

BALEEN: BIASING WITH LATENT ENCODED ENTITIES FOR CONTEXT-AWARE ASR*Chihiro Taguchi*^{**}*Yotaro Kubo*[†]*Rujikorn Charakorn*[†]^{*} University of Notre Dame, Department of Computer Science and Engineering, IN, USA[†] Sakana AI, Tokyo, Japan**ABSTRACT**

Transcribing domain-specific entities and rare proper nouns remains a major challenge in automatic speech recognition (ASR). In this paper, we propose **BaLEEN** (Biasing with Latent Encoded Entities), a lightweight, hypernetwork-based framework for dynamic contextual adaptation without fine-tuning the underlying ASR model. BaLEEN encodes variable-length contextual keywords using a pretrained language model, compresses them into a fixed sequence of latent vectors via a Perceiver bottleneck, and injects context-dependent bias vectors directly into the intermediate encoder representations of the ASR model. Because both the language model and the backbone ASR model remain entirely frozen during training, BaLEEN operates as a plug-and-play adapter that incurs zero computational overhead at inference time when context biases are precomputed. We evaluate our method on a CTC-based ASR model using a Wikipedia-derived corpus with annotated named entities and synthetic speech. Experimental results demonstrate that BaLEEN reduces keyword miss rate by 8.7% on the test set relative to the unbiased baseline while simultaneously improving overall word error rate by 21% and character error rate by 28%.

Index Terms— automatic speech recognition, contextual biasing, hypernetworks, Perceiver, CTC models

1. INTRODUCTION

Context-awareness in automatic speech recognition (ASR) remains a central challenge in the field, even as ASR technology has been widely adopted in practical applications. To achieve higher recognition accuracy and tailored user experiences, the model must transcribe speech by appropriately accounting for context in which they are deployed. For example, in mobile ASR, the model should recognize proper nouns such as the contact names stored on the device. Similarly, an ASR model for specific business domains should accurately identify specialized terminology and organization names. Because the spelling of these named entities can be highly arbitrary and irregular, modern ASR architectures require access to external contextual information for transcribing them accurately.

Since many modern ASR models rely on attention mechanisms, several proposed extensions inject contextual phrases directly into a cross-attention module [1, 2]. Another common approach integrates an external language model to adjust token probability distributions [3]. However, both types of the extensions incur additional computational cost and search latency at inference time, with cross-attention scaling linearly with the number of contextual keywords.

Alternatively, fine-tuning can adapt a model to a specific target domain if sufficient domain-specific audio-text pairs are available. For instance, LoRA [4] is a widely adopted parameter-efficient

method for domain adaptation that only trains lightweight adapter matrices while keeping the backbone model frozen. However, acquiring sufficient data and training a dedicated model for every domain with arbitrary context poses severe practical limitations due to high costs.

In this paper, we propose **BaLEEN** (Biasing with Latent Encoded Entities), an adaptation method that relaxes training-data constraints without modifying or updating the target ASR model’s parameters. Our approach utilizes a hypernetwork architecture [5], a neural network designed to generate parameters for another network. Specifically, we construct a hypernetwork that converts context embeddings into layer-wise offset parameters for the ASR model. By integrating these generated parameters directly into hidden representations, one can obtain a task-specialized model instantly, eliminating the need to fine-tune or retrain the backbone ASR architecture.

To train and evaluate hypernetwork-based biasing under rich domain diversity, we construct a synthetic dataset comprising 144k audio samples derived from English Wikipedia articles using LLM entity extraction and neural text-to-speech (TTS) synthesis. Using this dataset, we train a Perceiver-based hypernetwork [6] to generate bias vectors injected to the English ParakeetCTC baseline. When contextual bias vectors are precomputed, our method incurs virtually zero computational overhead at inference time. Experimental results demonstrate that BaLEEN reduces the keyword miss rate (KMR) by 14.3% relative to the unbiased baseline while simultaneously improving overall word and character error rates. Our code, dataset, and trained models will be made publicly available.

2. RELATED WORK

Prior adaptation methods can be broadly categorized into three paradigms: decoding-time biasing, training a separate bias encoder module, and biasing the decoding CTC layer.

Decoding-time biasing. The earliest approaches leave the acoustic model untouched and intervene during decoding. Shallow fusion interpolates an external LM into the beam search score computation [3], and trie- or WFST-based deep biasing restricts and reweights hypotheses using a prefix structure built from a keyword list [7]. More recent work has shifted toward training-free setups, utilizing CTC-based word spotting to trigger boosting [8] or synthetic multi-pronunciation tries to bias a frozen model zero-shot [9]. For autoregressive models, prompt prefixing has also emerged as a practical decoding-time solution [10].

Learned bias encoders. Another line of research directly models the interaction between acoustic features and contextual information. CLAS introduced an attention-based bias encoder that embeds each phrase, allowing the decoder to attend to the resulting embedding set [1]. The context-aware Transformer-Transducer extended this mechanism to multi-head attention over both audio and label representations [11]. Contextual adapters made this framework

^{*}Work done at Sakana AI.

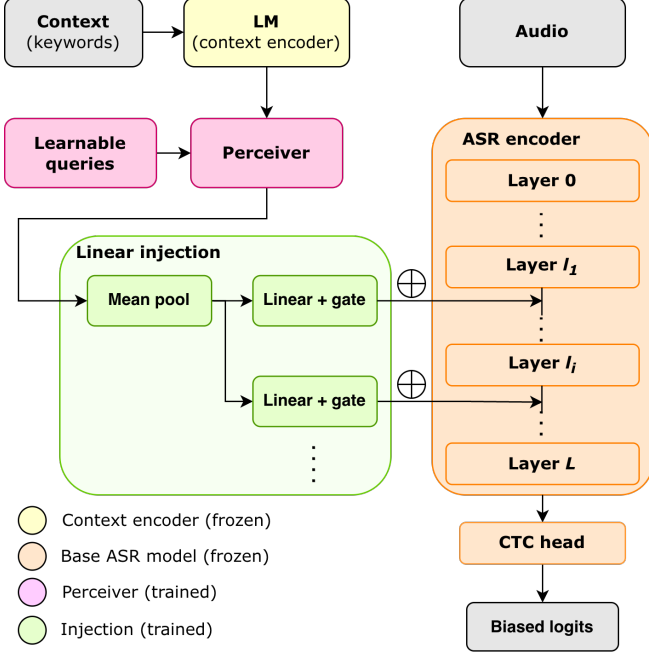


Fig. 1. Our proposed methods for context biasing. The linear injection layer is instantiated for each target ASR layer.

parameter-efficient by inserting lightweight attention modules into a frozen transducer [12], which was later refined using learned gating to deactivate biasing on non-entity frames [13]. However, because these approaches represent the bias list as one embedding per phrase, the computational cost of attending to the list scales linearly with its size, and the attention distributions become increasingly diffuse on large lists. The neural associative memory line makes this tension explicit, addressing it with two-pass top- K retrieval so that the acoustic encoder attends only to a shortlist [14, 15, 16, 17].

Biasing CTC models. Because CTC decoding is non-autoregressive and assumes frame-level conditional independence, contextual information must be injected into the encoder representations or frame posteriors rather than a decoder state. CPPNet fuses an attention-derived context vector into the encoder output and adds an auxiliary contextual-phrase prediction loss [18], whereas Zhang et al. [19] add an attention-weighted bias score directly to the CTC linear projections in a hybrid CTC/attention architecture. Other approaches inject contextual supervision at intermediate encoder layers [20, 21, 22], apply biasing exclusively at CTC spike frames [23], or expand the output vocabulary with dedicated bias tokens [24].

Unlike these methods incurring additional computation to inject keyword bias, our proposed **BaLEEN** uses a hypernetwork to add latent contextual bias vectors directly into the backbone model without changing its decoding pipeline.

3. METHOD

Our architecture comprises three main components: (1) a language model (LM) for context encoding, (2) a CTC-based ASR backbone, and (3) a bottleneck module that compresses the encoded context and injects contextual knowledge into the ASR model. To convert a conventional ASR model into a dynamically adaptable system, we adopt a hypernetwork framework that computes context-dependent

bias parameters for the ASR model. Figure 1 provides a schematic overview of our method.

In this architecture, the model is designed to handle arbitrary textual context. The LM converts the context into a sequence of representation vectors, and the context bottleneck compresses these token-wise representations into fixed-size semantic representations. Because contextual inputs vary in length, we implement the bottleneck using a Perceiver network [6]. The Perceiver compresses variable-length sequence inputs into a fixed-length latent sequence via cross-attention with learnable query vectors. While originally proposed for non-text modality integration (e.g., vision), the Perceiver has also proven effective for compressing text [25].

Formally, let x denote the input context. We extract hidden outputs from the ℓ -th intermediate layer of the LM and project them into d -dimensional embeddings: $C = \text{FFN}(\text{LM}^{(\ell)}(x)) \in \mathbb{R}^{n \times d}$, where n is the number of context tokens. Let $Q \in \mathbb{R}^{N \times d}$ denote the learnable latent query vectors, where N and d are the latent query length and hidden dimension, respectively (set to $N = 16$ and $d = 512$ by default).¹ The sequence C passes through a single Perceiver block consisting of cross-attention followed by self-attention, with residual connections around each layer: $M = \text{Perceiver}(Q, C) \in \mathbb{R}^{N \times d}$.

The compressed representation M is subsequently injected into targeted intermediate layers of the ASR encoder via linear projections. For a selected subset of target ASR layers $\mathcal{L} \subseteq \{1, \dots, L\}$ (where L is the total number of encoder layers), M is mean-pooled across its sequence dimension into a vector $m = \pi(M) \in \mathbb{R}^d$. For each target layer $l \in \mathcal{L}$, m is projected through a layer-specific weight matrix $W_l \in \mathbb{R}^{d \times d_{\text{ASR}}}$ and bias vector $b_l \in \mathbb{R}^{d_{\text{ASR}}}$:

$$B_l = W_l^\top m + b_l \in \mathbb{R}^{d_{\text{ASR}}}. \quad (1)$$

The resulting contextual bias vector B_l is injected into the hidden representation matrix $H^{(l)} \in \mathbb{R}^{T \times d_{\text{ASR}}}$ across all T acoustic frames:

$$H^{(l)} \leftarrow H^{(l)} + \gamma_l \mathbf{1}_T B_l^\top, \quad (2)$$

where $\mathbf{1}_T \in \mathbb{R}^{T \times 1}$ is a column vector of ones and $\gamma_l \in \mathbb{R}$ is a scalar scaling parameter. At the beginning of training, γ_l is initialized to zero for all l to avoid catastrophic forgetting. Under this design, the generated context bias is independent of the audio input and can be added to the ASR intermediate representations in a modular, plug-and-play fashion. The user can easily disable contextual biasing at inference time by detaching the hypernetwork module or setting $\gamma_l = 0$.

A natural question regarding uniform bias injection is whether adding a static, time-invariant offset B_l across all time frames risks degrading non-keyword speech or triggering false positives. Our framework is motivated by the hypothesis that the internal self-attention and non-linear projections of the backbone ASR encoder serve as dynamic temporal filters. While B_l remains constant over time, local acoustic representations vary. When acoustic evidence phonetically aligns with a target entity, self-attention and feed-forward projections amplify the context-biased manifold. Conversely, on frames lacking relevant acoustic support, non-linear activations suppress B_l , mapping hidden states back toward standard subword or CTC blank (ϵ) posterior distributions. By distributing this static injection across all encoder layers, the network achieves progressive, multi-stage contextual steering without requiring frame-wise cross-attention or on-the-fly context encoding at inference time.

¹For simplicity, normalization layers are omitted in equations henceforth; see our implementation details released publicly upon acceptance.

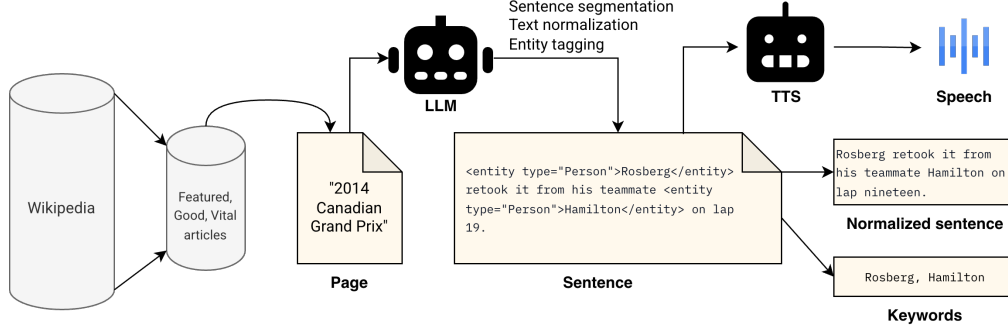


Fig. 2. Synthetic dataset construction flow.

The model is trained by minimizing the standard CTC loss. Throughout training, the parameters of both the context encoder LM and the backbone ASR model remain frozen, and only the parameters of the auxiliary context bottleneck and projection layers are updated. At inference time, when the context set is fixed, the bias term $\gamma_l \mathbf{1}_T B_l^\top$ can be precomputed once. As a result, the bias acts as a set of static offset parameters, avoiding any recomputation through the LM or bottleneck module and incurring virtually zero runtime overhead.

4. DATASET

Training the context bottleneck requires triplet data consisting of (audio, transcription, keyword list) across a sufficiently diverse sample space to achieve robust generalized context adaptation. To address this, we construct a synthetic dataset comprising 144k audio samples derived from 993 English Wikipedia articles using LLM-powered entity tagging and text-to-speech (TTS) synthesis. Figure 2 illustrates the dataset generation pipeline to obtain the triplet data.

We leverage Wikipedia to construct a corpus spanning diverse domains with high keyword density and high-quality text. First, we extract the Featured Articles², Good Articles³, and Vital Articles⁴, curated by Wikipedia editors based on completeness, neutrality, and prose depth, using the Wikipedia-API Python library⁵. We then filter out disambiguation pages and articles shorter than 3,000 characters. Next, each remaining article is segmented into sentences and annotated with named entity tags using GPT-5.4 Mini. Finally, each sentence is converted to speech using Gemini 3.1 Flash TTS, where a voice is randomly sampled from 30 prebuilt speaker profiles to ensure acoustic and speaker diversity.

The dataset supports two keyword sampling configurations: (1) using only the keywords that appear strictly in the spoken audio transcription, or (2) using all keywords appearing within the source article. The latter configuration introduces contextually relevant hard-negative distractors alongside spoken keywords, creating a more challenging training scenario that fosters stronger context adaptability.

We partition the dataset into 80% training, 10% validation, and

10% test splits. Crucially, these splits have zero overlap in their source articles (*i.e.*, unique contexts) to prevent context leakage during evaluation.

5. EXPERIMENTS

To encode keywords in a spelling-aware manner, we extract hidden representations from layer ℓ ($\ell = 7$ by default) of ByT5 [26]. For the ASR backbone, we employ the CTC variant of parakeet-tdt-ctc-110m [27], which consists of 17 Conformer encoder layers followed by a CTC layer trained on English. The CTC vocabulary comprises 1,025 case-sensitive subword tokens constructed via Byte-Pair Encoding [28]. To prevent out-of-vocabulary (OOV) issues arising from the CTC model’s relatively small subword vocabulary, all transcript text in the dataset is normalized. For example, because the ParakeetCTC vocabulary does not contain numeric digits, numbers are expanded into spelled-out words (e.g., “3” to “three”) by the sentence processing LLM at the dataset creation phase (cf. Figure 2). Additionally, non-phonetic punctuation marks (excluding periods and commas) are removed prior to TTS synthesis. Utterances containing any remaining unsupported characters are filtered out from both training and evaluation splits. To evaluate performance, we report the *keyword miss rate* (KMR), which is defined as the proportion of ground-truth keywords omitted in predictions ($1 - \text{Recall}$), alongside word error rate (WER) and character error rate (CER).

To investigate which ASR layers are critical for keyword biasing, we evaluate four distinct layer selection configuration $\mathcal{L}$: (1) later layers ($\mathcal{L} = \{l^{(13)}, l^{(14)}, l^{(15)}, l^{(16)}\}$), (2) middle-to-later layers ($\mathcal{L} = \{l^{(10)}, l^{(12)}, l^{(14)}, l^{(16)}\}$), (3) uniformly distributed layers ($\mathcal{L} = \{l^{(4)}, l^{(8)}, l^{(12)}, l^{(16)}\}$), (4) all layers ($\mathcal{L} = \{l^{(i)}\}_{i=0}^{16}$). Additionally, we evaluate whether augmenting training context lists with probabilistically sampled hard negatives from the same source article enhances model generalization. Note that dynamic distractor sampling is used only during training, while evaluation contexts remain fixed with keywords strictly appearing in the utterance. In total, $4 \times 2 = 8$ settings are compared against the non-fine-tuned ASR baseline. The resulting context bottleneck module comprises 37.7M parameters, with each linear layer projection adding 2.6M parameters.

Unless specified otherwise, we adopt the following hyperparameter settings across all experiments: a learning rate of 5×10^{-5} with a 10% linear warmup followed by linear decay, 10 training epochs, a batch size of 32, and a single Perceiver encoder block. The Perceiver decoder was omitted after preliminary experiments indicated it pro-

²https://en.wikipedia.org/wiki/Wikipedia:Featured_articles

³https://en.wikipedia.org/wiki/Wikipedia:Good_articles

⁴https://en.wikipedia.org/wiki/Wikipedia:Vital_articles

⁵<https://pypi.org/project/Wikipedia-API>

Architecture	WER ^(↓)	CER ^(↓)	KMR ^(↓)
Baseline (no FT)	9.53	2.74	40.66
Baseline (XAttn + logits)	11.47	3.29	35.59
Baseline (XAttn + logits), DC	11.16	2.58	35.26
Linear + hidden (late)	7.97	2.06	36.37
Linear + hidden (midlate)	7.86	2.06	36.15
Linear + hidden (uniform)	7.86	2.04	35.67
Linear + hidden (all)	7.62	1.98	35.71
Linear + hidden (late), DC	7.75	2.01	35.57
Linear + hidden (midlate), DC	7.97	2.09	35.81
Linear + hidden (uniform), DC	8.31	2.12	35.47
Linear + hidden (all), DC	7.38	1.87	34.85

Table 1. Evaluation results on the validation set. All metrics are reported in percentages (%). DC stands for dynamic context with probabilistically mixed hard-negative keywords.

vided no measurable performance benefit. For each sample, keywords were concatenated into a single string, separated by a comma and a space. For training stability and batching efficiency, speech samples shorter than 1 second or longer than 20 seconds are excluded from training. All experiments are run on an NVIDIA H100 GPU (80GB), and all reported metrics are computed using greedy decoding.

Inspired by prior work using cross-attention for contextual injection [11] we also implement a direct-biasing baseline that modifies logits $\mathbf{s} = \text{CTC}(H^{(L)})$ by integrating the compressed vectors M through cross-attention: $\mathbf{s} \leftarrow \mathbf{s} + \text{XAttn}(H^{(L)}, M, M)$.

6. RESULTS

Table 1 reports the performance metrics for model checkpoints achieving the lowest KMR on the validation set. All proposed adaptation variants improve keyword recognizability while simultaneously outperforming the non-fine-tuned baseline ASR model in overall WER and CER. Incorporating hard-negative distractors into the contextual keyword list during training further enhances model robustness. The overall best-performing configuration, injecting linearly generated biases into every ASR encoder layer using dynamic context with hard-negative distractors, reduces WER by 22.6%, CER by 31.7%, and KMR by 14.3% relative to the baseline. While directly biasing output logits via cross-attention also improves keyword recognition, it exhibits training instability, consistently appending an extraneous random token to the end of predicted transcripts, likely due to unconditioned cross-attention corrupting the CTC blank token distribution.

Evaluation on the test set (Table 2) reveals similar trends. Direct logit biasing via cross-attention without distractor keywords achieves the lowest KMR, yielding an 11.0% relative improvement over the non-fine-tuned baseline. Nevertheless, it suffers from the same decoding instability, consistently appending spurious tokens and degrading overall WER and CER. Furthermore, this KMR gain occurs only when the keyword lists are restricted to keywords uttered in the audio; however, this idealized setup is rarely encountered in real-world deployments. Under realistic dynamic context conditions, hidden-state biasing across all ASR layers yields the best balanced performance. This setting reduces WER by 21% and CER by 28% relative to the non-fine-tuned baseline, while maintaining a competitive 8.7% relative reduction in KMR. It is also important to

Architecture	WER ^(↓)	CER ^(↓)	KMR ^(↓)
Baseline (no FT)	10.36	3.04	43.34
Baseline (XAttn + logits)	11.19	3.21	38.59
Baseline (XAttn + logits), DC	13.91	5.12	42.32
Linear + hidden (late)	9.51	2.63	45.92
Linear + hidden (midlate)	9.16	2.49	44.41
Linear + hidden (uniform)	8.70	2.36	41.32
Linear + hidden (all)	9.22	2.56	44.05
Linear + hidden (late), DC	8.69	2.36	42.24
Linear + hidden (midlate), DC	9.78	2.72	47.39
Linear + hidden (uniform), DC	10.05	2.70	43.48
Linear + hidden (all), DC	8.18	2.19	39.55

Table 2. Evaluation results on the test set. All metrics are reported in percentages (%).

note that the compressed context vectors in the direct logit biasing baseline can attend to encoded acoustic frames with incurred computational costs. In contrast, our proposed method injects bias under a more restrictive condition agnostic of the audio input.

Comparing late, middle-late, and uniform 4-layer injection strategies indicates that among partial layer subsets, the specific choice of targeted encoder layers does not significantly alter performance. However, comparing performance metrics across the validation and test sets reveals that targeting only a subset of ASR encoder layers yields weak generalization, while biasing all layers with dynamic context simultaneously shows improvement over the non-fine-tuned baseline.

Finally, constructing a large-scale synthetic dataset was necessary to evaluate contextual biasing under dense, arbitrary keyword lists. Accordingly, the primary objective of our experiments was to establish context generalizability under these complex synthetic conditions, which was validated by our results. Nevertheless, we acknowledge that relying on synthesized speech presents a practical limitation regarding generalization to real-world human voices, accents, and diverse acoustic environments.

7. CONCLUSION

This paper introduced **BaLEEN** (Biasing with Latent Encoded Entities), a lightweight, plug-and-play contextual biasing framework based on hypernetworks. Our method encodes and compresses arbitrary, variable-length contextual entities into fixed-length latent vectors, which are then injected into intermediate hidden representations of a frozen ASR model. This approach enables context-aware adaptation without modifying or fine-tuning the underlying backbone weights. Furthermore, because the generated bias vectors can be precomputed for a static context list, runtime computational overhead and latency are zero during inference. To train and benchmark our model, we constructed a large-scale Wikipedia-based synthetic dataset consisting of triplets of audio, transcripts, and contextual keyword lists. Experimental evaluations demonstrate that injecting linearly projected bias vectors across all ASR encoder layers achieves the best overall performance, reducing the KMR by 14.3% while simultaneously improving WER and CER over the baseline. In addition, we showed that incorporating hard-negative distractors into the context list during training enhances model generalization under realistic deployment conditions.

8. REFERENCES

- [1] Golan Pundak, Tara N. Sainath, Rohit Prabhavalkar, Anjali Kannan, and Ding Zhao, “Deep context: end-to-end contextual speech recognition,” in *arXiv preprint arXiv:1808.02480*, 2018.
- [2] Suyoun Kim and Florian Metze, “Dialog-context aware end-to-end speech recognition,” in *Proc. IEEE SLT*, 2018, pp. 434–440.
- [3] Ding Zhao, Tara N. Sainath, David Rybach, Pat Rondon, Deepti Bhatia, Bo Li, and Ruoming Pang, “Shallow-Fusion End-to-End Contextual Biasing,” in *Proc. Interspeech*, 2019, pp. 1418–1422.
- [4] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen, “LoRA: Low-rank adaptation of large language models,” in *Proc. ICLR*, 2022.
- [5] David Ha, Andrew M. Dai, and Quoc V. Le, “HyperNetworks,” in *Proc. ICLR*, 2017.
- [6] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira, “Perceiver: General perception with iterative attention,” in *Proc. ICML*, 2021, pp. 4651–4664.
- [7] Duc Le, Mahaveer Jain, Gil Keren, Suyoun Kim, Yangyang Shi, Jay Mahadeokar, Julian Chan, Yuan Shangguan, Christian Fuegen, Ozlem Kalinli, Yatharth Saraf, and Michael L. Seltzer, “Contextualized streaming end-to-end speech recognition with trie-based deep biasing and shallow fusion,” in *Proc. Interspeech*, 2021, pp. 1772–1776.
- [8] Andrei Andrusenko, Aleksandr Laptev, Vladimir Bataev, Vitaly Lavrukhin, and Boris Ginsburg, “Fast context-biasing for CTC and transducer ASR models with CTC-based word spotter,” in *Proc. Interspeech*, 2024, pp. 757–761.
- [9] Changsong Liu, Yizhou Peng, and Eng Siong Chng, “Zero-shot context biasing with trie-based decoding using synthetic multi-pronunciation,” in *Proc. APSIPA*, 2025.
- [10] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever, “Robust speech recognition via large-scale weak supervision,” in *arXiv preprint arXiv:2212.04356*, 2022.
- [11] Feng-Ju Chang, Jing Liu, Martin Radfar, Athanasios Mouchtaris, Maurizio Omologo, Ariya Rastrow, and Siegfried Kunzmann, “Context-aware transformer transducer for speech recognition,” in *Proc. IEEE ASRU*, 2021.
- [12] Kanthashree Mysore Sathyendra, Thejaswi Muniyappa, Feng-Ju Chang, Jing Liu, Jinru Su, Grant P. Strimel, Athanasios Mouchtaris, and Siegfried Kunzmann, “Contextual adapters for personalized speech recognition in neural transducers,” in *Proc. ICASSP*, 2022, pp. 8537–8541.
- [13] Anastasios Alexandridis, Kanthashree Mysore Sathyendra, Grant Strimel, Feng-Ju (Claire) Chang, Ariya Rastrow, Nathan Susanj, and Thanasis Mouchtaris, “Gated contextual adapters for selective contextual biasing in neural transducers,” in *Proc. ICASSP*, 2023, pp. 1–5.
- [14] Tsendsuren Munkhdalai, Khe Chai Sim, Angad Chandorkar, Fan Gao, Mason Chua, Trevor Strohman, and Françoise Beaufays, “Fast contextual adaptation with neural associative memory for on-device personalized speech recognition,” in *Proc. ICASSP*, 2022, pp. 6632–6636.
- [15] Tsendsuren Munkhdalai, Zelin Wu, Golan Pundak, Khe Chai Sim, Jiayang Li, Pat Rondon, and Tara N. Sainath, “Nam+: Towards scalable end-to-end contextual biasing for adaptive asr,” in *Proc. IEEE SLT*, 2023, pp. 190–196.
- [16] Zelin Wu, Tsendsuren Munkhdalai, Pat Rondon, Golan Pundak, Khe Chai Sim, and Christopher Li, “Dual-Mode NAM: Effective Top-K Context Injection for End-to-End ASR,” in *Proc. Interspeech*, 2023, pp. 221–225.
- [17] Zelin Wu, Gan Song, Christopher Li, Pat Rondon, Zhong Meng, Xavier Velez, Weiran Wang, Diamantino Caseiro, Golan Pundak, Tsendsuren Munkhdalai, Angad Chandorkar, and Rohit Prabhavalkar, “Deferred NAM: Low-latency top-k context injection via deferred context encoding for non-streaming ASR,” 2024.
- [18] Kaixun Huang, Ao Zhang, Zhanheng Yang, Pengcheng Guo, Bingshen Mu, Tianyi Xu, and Lei Xie, “Contextualized End-to-End Speech Recognition with Contextual Phrase Prediction Network,” in *Proc. Interspeech*, 2023, pp. 4933–4937.
- [19] Zhengyi Zhang and Pan Zhou, “End-to-end contextual asr based on posterior distribution adaptation for hybrid ctc/attention system,” in *arXiv preprint arXiv:2202.09003*, 2022.
- [20] Muhammad Shakeel, Yui Sudo, Yifan Peng, and Shinji Watanabe, “Contextualized end-to-end automatic speech recognition with intermediate biasing loss,” in *Proc. Interspeech*, 2024, p. 3909–3913.
- [21] Yu Nakagome and Michael Hentschel, “Interbiasing: Boost unseen word recognition through biasing intermediate predictions,” in *Proc. Interspeech*, 2024, pp. 207–211.
- [22] Yu Nakagome and Michael Hentschel, “WCTC-Biasing: Retraining-free Contextual Biasing ASR with Wildcard CTC-based Keyword Spotting and Inter-layer Biasing,” in *Proc. Interspeech*, 2025, pp. 5178–5182.
- [23] Kaixun Huang, Ao Zhang, Binbin Zhang, Tianyi Xu, Xingchen Song, and Lei Xie, “Spike-triggered contextual biasing for end-to-end mandarin speech recognition,” in *Proc. IEEE ASRU*, 2023, pp. 1–8.
- [24] Zhennan Lin, Kaixun Huang, Wei Ren, Linju Yang, and Lei Xie, “Contextualized automatic speech recognition with dynamic vocabulary prediction and activation,” in *Proc. Interspeech*, 2025, pp. 3174–3178.
- [25] Rujikorn Charakorn, Edoardo Cetin, Yujin Tang, and Robert Tjarko Lange, “Instant transformer adaption via hyperLoRA,” in *Proc. FITML*, 2024.
- [26] Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel, “ByT5: Towards a token-free future with pre-trained byte-to-byte models,” *TACL*, vol. 10, pp. 291–306, 2022.
- [27] Dima Rekish, Nithin Rao Koluguri, Samuel Krizan, Somshubra Majumdar, Vahid Noroozi, He Huang, Oleksii Hrinchuk, Krishna Puvvada, Ankur Kumar, Jagadeesh Balam, and Boris Ginsburg, “Fast conformer with linearly scalable attention for efficient speech recognition,” in *Proc. IEEE ASRU*, 2023, pp. 1–8.
- [28] Rico Sennrich, Barry Haddow, and Alexandra Birch, “Neural machine translation of rare words with subword units,” in *Proc. ACL*, 2016, pp. 1715–1725.