Interspeech 2022
18-22 September 2022, Incheon, Korea

On-the-fly ASR Corrections with Audio Exemplars

Golan Pundak, Tsendsuren Munkhdalai, Khe Chai Sim

Google Inc., U.S.A

{golan,tsendsuren,khechai}@google.com

Abstract

On-device end-to-end (E2E) models are required to handle long-tail vocabulary and a large number of acoustic conditions. With finite amount of training data some of these conditions and vocabulary words are unseen during training, which often leads to recognition errors. Text-based contextual biasing is intended to mitigate this problem, yet it works well only when sufficient textual context is provided, and when the speech signal is well modeled by the ASR system. In this work, we propose to extend biasing to operate directly in the audio domain. We address a scenario where audio samples and the associated transcriptions are available, as is the case of manually corrected voice typing. We propose to directly compare incoming audio embeddings against a list of Audio Exemplars (AE), each associated with a text correction. We demonstrate the effectiveness of our approach by correcting the outputs of a production-quality RNNT model, which results in relative-WER reduction of 21.7% (one-shot) and 33.7% (multi-shot) on the Wiki-Names data set.
Index Terms: on-device end-to-end models

1. Introduction

Automatic Speech Recognition (ASR) systems are required to handle a large vocabulary and a large number of acoustic conditions, in order to be useful. To obtain such large coverage End-to-end (E2E) models, which map audio directly to text, are typically trained with a large and diverse data set (as in [1]), augmented with an inference-time ability to inject context into the recognition process (e.g. contact names or phone numbers, which are often misrecognized).

Injection of context information (often referred to as "biasing") in E2E ASR systems is typically done via direct modification of the recognition lattice, thus operating in the text domain [2, 3, 4]. This works well when a prefix (or suffix) is used to trigger biasing (as in "call john"). Other works attempt to inject the text directly into the ASR model and learn the contextualization process in an E2E manner - reducing the need for a triggering prefix [5, 6, 7, 8]. These *model-based* approaches attempt to modify the model's internal representations to better match the given context phrases. These methods tend to work well only if the acoustic conditions at test time are well represented in the training data (e.g. accent and noise conditions). Another problem with text-based biasing systems is non-standard spellings, which can be somewhat mitigated by using phonetic representations as in [9].

In this work we explore the extension of biasing to the audio domain by utilizing *Audio Exemplars* (AE). Our system learns from user-corrections of misrecognized utterances by extracting audio exemplars from them and associating them with the corrected text. In test-time, these exemplars are compared to incoming audio and the ASR hypotheses are corrected if there is a good match. See Fig. 1 for a system overview.

In our experiments we use Recurrent-Neural-Network-Transducer (RNNT) [10] both as an ASR system and as a source for audio-representations, used as exemplars.

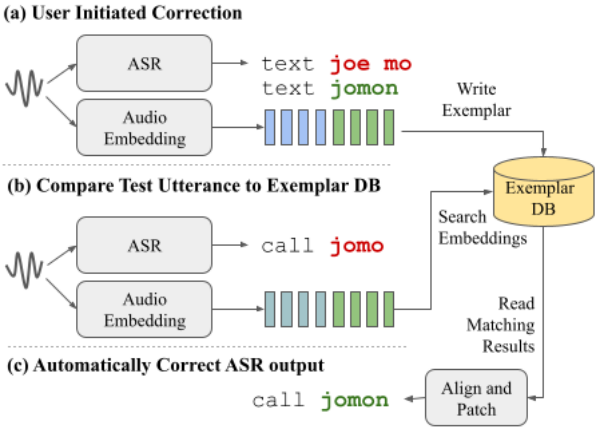

Figure 1: *System Overview: (a) A user edits a recognized utterance, and the system performs an extraction of the corrected text, paired with the corresponding audio-exemplar, which is then placed in an Exemplar Database (DB). (b) In test time an incoming utterance is matched against the entire DB frame by frame. (c) Matching exemplars are aligned to the test utterance and the ASR output is corrected (patched) with the associated text.*

Our proposed system can work with a single audio-exemplar (one-shot) or with multiple ones (multi-shot), corresponding to first-time or a repeated user-correction. Our system can handle mismatched acoustic conditions and words unseen in training, including rare stylization (e.g. "Ke$ha").

Our system is suitable for on-device streaming recognition (for simplicity, we present a non-streaming implementation in this paper, but all computations can be made streaming in principle). Furthermore, our system can be integrated with existing RNN-T solutions without a need for an architectural change or even a dedicated training phase.

Exemplar-based systems have been used in ASR systems before. Some examples include [11] where text-exemplars are used with nearest-neighbours database for text-domain biasing. In [12, 13] audio-exemplars are used for keyword detection, and in [14] a database of frame-embedding is used to improve an Acoustic Model. However, our work is the first, to the best of our knowledge, that uses Audio Exemplars to improve large-vocaublary speech recognition in an E2E setup.

We demonstrate the effectiveness of our Audio-Exemplar approach on a new version of the Wiki-Names dataset [15], where it achives relative-WER reductions of 21.7% (one-shot) and 33.7% (multi-shot). We also combine our system with the Neural Associative Memory (NAM) system, introduced in [7] where we obtain 39.5% one-shot relative-WER reduction.

The reminder of the paper is organized as follows. In Sec-

10.21437/Interspeech.2022-389

tion 2 we describe our method. In Sections 3 and 4 we describe our experiments and results, and conclude in Section 5.

2. Corrections with Audio-Exemplars

The proposed correction process is depicted in Fig. 1. In this section we describe it in more details.

2.1. Audio Representation

One basic requirement from an audio representation is robustness to different acoustic conditions. This can be achieved by using any pre-trained audio encoder. In this work we propose to use the same RNN-T encodings that are used by our ASR system. The main benefit from this choice is simple system deployment on devices that already support the RNN-T model. The raw RNN-T embeddings can be read from any of the encoder's layers, and we denote them as $\hat{X}_t \in \mathbb{R}^D$ with $t \in \{1...T\}$, for a length T utterance. Individual features are denoted as $\hat{X}_{it}$.

In order to reduce the variance in the raw features we perform whitening of the input features $\hat{X}_{it}$ by mapping them to $X_{it} = (\hat{X}_{it} - \mu_i)/\sigma_i$ where μ_i and σ_i are respectively the feature's mean and standard deviation across time.

2.1.1. Vector Suppression

Some vectors are too common to be used in an alignment (e.g. those that corresponding to silence or background noise), and might results in false matches (for example, see the dark region at the start of the utterances in Fig. 2). We would like to suppress these vectors and exclude them while matching. To test how common a vector is we compute the cosine similarity between all the frames in a given utterance and suppress the ones with a large equivalence class. As neighbouring vectors are often similar to each other we only count vectors that are at least C_w frames away, and suppress the ones that are equivalent to at least C_s such other vectors.

2.2. Exemplar Extraction

A user-correction is an edit operation that the user performs on a span of text within the top recognized hypothesis, e.g. *"text **joe mo**"* is corrected to *"text **jomon**"* as in Fig. 1.a. We denote by Y_{err} and Y_{corr} the misrecognized and corrected text spans respectively.

Our first task is to find the start and end frames of Y_{err} which we denote t_s and t_e. We could set these to be the first and last frames associated with the word-pieces of Y_{err}, but this does not work well for RNN-T which tends to delay its outputs and emit <*blank*> symbols while doing so. Therefore, we include the <*blank*> outputs preceding Y_{err} and push t_s "backwards", until hitting either a non-blank output symbol or a suppressed vector (as defined in Sec. 2.1.1). With that our exemplar is the vector sequence $V = X_{t_s}...X_{t_e}$ and we store it in the exemplar database as the tuple $[V, Y_{corr}]$.

We note that the estimation of both t_s and t_e may be inexact, and we account for this both in alignment generation (Algo. 1) and in the top hypothesis patching phase (Sec 2.5).

2.3. Exemplar Database Search

Given a test utterance U, we search all of its frames in the Exemplar DB (Fig. 1.b). We group the set of matching vectors returned from the database by exemplar-id, and only consider exemplars with sufficient "hits" for alignment.

To reduce the DB search cost, any approximate nearest neighbour search algorithm can be used (e.g LSH [16]). Memory footprint can be reduced by lowering the exemplar's precision: we actually found that keeping just the sign (1 bit) is sufficient for our whitened feature value distributions.

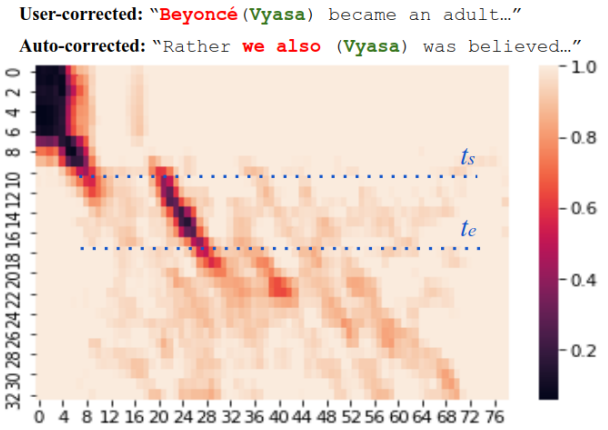

Figure 2: *A cosine distance map between all the frames of a user-corrected utterance U (y-axis) and an auto-corrected test utterance. The exemplar from U is extracted between times t_s and t_e (See details in sections 2.2 and 2.4). The dark region in the start of the utterances is due to silence.*

2.4. Exemplar Alignment

We now describe the alignment process (Fig. 1.c).

We align the frames of our test utterance U against detected tuples $\{(V, Y_{corr})\}$, returned from the exemplar DB. We start by computing the cosine distance matrix M between all the features of the test utterance U and V, as in Fig. 2. Next, we use a Dynamic Time Warping (DTW) [17] matching procedure with insertion and deletion penalties (Algo. 1) that allows us to reject poor matches.

We trace back the steps in the distance matrix D returned by the alignment algorithm (Algo. 1) to obtain an initial partial alignment between U and V. In order to improve recall we further refine the alignment by allowing it to extend a few frames in each direction. This refinement step is done on a smaller search space then the first one thus does not require much additional computation. We note that the back tracking step can be replaced by beam-search for streaming applications.

DTW allows local pace changes in the alignment, which might lead to unrealistic false matches. To avoid these, we only keep the largest span of matching frames that contract or expand the aligned times by a factor of 3 or less.

2.4.1. Aligning Multiple Positive Exemplars

An advantage of our system over text-based systems is the ability to leverage multiple positive examples (same text, different audio). The simplest approach for handling multiple positive cases is to align each one of them separately and accept the one with the best match. This is indeed the approach we take in our experiments in Section 4.1, while leaving more complex approaches as future work.

Algorithm 1: Dynamic Time Warping (DTW) with Insertion / Deletion Penalties

Input:
$w_{ins} \in R$; // Insertion penalty
$w_{del} \in R$; // Deletion penalty
$M \in R^{n \times m}$; // Cosine Distance
Output: $D \in R^{(n+1)\times(m+1)}$
$D_{i,j} \leftarrow 0 \ \forall i,j$; // Initialize D
for $i = 0$ **to** n **do**
  $D_{i,0} \leftarrow i \times w_{del}$; // Initialize D
  $D_{0,i} \leftarrow i \times w_{ins}$; // Initialize D
end
for $i = 1$ **to** n **do**
  for $j = 1$ **to** m **do**

$$D_{i,j} \leftarrow min \begin{cases} D_{i-1,j-1} + M_{i,j}; & \texttt{// Match} \\ D_{i,j-1} + w_{ins}; & \texttt{// Insertion} \\ D_{i-1,j} + w_{del}; & \texttt{// Deletion} \end{cases}$$

  end
end

2.5. Hypothesis Patching

We now describe the automated correction process, which we call *patching* (Fig. 1.c).

Given an alignment between a test utterance U and an exemplar V (associated with a correction Y_{corr}), we proceed by determining a *target range* in U to be corrected. The target range is determined by considering the largest range of frames in U aligned to matching frames in V.

Next, we modify the top hypothesis transcript. We start by determining the text span associated with the target range. Replacing that span directly may result in mid-word insertions, so instead we extend both ends of the span until word boundaries are detected. We then replace that span with Y_{corr}. While this procedure might introduce word-deletions, it is found to perform well in practice.

3. Experiments

3.1. Datasets

We test our approach on a subset of the Wiki-Names corpus [15](second version). The Wiki-Names corpus was designed for entity mention personalization. This data set contains read speech collected from 500 speakers. The text prompts used for the data collection were extracted from US English Wikipedia pages containing named entity mentions with type "person" that are misrecognized by a reference ASR model. This dataset is collected in a similar way to the first version [15], but contains less accented speech, making it more suitable for testing isolated user edits. We test on a 100 speakers, each associated with 100 misrecognized utterances, covering 10 different entities. We extract an exemplar from each utterance, discarding exemplars shorter than 240ms (this happens in less than 1% of the cases).

Each utterance also participates as a test utterance, where it is tested against exemplars from other N_{pos} utterances with matching entity (different audio) and N_{neg} exemplars with different entity, which we call **distractors**. Altogether our exemplar database has size of $N = N_{pos} + N_{neg}$, testing a scenario in which a user has already made N corrections. We set $N_{neg} = 80$, unless specified otherwise and N_{pos} is varying by experiment. The constants in our system were tuned over a small held-out dev set.

3.2. ASR Model and Audio Embedding

The model we use for both speech recognition and audio embedding is an RNN-T model with Cascaded Conformer Encoder as in [18]. The model was trained with FastEmit [19] which encourages the model to emit prediction earlier, this contributing to the accuracy of the audio-to-wordpiece alignment. The model was trained with multi-domain data, as described in [1] and [20]. The encoder contains 12 causal (streaming) conformer blocks followed by 5 non-causal (non-streaming) conformer blocks. As input, the model uses 128-dimensional log-Mel pre-processed as in [7]. We used a word-piece model with vocabulary size of 4096 for the transcript tokenization.

For audio embedding we use 512-dimensional vectors read as the outputs of the 10th causal conformer block, accounting for 60ms of audio. We perform our experiments with 32bit precision but note that the memory footprint can be reduced by keeping just the sign of the exemplar's features (1bit per feature, or 61KByte per 1 minute of audio) - this seems to have only a minor effect on WER.

3.3. Audio-Exemplars Detection and Correction Setup

For the Exemplar Search Phase (Section 2.3) exemplars are returned from the database only if thay have at least 4 matching vectors (cosine similarity of 0.7 or more) with a test vector. These values where found to filter out 95% of the database with no significant recall loss. For the Exemplar Alignment phase (Section 2.4) We use $w_{ins} = w_{del} = 0.2$. For the refined alignment we extend the original match by 3 frames on each side and realign with $w_{ins} = w_{del} = 0.3$. We accept an alignment of a given exemplar if the number of test vectors marked as 'Match' by DTW (Algo 1), surpasses a specified *Accept Threshold*. Finally the top accepted exemplar (largest match) is used for text patching (Sec. 2.5).

For *vector suppression* (Section 2.1.1) we use $C_s = 2$ and $C_w = 3$, and use the mean similarity of vectors separated by $C_w + 1$ frames as equivalence-threshold.

3.4. Neural Associative Memory

Neural Associative Memory (NAM), introduced in [7] is a recently proposed neural architecture that allows RNN-T model-based-biasing by modifying the encoded audio representations to account for given text bias phrases. We use the same setup as in [7], but replace the underlying ASR system with the same one as in our setup (Sec. 3.2). The NAM layer is placed on top of the casual encoder (above the 12th conformer block), downstream from the conformer block used for Audio-Exemplar extraction (block 10). We provide NAM with a list of text bias phrases from our exemplar DB. We test NAM in a setup that has one correct match and 10 unique distractors.

4. Results

4.1. Same Speaker Matching

Table 1 shows the system's performance with several Accept Thresholds for the case of 1 match and 80 distractors in the Exemplar DB (One-Shot setup). Table 2 shows similar statistics for 0 matches and 80 distractors. We find that with an Accept

Threshold of 8 matching frames (480ms) we are able to obtain 21.76% relative WER improvement in the case of a single match while sustaining a 3.1% relative WER degradation for the no-match case.

Accept Thresh.	Precision	Recall	Base WER	Exp WER	Rel. WER
4	74.5	71.7	22.2	18.20	20.6
6	80.6	61.2	22.2	17.70	22.7
7	87.1	50.5	22.2	17.58	**23.0**
8	93.0	40.8	22.2	17.83	21.7
9	96.8	33.1	22.2	18.30	19.3
10	98.5	26.9	22.2	18.8	16.6

Table 1: *Same speaker, Exemplar DB with 80 distractors and 1 match (One-Shot).*

Accept Thresh.	FA Rate	Base WER	Exp WER	Rel. WER
4	46.6	22.2	27.6	-25.9
6	27.0	22.2	25.3	-14.8
7	13.7	22.2	23.7	-7.5
8	5.5	22.2	22.8	-3.1
9	2.0	22.2	22.4	-1.0
10	0.8	22.2	22.3	**-0.4**

Table 2: *Same speaker, Exemplar DB with 80 distractors, and 0 matches.*

In Table 3 we set a threshold of 8 frames and vary the number of matching exemplars, N_{pos}. We see that with 8 positive exemplars (and 80 distarctors) we can obtain 33.3% relative WER improvement.

N_{pos}	Base WER	Exp WER	Rel. WER
0	22.2	22.8	-3.1
1	22.2	17.8	21.7
4	22.2	15.8	30.6
8	22.2	15.2	33.3

Table 3: *Same speaker, Exemplar DB with 80 distractors, and multiple matches (Multi-Shot). Accept Threshold is set to 8.*

4.1.1. Comparison to NAM

In Table 4 we compare our Audio Exemplars (AE) method with the NAM biasing method [7], which was trained and tested in matching conditions to ours. We also test a simple combination of NAM and our AE system by running both simultaneously - this works well since NAM modifies the embeddings of a higher layer than the ones used for AE. We see that NAM performs better than AE and their combination performs the best. In Table 5 we show that AE and NAM are in fact complementary, as these correct different utterances. We plan to explore a more nuanced system combination in a future work.

System	Base WER	Exp WER	Rel. WER
AE (ours)	22.0	18.1	20.4
NAM	22.0	14.2	35.4
NAM + AE	22.0	13.3	39.5

Table 4: *Comparison of our method to other systems.*

4.2. Speaker Independent Matching

In Table 6 we test a Speaker Independent (SI) setup where the exemplars are taken from a different speaker as in the test utterance. We compare it to the Speaker Dependent (SD) setup, that was used for previous results. We note that with $N_{pos} = 1$ (single positive exemplar) the SI recall is much lower, but improves much with $N_{pos} = 8$. We note that relative WER gains remain low even when the recall improves, due to alignment problems. This result suggest that the embeddings we use are somewhat speaker-specific. Indeed, SI biasing remains a challenge for our system and we plan to improve it in future work.

	NAM Rejects	NAM Accepts
AE Rejects	23%	41%
AE Accepts	13%	23%

Table 5: *How often NAM and AE perform a correction as fraction of test utterances.*

Setup	N_{pos}	Precision	Recall	Rel. WER
SD	1	93.8	45.6	21.7
SD	8	97.9	68.2	33.3
SI	1	95.7	27.3	4.6
SI	8	94.6	45.7	11.2

Table 6: *Speaker Dependant (SD) and Speaker Independent (SI) Results, Exemplar DB with 80 distractors and N_{pos} matches.*

4.3. Error Analysis

Since we opt to operate in a high-precision regime our system tends to make recall errors, due to short or low-similarity exemplars.

A different type of errors, on correct accepts, are introduced in the patching process (Sec. 2.5). When examining our outputs we find that many of these mistakes are related to missing possessive (e.g. "rafi" instead of "rafi's"), deletion of articles (e.g. "the madison" is corrected to "madson" instead of "the madson") and repeated words (insertions) for a multi-word exemplars, aligned to too little audio. These could be avoided by replacing our heuristic patching process with a learned one. We leave this as future work.

5. Conclusions

In this paper we presented a flexible, Audio-Exemplar (AE) Contextualization method for E2E ASR that not require architectural changes and has only a small number of hyper parameters. We demonstrated the effectiveness of AE, especially in a speaker-dependent setup. We further demonstrated that our method can be used jointly with a NAM model to achieve further gains.

6. Acknowledgements

The authors would like to thank Trevor Strohman, Zelin Wu, Pat Rondon and Ricky Rosen for their advice and suggestions.

7. References

[1] A. Narayanan, R. Prabhavalkar, C.-C. Chiu, et al., "Recognizing Long-Form Speech Using Streaming End-to-End Models," in *Proc. ASRU*, 2019.

[2] Petar Aleksic, Cyril Allauzen, David Elson, Aleksandar Kracun, Diego Melendo Casado, and Pedro J Moreno, "Improved recognition of contact names in voice commands," in *ICASSP*. IEEE, 2015, pp. 5172–5175.

[3] 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 *Proc. ICASSP*, 2016.

[4] D. Zhao, T. N. Sainath, D. Rybach, et al., "Shallow-Fusion End-to-End Contextual Biasing," in *Proc. Interspeech*, 2019.

[5] G. Pundak, T. Sainath, R. Prabhavalkar, A. Kannan, and D. Zhao, "Deep Context: End-to-End Contextual Speech Recognition," in *Proc. SLT*, 2018.

[6] Uri Alon, Golan Pundak, and Tara N Sainath, "Contextual speech recognition with difficult negative training examples," in *ICASSP*. IEEE, 2019, pp. 6440–6444.

[7] 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," 2021.

[8] Duc Le, Mahaveer Jain, Gil Keren, Suyoun Kim, Yangyang Shi, Jay Mahadeokar, Julian Chan, Yuan Shangguan, Christian Fuegen, Ozlem Kalinli, et al., "Contextualized streaming end-to-end speech recognition with trie-based deep biasing and shallow fusion," *arXiv preprint arXiv:2104.02194*, 2021.

[9] Antoine Bruguier, Rohit Prabhavalkar, Golan Pundak, and Tara N Sainath, "Phoebe: Pronunciation-aware contextualization for end-to-end speech recognition," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6171–6175.

[10] A. Graves, "Sequence Transduction with Recurrent Neural Networks," *CoRR*, vol. abs/1211.3711, 2012.

[11] Christopher Li, Pat Rondon, Diamantino Caseiro, Leonid Velikovich, Xavier Velez, and Petar Aleksic, "Improving entity recall in automatic speech recognition with neural embeddings," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 7353–7357.

[12] Guoguo Chen, Carolina Parada, and Tara N. Sainath, "Query-by-example keyword spotting using long short-term memory networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5236–5240.

[13] Carolina Parada, Abhinav Sethy, and Bhuvana Ramabhadran, "Query-by-example spoken term detection for oov terms," in *2009 IEEE Workshop on Automatic Speech Recognition Understanding*, 2009, pp. 404–409.

[14] Yanbo Xu, Olivier Siohan, David Simcha, Sanjiv Kumar, and Hank Liao, "Exemplar-based large vocabulary speech recognition using k-nearest neighbors," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5167–5171.

[15] Khe Chai Sim, Françoise Beaufays, Arnaud Benard, Dhruv Guliani, Andreas Kabel, Nikhil Khare, Tamar Lucassen, Petr Zadrazil, Harry Zhang, Leif Johnson, Giovanni Motta, and Lillian Zhou, "Personalization of end-to-end speech recognition on mobile devices for named entities," in *ASRU*, 2019.

[16] Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al., "Similarity search in high dimensions via hashing," in *Vldb*, 1999, vol. 99, pp. 518–529.

[17] Donald J Berndt and James Clifford, "Using dynamic time warping to find patterns in time series.," in *KDD workshop*. Seattle, WA, USA:, 1994, vol. 10, pp. 359–370.

[18] A. Narayanan, T. N. Sainath, R. Pang, et al., "Cascaded encoders for unifying streaming and non-streaming ASR," in *Proc. ICASSP*, 2021.

[19] J. Yu, C.-C. Chiu, B. Li, et al., "FastEmit: Low-latency Streaming ASR with Sequence-level Emission Regularization," in *Proc. ICASSP*, 2021.

[20] A. Narayanan, A. Misra, K.C. Sim, et al., "Toward Domain-Invariant Speech Recognition via Large Scale Training," in *Proc. SLT*, 2018.