, 2020.

[39] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph *et al.*, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *INTERSPEECH*, 2019.

[40] M. Rio, N. Delworth, R. Westerman, M. Huang, N. Bhandari *et al.*, "Earnings-21: A practical benchmark for ASR in the wild," *INTERSPEECH*, 2021.

[41] Z. Yao, L. Guo, X. Yang, W. Kang, F. Kuang *et al.*, "Zipformer: A faster and better encoder for automatic speech recognition," *ICLR*, 2024.

[42] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL*, 2019.